Partially Observable Mean Field Reinforcement Learning

Sriram Ganapathi Subramanian
University of Waterloo
Waterloo, Canada
s2ganapa@uwaterloo.ca

Matthew E. Taylor
University of Alberta, Dept. of Computing Science
Alberta Machine Intelligence Institute (Amii)
Edmonton, Canada
matthew.e.taylor@ualberta.ca

Mark Crowley
University of Waterloo
Waterloo, Canada
mcrowley@uwaterloo.ca

Pascal Poupart
University of Waterloo, Waterloo, Canada
Vector Institute, Toronto, Canada
ppoupart@uwaterloo.ca

ABSTRACT

Traditional multi-agent reinforcement learning algorithms are not scalable to environments with more than a few agents, since these algorithms are exponential in the number of agents. Recent research has introduced successful methods to scale multi-agent reinforcement learning algorithms to many agent scenarios using mean field theory. Previous work in this field assumes that an agent has access to exact cumulative metrics regarding the mean field behaviour of the system, which it can then use to take its actions. In this paper, we relax this assumption and maintain a distribution to model the uncertainty regarding the mean field of the system. We consider two different settings for this problem. In the first setting, only agents in a fixed neighbourhood are visible, while in the second setting, the visibility of agents is determined at random based on distances. For each of these settings, we introduce a Q-learning based algorithm that can learn effectively. We prove that this Q-learning estimate stays very close to the Nash Q-value (under a common set of assumptions) for the first setting. We also empirically show our algorithms outperform multiple baselines in three different games in the MAgents framework, which supports large environments with many agents learning simultaneously to achieve possibly distinct goals.

KEYWORDS

Multi-Agent Reinforcement Learning, Reinforcement Learning, Mean Field Theory, Partial Observation

ACM Reference Format:
Sriram Ganapathi Subramanian, Matthew E. Taylor, Mark Crowley, and Pascal Poupart. 2021. Partially Observable Mean Field Reinforcement Learning. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 9 pages.

1 INTRODUCTION

Multi-agent systems involve several learning agents that are learning simultaneously in an environment to solve a task or satisfy an objective. These agents may have to compete or cooperate with each other in the given context. Multi-agent systems are non stationary [9], making it hard to derive learning policies that are as effective as in the single agent context. As the number of learning agents increases, the possible number of learning situations in the environment increases exponentially. Many algorithms introduced in the Multi-Agent Reinforcement Learning (MARL) literature suffer from scalability issues [4, 13], and hence are typically not well suited for environments in which agents are infinite in the limit, called *many agent systems*. Mean field theory has been used to scale MARL to many agent scenarios in previous research efforts [31, 32], most of which have assumed some notion of aggregation that is made available by an engine or is directly observable in the environment. For example, Guo et al. [7] assume that a population distribution parameter can be obtained from the game engine and Yang et al. [32] assume that the mean action of all agents in the environment can be observed directly by all agents.

Partial observability is an important research area in single agent reinforcement learning (RL) [8, 11, 34], but these advances are not applicable to the many agent RL paradigm, since the stationary environment assumption is broken. Also, partial observation in single agent RL corresponds only to partial observability of state features, but in multi-agent systems this could also correspond to partial observability of other agents.

This paper relaxes the assumption that agents observe the aggregate state variable in a mean field update. Instead, we maintain a belief over the aggregate parameter that is used to help agent action selection. We focus on discrete state and action space Markov decision processes (MDPs) and modify the update rules from Yang et al. [32] to relax the assumptions of (1) global state availability and (2) exact mean action information for all agents. We consider two settings in this paper. The *Fixed Observation Radius (FOR)* setting assumes that all agents in each agent's small field of view are always observed (and those outside the radius are not). The *Probabilistic Distance-based Observability (PDO)* setting relaxes FOR such that we model the probability of an agent seeing another agent as a function of the distance between them (and this distribution defines what agents are "viewable"). We introduce a new Q-learning algorithm for both settings, addressing the **Partially Observable Mean Field (POMF) Q-learning** problem, using Bayesian updates to maintain a distribution over the mean action parameter.

This paper's contributions are to (1) introduce two novel POMF settings, (2) introduce two novel algorithms for these settings, (3) prove that the first algorithm ends up close to the Nash Q-value

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.

[10], and (4) empirically show that both algorithms outperform existing baselines in three complex, large-scale tasks. We will assume stationary strategies as do other related previous work [10] [32]. Our full paper with appendices is available on arXiv [24].

2 BACKGROUND CONCEPTS

Reinforcement learning [25] is a problem formulated on top of MDPs $\langle S, A, P, R\rangle$, where S is the state space that contains the environmental information accessible to an agent at each time step, A gives the actions that the agent can take at each time step, the reward function R provides real-valued rewards at each time step, and the transition dynamics P is the probability of moving to a state s' when the agent takes action a at state s. Q-learning [30] learns a policy ($S \mapsto A$) by updating Q-values based on experience.

Stochastic games generalize from single agent to N-agent MDPs. Each step in a stochastic game (a stage game) depends on the experiences of the agents in previous stages. A N-player stochastic game is defined as a tuple $\langle S, A^1, \ldots, A^N, R^1, \ldots, R^N, P, \gamma\rangle$, where S is the state space, A^j is the action space of agent j, and $R^j : S \times A^1 \cdots \times A^N \to \Re$ is the reward function of agent j. Agents maximize their discounted sum of rewards with $\gamma \in [0,1)$ as the discount factor. From this formulation, it can be seen that agents can have completely different reward functions (competitive and competitive-cooperative games) or can agree to maintain a shared reward structure (cooperative games). Transition function $p : S \times A^1 \cdots \times A^N \to \Omega(S)$, returns the probability distribution over the next state ($\Omega(S)$) when the system transitions from state s given actions $(a^1, \ldots, a^N)$ for all agents. The joint action is $\boldsymbol{a} \triangleq [a^1, \ldots, a^N]$. The transition probabilities are assumed to satisfy $\sum_{s'} p(s'|s, a^1, \cdots, a^N) = 1$. The joint policy (strategy) of agents can be denoted by $\boldsymbol{\pi} \triangleq [\pi^1, \ldots, \pi^N]$. Given an initial state s, the value function of agent j is the expected cumulative discounted reward given by $v^j_{\boldsymbol{\pi}}(s) = \sum_{t=0}^{\infty} \gamma^t \mathbb{E}_{\boldsymbol{\pi},p}[r^j_t|s_0 = s]$. The Q-function can then be formulated as $Q^j_{\pi}(s, \boldsymbol{a}) = r^j(s, \boldsymbol{a}) + \gamma \mathbb{E}_{s'\sim p}[v^j_{\pi}(s')]$, where s' represents the next state.

Nash Q-learning: Hu and Wellman [10] extended the Nash equilibrium solution concept in game theory to stochastic games. The Nash equilibrium of a general sum stochastic game is defined as a tuple of strategies $(\pi^1_*, \cdots, \pi^N_*)$, such that for all $s \in S$ [10],

$$v^i(s, \pi^1_*, \cdots, \pi^i_*, \cdots, \pi^N_*) \geq v^i(s, \pi^1_*, \cdots, \pi^i, \cdots, \pi^N_*) \quad \forall \pi^i \in \Pi^i$$

Here, v^i denotes the value function of agent i. This implies that no agent can deviate from its equilibrium strategy and get a strictly higher payoff when all other agents are playing their equilibrium strategies. The Nash Q-function, $Q^i_*(s, \boldsymbol{a})$, is the sum of agent i's current reward and its discounted future rewards when all agents follow the Nash equilibrium strategy. Hu and Wellman proved that under a set of assumptions, the Nash operator defined by $\mathscr{H}^{\text{Nash}}\boldsymbol{Q}(s, \boldsymbol{a}) = \mathbb{E}_{s'\sim p}[\boldsymbol{r}(s, \boldsymbol{a}) + \gamma \boldsymbol{v}^{\text{Nash}}(s')]$ converges to the Q value of the Nash equilibrium. Here, $\boldsymbol{Q} \triangleq [Q^1, \ldots, Q^N]$, $\boldsymbol{r}(s, \boldsymbol{a}) = [r^1(s, \boldsymbol{a}), \ldots, r^N(s, \boldsymbol{a})]$ and $\boldsymbol{v}^{Nash}(s) \triangleq [v^1_{\boldsymbol{\pi}^*}(s), \ldots, v^N_{\boldsymbol{\pi}^*}(s)]$.

Mean field reinforcement learning extends the stochastic game framework to environments where the number of agents are infinite in the limit [14]. All agents are assumed to be indistinguishable and independent from each other. In this case, all the agents in the environment can be approximated as a single virtual agent to which the learning agent (called the *central agent*) formulates best response strategies. Yang et al. [32] approximates the multi-agent Q-function by the mean field Q-function (MFQ) using an additive decomposition and Taylor's expansion (Eq. 1).

$$Q^j(s_t, \mathbf{a}_t) \approx Q^j(s_t, a^j_t, \overline{a}^j_t) \quad (1)$$

The MFQ is recurrently updated using Eqs. 2 – 5:

$$Q^j(s_t, a^j_t, \overline{a}^j_t) = (1-\alpha)Q^j(s_t, a^j_t, \overline{a}^j_t) + \alpha[r^j_t + \gamma v^j(s_{t+1})] \quad (2)$$

$$\text{where } v^j(s_{t+1}) = \sum_{a^j_{t+1}} \pi^j(a^j_{t+1}|s_{t+1}, \overline{a}^j_t) Q^j(s_{t+1}, a^j_{t+1}, \overline{a}^j_t) \quad (3)$$

$$\overline{a}^j_t = \frac{1}{N^j} \sum_{k \neq j} a^k_t, a^k_t \sim \pi^k(\cdot|s_t, \overline{a}^k_{t-1}) \quad (4)$$

$$\text{and } \pi^j(a^j_t|s_t, \overline{a}^j_{t-1}) = \frac{\exp(-\beta Q^j(s_t, a^j_t, \overline{a}^j_{t-1}))}{\sum_{a^{j'}_t \in A^j} \exp(-\beta Q^j(s_t, a^{j'}_t, \overline{a}^j_{t-1}))} \quad (5)$$

r^j_t is the reward for agent j, s_t is the (global) old state, s_{t+1} is the (global) resulting state, α is the learning rate, v^j is the value function of j and β is the Boltzmann parameter. a^j_t denotes the (discrete) action of agent j represented as a one-hot encoding whose components are one of the actions in the action space. The $\overline{a}^j_t$ is the mean action of all other agents apart from j and π denotes the Boltzmann policy. In Eq. 3, there is no expectation over $\overline{a}^j$, because Yang et al. [32] guaranteed that the MFQ updates will be greedy in the limit ($t \to \infty$). Finally, N^j is the number of agents in the neighbourhood of j. We highlight that, for the mean action calculation in Eq. 4, the policies of all other agents needs to be maintained by the central agent. Now, this policy can only be obtained by observing all other agents at every time step, which is a strong assumption in a large environment with many agents. Yang et al., overcome this problem by introducing *neighbourhoods*. However, the neighbourhood needs to be large enough to contain the whole environment for these methods to work, as agents can go in and out of the neighbourhoods and go out of vicinity otherwise, which will make the computation of mean action as in Eq. 4 inapplicable. In our work, we will relax this strong assumption in estimating the mean field action. We will not assume the observability of all other agents.

3 PARTIALLY OBSERVABLE MEAN FIELD Q-LEARNING: FOR

In this section, we study the Fixed Observation Radius (FOR) version of our problem, where all agents within a fixed neighbourhood from the central agent are visible to the central agent, and the others are not visible. Our setting is same as that in Yang et al. [32] but we proceed to relax the assumption of global state observability. We modify the update in Eqs. 2 – 5 by maintaining a categorical distribution for the mean action parameter ($\overline{a}$). We will only use the local state s^j of agent j and not the global state. Eq. 6 gives our corresponding Q update equation. Since the conjugate prior of a categorical distribution is the Dirichlet distribution, we use a Dirichlet prior for this parameter. Let L be the size of the action space. Let η denote the parameters of the Dirichlet $(\eta_1, \ldots, \eta_L)$, θ denote a categorical distribution $(\theta_1, \ldots, \theta_L)$, and $\mathcal{X}$ denote an observed action sample $(x_1, \ldots, x_G)$ of G agents. Then the Dirichlet for agent j can be given by $\mathcal{D}^j(\theta|\eta) \propto \theta_1^{\eta_1-1}\cdots\theta_L^{\eta_L-1}$ and the likelihood is given by $p(\mathcal{X}|\theta) \propto \theta_1^{[\mathcal{X}=1]}\cdots\theta_L^{[\mathcal{X}=L]} \propto \theta_1^{c_1}\cdots\theta_L^{c_L}$, where $[X = i]$ is the Iverson bracket, which evaluates to 1 if $X = i$ and 0 otherwise. This value corresponds to the number of occurrences of each category $(c_1, \ldots, c_L)$, denoted by c. Using a Bayesian update, the posterior is a Dirichlet distribution given by Eq. 7 where the parameters of this Dirichlet are given by $\mathcal{D}^j(\theta|\eta + c)$.

The modified Q updates are:

$$Q^j(s_t^j, a_t^j, \tilde{a}_t^j) = (1-\alpha)Q^j(s_t^j, a_t^j, \tilde{a}_t^j) + \alpha[r_t^j + \gamma v(s_{t+1}^j)] \quad (6)$$

$$\mathcal{D}^j(\theta) \propto \theta_1^{\eta_1-1+c_1}\cdots\theta_L^{\eta_L-1+c_L}; \quad \mathcal{D}^j(\theta|\eta+c) \quad (7)$$

$$\text{Where } v^j(s_{t+1}^j) = \sum_{a_{t+1}^j} \pi^j(a_{t+1}^j|s_{t+1}^j, \tilde{a}_t^j)Q^j(s_{t+1}^j, a_{t+1}^j, \tilde{a}_t^j) \quad (8)$$

$$\tilde{a}_{i,t}^j \sim \mathcal{D}^j(\theta; \eta + c); \quad \tilde{a}_t^j = \frac{1}{\mathcal{S}}\sum_{i=1}^{i=\mathcal{S}} \tilde{a}_{i,t}^j \quad (9)$$

$$\text{and } \pi^j(a_t^j|s_t^j, \tilde{a}_{t-1}^j) = \frac{\exp(-\beta Q^j(s_t^j, a_t^j, \tilde{a}_{t-1}^j))}{\sum_{a_t^{j'}\in A^j}\exp(-\beta Q^j(s_t^j, a_t^{j'}, \tilde{a}_{t-1}^j))} \quad (10)$$

We have replaced the mean field aggregation from Yang et al. (Eq. 4) with the Bayesian updates of the Dirichlet distribution from Eq. 7 and we take $\mathcal{S}$ samples from this distribution in Eq. 9 to estimate the partially observable mean action ($\tilde{a}$). This approach relaxes the assumption of complete observability of the global state. We use samples from the Dirichlet to introduce noise in the mean action parameter, enabling further exploration and helping agents to escape local optima. Being stuck in a local optimum is one of the major reasons for poor performance of learning algorithms in large scale systems. For example, Guo et al. [7] show that MFQ and Independent Q-learning (IL) [27] remain stuck at a local optimum and do not move towards a global optimum in many settings, even after many training episodes. Yang et al. also report that the Mean Field Actor Critic (MFAC) and MFQ algorithms may remain stuck at a local optimum for a long period of training episodes in a simple Gaussian squeeze environment as the number of agents becomes exponentially large. Intuitively, this problem is even worse in a partially observable setting as the agents get a smaller observation and their best response policy is directed towards this observation sample. Sampling methods as in Eq. 9 are also used in established algorithms like Thompson sampling [19, 28]. Finally, we update the Boltzmann policy like Yang et al. in Eq. 10. We provide more theoretical guarantees for our update equations in Section 6.

This version of our problem is generally applicable to many different environments. However, in some domains, agents may not be able to see all the other agents in the vicinity, but closer agents will have a high probability of being seen. The next section considers a new version of our problem where some special kinds of distributions are used to model the observed agents.

4 PARTIALLY OBSERVABLE MEAN FIELD Q-LEARNING: PDO

This section considers the Probabilistic Distance-based Observability (PDO) problem, assuming that each agent can observe other agents with some probability that decreases as distance increases.

We introduce a distance vector $\mathscr{D}$ that represents the distance of other agents in the environment to the central agent. Hence, $\mathscr{D} = (d_1, \ldots, d_N)$, where d_i denotes the distance of agent i from the central agent. We use the exponential distribution to model the probability of the distance of agent i from the central agent. Exponential distribution assigns a higher probability to smaller distances and this probability exponentially drops off as distance increases. Since, in a large environment the agents that matter are closer to the central agent, than far off, the exponential distribution is appropriate to model this variable. We drop the subscript of d for clarity. This distribution is parameterized by $\hat{\theta}$ so that $d|\hat{\theta} \propto \exp(\hat{\theta})$. Since the conjugate prior of the exponential distribution is the gamma distribution, we use a gamma prior, and the prior distribution is parameterised by α, β. Hence, we write $\hat{\theta} \propto Gamma(\hat{\alpha}, \hat{\beta})$.

We also maintain an additional parameter b_i that determines whether a given agent i is visible to the central agent j. The variable b_i takes two values: 1 if this agent is in the field of view and 0 if this agent is not in the field of view. Again, we will drop the subscript of b. We maintain a Bernoulli distribution conditioned on the distance d. The probability that an agent at a distance d is visible is given by $Pr(b = 1|d, \lambda) = \lambda e^{-d\lambda}$. Note that this is not an exponential distribution, but rather a Bernoulli distribution with a probability defined by the same algebraic formula as the exponential distribution. In this setting, we will assume that the central agent will see varying numbers of other agents based on this distribution. Since the parameter λ cannot be estimated by an agent from observation (the agent needs to know which other agents it is seeing and not seeing to infer λ), we will assume that the scalar value of λ is common knowledge for all the agents. Since $\lambda e^{-\lambda d}$ should be in $[0, 1]$ because it is a probability, only λ values in $[0, 1]$ satisfy this requirement. We will fix the value of λ to be 1, but it could be any other value in the given range. This gives a definite distribution that determines the conditional of b. We are particularly interested in the posterior term $Pr(\hat{\theta}|d, b = 1)$, which denotes the probability of $\hat{\theta}$, given the distance d of another agent i, and that i is in the field of view of the central agent j.

$$Pr(\hat{\theta}|d, b=1) \propto Pr(d|\hat{\theta}, b=1)Pr(\hat{\theta}|b=1) \propto Pr(d|\hat{\theta}, b=1)Pr(\hat{\theta}) \quad (11)$$

In the last term of Eq. 11, the variable θ does not depend on the variable b, so we remove the conditional. Now consider,

$$\begin{aligned}
&Pr(d|\hat{\theta}, b=1) = Pr(d, b|\hat{\theta})/Pr(b|\hat{\theta}) \\
&= Pr(b=1|\hat{\theta}, d)Pr(d|\hat{\theta})/Pr(b=1|\hat{\theta}) \\
&= Pr(b=1|d)Pr(d|\hat{\theta})/Pr(b=1|\hat{\theta}) \\
&= \lambda e^{-d}\lambda\hat{\theta}e^{-d\hat{\theta}}/\int_d Pr(b=1|d)Pr(d|\hat{\theta}) \\
&= e^{-d}\hat{\theta}e^{-d\hat{\theta}}/\int_{d=0}^{d=\infty}\hat{\theta}e^{-d\hat{\theta}}\lambda e^{-d\lambda} = e^{-d}\hat{\theta}e^{-d\hat{\theta}}/\int_{d=0}^{d=\infty}\hat{\theta}e^{-d(\hat{\theta}+1)} \\
&= e^{-d}\hat{\theta}e^{-d\hat{\theta}}(\hat{\theta}+1)/\hat{\theta}
\end{aligned} \quad (12)$$

Applying Eq. 12 in Eq. 11,

$$\begin{aligned}
&Pr(\hat{\theta}|d, b=1) \propto Gamma(\hat{\alpha}, \hat{\beta}) \times e^{-d}\hat{\theta}e^{-d\hat{\theta}}(\hat{\theta}+1)/\hat{\theta} \\
&\propto \hat{\theta}^{\hat{\alpha}-1}e^{-\hat{\beta}\hat{\theta}} \times e^{-d}\hat{\theta}e^{-d\hat{\theta}}(\hat{\theta}+1)/\hat{\theta} \\
&\propto \hat{\theta}^{\hat{\alpha}}e^{-\hat{\theta}(\hat{\beta}+d-d/\hat{\theta})} + \hat{\theta}^{\hat{\alpha}-1}e^{-\hat{\theta}(\hat{\beta}+d-d/\hat{\theta})}
\end{aligned}$$

The posterior of $\hat{\theta}$ is therefore given by a mixture of Gamma distributions (i.e., $Gamma(\hat{\alpha}+1, d+\hat{\beta}-d/\hat{\theta})$ and $Gamma(\hat{\alpha}, d+\hat{\beta}-d/\hat{\theta})$). We can obtain a single Gamma posterior, corresponding to a projection of this mixture of Gammas, by updating the new value of $\hat{\alpha}$ as $\hat{\alpha}+0.5$. We sample from this single Gamma distribution to get a new parameter $\overline{\lambda}$ (Eq. 15). We denote the Gamma distribution for the agent j using superscript j (Eq. 14). Hence, we get:

$$Q^j(s_t^j, a_t^j, \tilde{a}_t^j, \overline{\lambda}_t^j) = (1-\alpha)Q^j(s_t^j, a_t^j, \tilde{a}_t^j, \overline{\lambda}_t^j) + \alpha[r_t^j + \gamma v^j(s_{t+1}^j)] \tag{13}$$

$$\hat{\theta}_P \propto Gamma^j(\hat{\alpha}+1, d+\hat{\beta}-d/\hat{\theta}) + Gamma^j(\hat{\alpha}, d+\hat{\beta}-d/\hat{\theta}) \tag{14}$$

Where:

$$\overline{\lambda}_{i,t}^j \sim [Gamma^j(\hat{\alpha}+0.5, d+\hat{\beta}-d/\hat{\theta})]; \quad \overline{\lambda}_t^j = \frac{1}{\mathcal{G}}\sum_{i=1}^{i=\mathcal{G}} \overline{\lambda}_{i,t}^j \tag{15}$$

$$v^j(s_{t+1}^j) = \sum_{a_{t+1}^j} \pi^j(a_{t+1}^j | s_{t+1}^j, \tilde{a}_t^j, \overline{\lambda}_t) Q^j(s_{t+1}^j, a_{t+1}^j, \tilde{a}_t^j, \overline{\lambda}_t^j) \tag{16}$$

$$\pi^j(a_t^j | s_t^j, \tilde{a}_{t-1}^j, \overline{\lambda}_{t-1}^j) = \frac{\exp(-\beta Q^j(s_t^j, a_t^j, \tilde{a}_{t-1}^j, \overline{\lambda}_{t-1}^j))}{\sum_{a_t^{j'} \in A^j} \exp(-\beta Q^j(s_t^j, a_t^{j'}, \tilde{a}_{t-1}^j, \overline{\lambda}_{t-1}^j))} \tag{17}$$

All the variables above have the same meaning as in Eqs. 6 – 10. The estimate of the $\tilde{a}$ is obtained as in Eq. 9. The $\hat{\theta}_P$ denotes the new (posterior) value of $\hat{\theta}$. The $\overline{\lambda}$ parameter is updated by sampling from the Gamma distribution, as in Eq. 15, by taking $\mathcal{G}$ samples.

5 ALGORITHM IMPLEMENTATIONS

The implementation of POMFQ follows prior work [32] that uses neural networks — Q-functions are parameterized using weights ϕ, but tabular representations or other function approximators should also work. Our algorithms are an integration of the respective update equations with Deep Q-learning (DQN) [18]. Algorithm 1 gives pseudo code for the algorithm for the "FOR" case and Algorithm 2 for the "PDO" case. The lines in Algorithm 2 that have changed from Algorithm 1 are marked in blue. We provide a complexity analysis of our algorithms in Appendix F.

6 THEORETICAL RESULTS

The goal of this section is to show that our FOR Q-updates are guaranteed to converge to the Nash Q-value. We will begin by providing a technical result that is generally applicable for any stochastic processes of which the Q-function is a specific example. Then we have a sequence of theorems that lead us to bound the difference between the POMF Q-value and the Nash Q-value in the limit ($t \longrightarrow \infty$). We outline a number of common assumptions that are needed to prove these theorems. For the purposes of a direct comparison of the POMF Q-function and the Nash Q-function, we assume that we have a system of N agents where agents have the full global state available and thus have the ability to perform a MFQ update (Eqs. 2 – 5) or a POMFQ update (Eqs. 6 – 10). By the definition of a Nash equilibrium, every agent should have the knowledge of every other agent's strategy. To recall, in a Nash equilibrium, no agent will have an incentive to unilaterally deviate, given the knowledge of other agent strategies. Our objective is also to make a direct comparison between the POMFQ update and the MFQ update and hence we will use the FOR setting algorithms of POMFQ update in the theoretical analysis as it is most directly related to MFQ. In this section, we will show that a representative agent j's Q-value will remain at least within a small distance of the Nash Q-value in the limit ($t \longrightarrow \infty$) as it performs a POMFQ update, which tells us that, in the worst case, the agents stay very close to the Nash equilibrium. We have provided a proof sketch for all our theorems in this section while the complete versions of our proofs can be found in Appendix A. In a mean field setting, the homogeneity of agents allows us to drop the agent index j [14] for

Algorithm 1 Partially observable mean field Q Learning - FOR

Initialize the weights of Q-functions $Q_{\phi^j}, Q_{\phi_-^j}$ for all agents $j \in 1, \ldots, N$.
Initialize the Dirichlet parameter $\mathcal{D}^j(\theta)$ for all agents j.
Initialize the mean action $\overline{a}^j$ for each agent $j \in 1, \ldots, N$.
Initialize the total steps (T) and total episodes (E).
while Episode < E **do**
  while Step < T **do**
    For each agent j, sample a^j from the policy induced by Q_{ϕ^j} according to Eq. 10 with the current mean action $\tilde{a}^j$ and the exploration rate β.
    For each agent j, update its Dirichlet distribution (Eq. 7).
    For each agent j, compute the new mean action $\tilde{a}^j$ (Eq. 9).
    Execute the joint action $\mathbf{a} = [a^1, \ldots, a^N]$. Observe the rewards $\mathbf{r} = [r^1, \ldots, r^N]$ and the next state $\mathbf{s'} = [s'^1, \ldots, s'^N]$.
    Store $\langle \mathbf{s}, \mathbf{a}, \mathbf{r}, \mathbf{s'}, \tilde{\mathbf{a}} \rangle$ in replay buffer B, where $\tilde{\mathbf{a}} = [\tilde{a}^1, \ldots, \tilde{a}^N]$ is the mean action.
  end while
  while j = 1 to N **do**
    Sample a minibatch of K experiences $\langle \mathbf{s}, \mathbf{a}, \mathbf{r}, \mathbf{s'}, \tilde{\mathbf{a}} \rangle$ from B.
    Set $y^j = r^j + \gamma v_{\phi_-^j}^{POMF}(s')$ according to Eq. 8.
    Update Q network by minimizing the loss $L(\phi^j) = \frac{1}{k}\sum(y^j - Q_{\phi^j}(s^j, a^j, \tilde{a}^j))^2$.
  end while
  Update params of target network for each agent j: $\phi_-^j \leftarrow \tau\phi^j + (1-\tau)\phi_-^j$.
end while

Algorithm 2 Partially observable mean field Q Learning - PDO

Initialize the weights of Q functions $Q_{\phi^j}, Q_{\phi_-^j}$ for all agents $j \in 1, \ldots, N$.
Initialize the Dirichlet parameter θ in $\mathcal{D}^j(\theta)$ for all agents j.
Initialize $\hat{\alpha}$ and $\hat{\beta}$ in $Gamma^j(\hat{\alpha}, \hat{\beta})$ for all agents $j \in 1, \ldots, N$.
Initialize the mean action $\overline{a}^j$ for each agent $j \in 1, \ldots, N$.
Initialize the total steps (T) and total episodes (E).
while Episode < E **do**
  while Step < T **do**
    For each agent j sample a^j from the policy induced by Q_{ϕ^j} according to Eq. 17 with the current mean action $\tilde{a}^j$ and the exploration rate β.
    For each agent j update its Dirichlet distribution (Eq. 7).
    For each agent j, update its Gamma distribution (Eq. 14).
    For each agent j, compute the new mean action $\tilde{a}^j$ (Eq. 9).
    For each agent j, update parameter $\overline{\lambda}$ (Eq. 15).
    Execute the joint action $\mathbf{a} = [a^1, \ldots, a^N]$. Observe the rewards $\mathbf{r} = [r^1, \ldots, r^N]$ and the next state $\mathbf{s'} = [s'^1, \ldots, s'^N]$.
    Store $\langle \mathbf{s}, \mathbf{a}, \mathbf{r}, \mathbf{s'}, \tilde{\mathbf{a}}, \overline{\boldsymbol{\lambda}} \rangle$ in replay buffer B, s.t. $\tilde{\mathbf{a}} = [\tilde{a}^1, ..., \tilde{a}^N]$, $\overline{\boldsymbol{\lambda}} = [\overline{\lambda}^1, ..., \overline{\lambda}^N]$
  end while
  while j = 1 to N **do**
    Sample minibatch of K experiences $\langle \mathbf{s}, \mathbf{a}, \mathbf{r}, \mathbf{s'}, \tilde{\mathbf{a}}, \overline{\boldsymbol{\lambda}} \rangle$ from B.
    Set $y^j = r^j + \gamma v_{\phi_-^j}^{POMF}(s')$ according to Eq. 16.
    Update Q network by minimizing $L(\phi^j) = \frac{1}{k}\sum(y^j - Q_{\phi^j}(s^j, a^j, \overline{a}^j, \overline{\lambda}))^2$.
  end while
  Update params of target network for each agent j: $\phi_-^j \leftarrow \tau\phi^j + (1-\tau)\phi_-^j$.
end while

the value and Q-function, which we adopt for clarity. Also, "w.p.1" represents "with probability one".

Consider an update equation of the following form (using the Tsitsiklis [29] formulation):

$$x_i(t+1) = x_i(t) + \alpha_i(t)(F_i(x^i(t)) - x_i(t) + w_i(t)) \quad (18)$$

Here, $x(t)$ is the value of vector x at time t and $x_i(t)$ denotes its ith component. Let, F be a mapping from $\mathscr{R}^n$ into itself. Let $F_1, \ldots, F_n : \mathscr{R}^n \rightarrow \mathscr{R}$ be the component mappings of F, that is $F(x) = (F_1(x), \ldots, F_n(x))$ for all $x \in \mathscr{R}^n$. Also, $w_i(t)$ is a noise term, and $x^i(t)$ can be defined as $x^i(t) = (x_1(\tau_1^i(t)), \cdots, x_n(\tau_n^i(t)))$, where each $\tau_j^i(t)$ satisfies $0 \leq \tau_j^i(t) \leq t$.

Next, we state some assumptions. The first three are the same as those in Tsitsiklis [29], but we modify the fourth assumption. The first assumption guarantees that old information is eventually discarded with probability one. The second assumption is a measurability condition and the third assumption is the learning rate condition, both of which are common in RL [26] [29]. The Assumption 4, is a condition on the F mapping, which is a weaker version than the fourth assumption in Tsitsiklis [29].

Assumption 1. *For any i and j, $\lim_{t\to\infty} \tau_j^i(t) = \infty$ w.p.1.*

Assumption 2. *a) $x(0)$ is $\mathcal{F}(0)$-measurable*
b) For every i, j, and t, $w_i(t)$ is $\mathcal{F}(t+1)$-measurable
c) For every i, j, and t, $\alpha_i(t)$ and $\tau_j^i(t)$ are $\mathcal{F}(t)$-measurable
d) For every i and t, we have $\mathbb{E}[w_i(t)|\mathcal{F}(t)] = 0$
e) For deterministic constants A and B,

$$\mathbb{E}[w_i^2(t)|\mathcal{F}(t)] \leq A + Bmax_j max_{\tau \leq t}|x_j(\tau)|^2$$

Assumption 3. *The learning rates satisfy $0 \leq \alpha_i(t) < 1$.*

Assumption 4. *a) The mapping F is monotone; that is, if $x \leq y$, then $F(x) \leq F(y)$*
b) The mapping F is continuous
c) In the limit ($t \to \infty$), the mapping F is bounded in an interval $[x^* - D, x^* + D]$, where x^* is some arbitrary point
d) If $e \in \mathcal{R}^n$ is the vector with all components equal to 1, and p is a positive scalar then, $F(x) - pe \leq F(x - pe) \leq F(x + pe) \leq F(x) + pe$

Now, we will state our first theorem. Theorem 1 is a technical result that we obtain by extending Theorem 2 in Tsitsiklis [29]. We will use this result to derive the main result in Theorem 4.

Theorem 1. *A stochastic process of the form given in Eq. 18 remains bounded in the range $[x^* - 2D, x^* + 2D]$ in the limit, if Assumptions 1 – 4 hold, and if the process is guaranteed not to diverge to infinity. D is the bound on the F mapping in Assumption 4(c).*

Proof (Sketch). Since the stochastic process in Eq. 18 is guaranteed to stay bounded (Assumption 4(c)), one can find other processes that lower bounds and upper bounds this process. Let us assume that we can show that the process in Eq. 18 always stays bounded by these two processes after some finite time t (that is for all $t' \geq t$). Now, if we can prove that the process A is upper bounded by a finite value, this value will be the upper bound of the process in Eq. 18 after t as well. Similarly, the lower bound of L will be its lower bound (after time t). □

Now we state three more assumptions, as used earlier [32].

Assumption 5. *Each action-value pair is visited infinitely often and the reward stays bounded.*

Assumption 6. *The agents' policy is Greedy in the Limit with Infinite Exploration (GLIE).*

Assumption 7. *The Nash equilibrium can be considered a global optimum or a saddle point in every stage game of the stochastic game.*

Assumption 5 is very common in RL [26]. Assumption 6 is needed to ensure that the agents are rational [32] and this is satisfied for POMFQ as the Boltzmann policy is known to be GLIE [21]. Assumption 7 has been adopted by previous researchers [10, 32]. Hu and Wellman [10] consider this to be a strong assumption, but they note that this assumption is needed to prove convergence in theory, even though it is not needed to observe convergence in practice.

Let $\tilde{a}_i$ be a component of vector $\tilde{a}$ and $\overline{a}_i$ be a component of vector $\overline{a}$. Now, we make a comparison between mean actions of the MFQ update (Eq. 4) and the POMFQ update (Eq. 9).

Theorem 2. *The MFQ mean action and the POMFQ mean action both satisfy*

$$|\tilde{a}_{i,t} - \overline{a}_{i,t}| \leq \sqrt{\frac{1}{2n} \log \frac{2}{\delta}}$$

as time $t \to \infty$, with probability >= δ, where n is the number of samples observed. $\tilde{a}$ is the mean action as obtained from the Dirichlet in Eq. 9 and $\overline{a}$ is the mean action in Eq. 4.

Proof (Sketch). This theorem is an application of the Hoeffding's bound which provides a probabilistic bound for the difference between the sample mean and the the true mean of a distribution. As $\tilde{a}$ is an empirical mean of the samples n observed at each time step, the Hoeffding's bound is applied to obtain the result. □

Theorem 3. *When the Q-function is Lipschitz continuous (with constant M) with respect to mean actions, then the POMF Q-function will satisfy the following relationship:*

$$|Q^{POMF}(s_t, a_t, \tilde{a}_{t-1}) - Q^{MF}(s_t, a_t, \overline{a}_{t-1})| \leq M \times L \times \log \frac{2}{\delta} \times \frac{1}{2n} \quad (19)$$

as $t \to \infty$ with probability $\geq (\delta)^{L-1}$, where $L = |A|$ and n is the number of samples.

Proof (Sketch). Once we have the bound on the mean actions of POMFQ update and MFQ update as in Theorem 2, with the assumption of Lipschitz continuity, a corresponding bound can be derived for the respective Q-functions too. This is done by applying the bound of the mean actions in the Lipschitz condition. □

From Theorem 3, we can see that in a similar setting, the POMFQ updates will not see a significant degradation in performance as compared to the MFQ updates. The probability of this holding is inversely proportional to the size of the action space available to each agent. In Theorem 3, the bound is between two Q-functions with the same state and action, but with different mean actions. Let $Z = M \times L \times \log \frac{2}{\delta} \times \frac{1}{2n}$ and from Theorem 3, $|Q^{POMF}(s_t, a_t, \tilde{a}_{t-1}) - Q^{MF}(s, a_t, \overline{a}_{t-1})| \leq Z$. Now, we would like to directly compare the value estimates of POMFQ and MFQ updates. Consider two different actions a^j and b^j for agent j. Under the assumption that the mean field Q-function is K-Lipschitz continuous with respect to actions,

$$|Q^{MF}(s_t^j, a_t^j, \overline{a}_{t-1}^j) - Q^{MF}(s_t^j, b_t^j, \overline{a}_{t-1}^j)| \leq K|a_t^j - b_t^j| \leq K\sqrt{2} \quad (20)$$

In the last step, we applied the fact that all components of a^j and b^j are less than or equal to 1 (a one hot encoding). Assume that the optimal action for Q^{POMF} is a^* and for Q^{MF} is b^*. Now consider,

$$\begin{aligned}
&|v^{POMF}(s_{t+1}) - v^{MF}(s_{t+1})| \\
&= |\max_{a_{t+1}} Q^{POMF}(s_{t+1}, a_{t+1}, \tilde{a}_t) - \max_{b_{t+1}} Q^{MF}(s_{t+1}, b_{t+1}, \overline{a}_t)| \\
&= |Q^{POMF}(s_{t+1}, a_{t+1}^*, \tilde{a}_t) - Q^{MF}(s_{t+1}, a_{t+1}^*, \overline{a}_t) \\
&+ Q^{MF}(s_{t+1}, a_{t+1}^*, \overline{a}_t) - Q^{MF}(s_{t+1}, b_{t+1}^*, \overline{a}_t)| \leq Z + K\sqrt{2} \triangleq D
\end{aligned} \quad (21)$$

In the first step we apply the fact that the Boltzmann policy will become greedy in the limit ($t \to \infty$). The last step is coming from Eqs. 19 and 20. We also reiterate that the Lipschitz continuity assumptions on the Q-function are consistent with prior work [32].

THEOREM 4. *When we update the Q functions using the partially observable update rule in Eq. 6, the process satisfies the condition in the limit ($t \to \infty$):*

$$|Q^*(s_t, \boldsymbol{a}_t) - Q^{POMF}(s_t, a_t, \tilde{a}_t)| \leq 2D$$

when Assumptions 3, 5, and 7 hold. Here Q^* is the Nash Q-value and D is the bound for value functions in Eq. 21. This holds with probability at least δ^{L-1}, where $L = |A|$.

PROOF (SKETCH). This result is an application of Theorem 1, where we show that all the assumptions of Theorem 1 are satisfied by the conditions in this theorem. □

It is important to note that we need only three minor assumptions (Assumptions 3, 5, and 7) to hold for Theorem 4, which is our main theoretical result. Theorem 4 shows that the POMFQ updates stay very close to the Nash equilibrium in the limit ($t \to \infty$). The lower bound on the probability of this is high for a small action space and low for a large action space. In a multi-agent setting, the Q-updates are in the form of POMFQ updates, and do not have the (intuitive) effect of having any fixed point as commonly seen in RL. Theorem 4 proves our update rule is very close to the Nash equilibrium, a stationary point for the stochastic game. Hence, the policy in Eq. 5 is approximately close to this stationary point, which guarantees that it becomes (asymptotically) stationary in the limit ($t \to \infty$).

The distance between the POMF Q-function and the Nash Q-function is inversely proportional to the number of samples from the Dirichlet (n). If the agent chooses to take a large number of samples, the POMF Q-estimate is very close to the Nash Q-estimate, but this may lead to a degradation in performance due to having no additional exploratory noise as discussed in Section 3. In MARL, the Nash equilibrium is not a guarantee of optimal performance, but only a fixed point guarantee. The (self-interested) agents would still take finite samples, for better performance. To balance the theory and performance, the value of n should not be too high nor too low.

Appendix C provides an experimental illustration of Theorem 4 in the Ising model, a mathematical model used to describe the magnetic spins of atomic particles. This model was also used in [32]. We show that the distance (error) of the POMF Q-function (tabular implementation of the FOR updates) from the Nash Q-function stays bounded after a finite number of episodes as in Theorem 4.

7 EXPERIMENTS AND RESULTS

This section empirically demonstrates that using POMFQ updates will result in better performance in a partially observable environment than when using the MFQ updates. All the code for the experiments is open sourced [5].

We design three cooperative-competitive games for each of the two problems (FOR and PDO) within the MAgent framework [33] to serve as testbeds. We will provide the important elements of these experimental domains here, while the comprehensive details (including exact reward functions and hyperparameter values) are deferred to Appendix D. For all games, we have a two stage process: training and faceoff (test). We consider four algorithms for all the games: MFQ, MFAC, IL, and POMFQ. In each stage, there are two groups of agents: group A and group B. Since the agents do not know what kind of opponents they will see in the faceoff stage, they train themselves against another group that plays the same algorithm in the training stage. Thus, in the training stage, each algorithm will train two networks (groups A and B). In the faceoff stage, groups trained by different algorithms fight against each other. Our formulation is consistent with past research using the MAgent framework [6, 32]. We plot the rewards obtained by group A in each episode for the training stage (group B also shows similar trends — our games are not zero sum) and the number of games won by each algorithm in the faceoff stage. For statistical significance, we report p-values of an unpaired 2-sided t-test for particular episodes in the training stage and a Fischer's exact test for the average performances in the faceoff stage. We treat p-values of less than 0.05 as statistically significant differences. The tests are usually conducted between POMFQ and next best performing algorithm in the final episode of training for the training results.

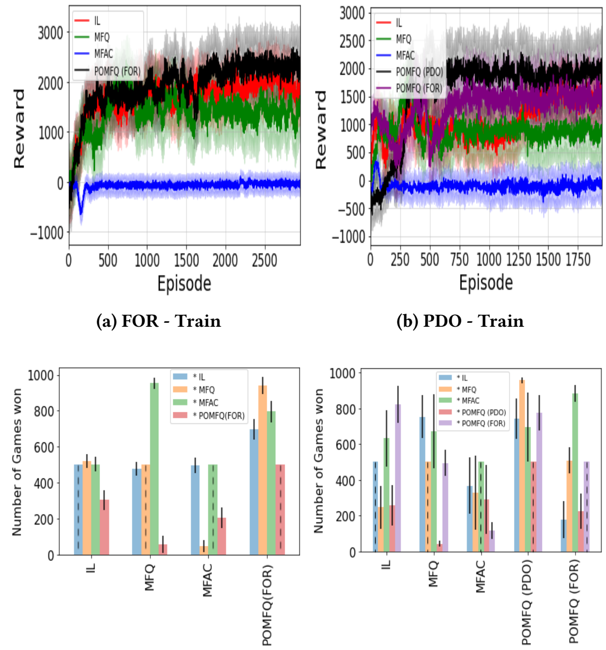
The Multibattle game has two groups of agents fight against each other. There are 25 agents in each group for a total of 50. Agents learn to cooperate within the group and compete across the group to win. We analyse both the FOR and PDO cases. In FOR, information about nearby agents is available, but agents further than 6 units are hidden. In PDO, the game engine maintains a Bernoulli distribution of visibility of each agent from every other agent as discussed in Section 4. Based on this probability, each agent in PDO could see different numbers of other agents at each time step. MFQ and MFAC use a frequentist strategy where agents observed at each time step are aggregated (Eq. 4) to obtain a mean action. We run 3000 episodes of training in FOR and 2000 episodes in PDO. Each episode has a maximum of 500 steps. For the faceoff, group A trained using the first algorithm and group B trained using the second algorithm fight against each other for 1000 games. We report all results as an average of 20 independent runs for both training and faceoff (with standard deviation). In our experiments, an average of 6 − 8 agents out of 50 agents are visible to the central agent at a given time step (averaged over the length of the game). Note that we use 50 agents per game, more agents are used in previous research [6, 32]. In our case, the ratio of agents seen vs. the total number of agents matters more than the simple absolute number of agents in the competition.

In the FOR setting of the Multibattle domain (Figure 1 (a)) the POMFQ algorithm plays the FOR variant (Algorithm 1). POMFQ dominates other baselines from about 1800 episodes ($p < 0.3$) until the end ($p < 0.03$). We see that MFAC quickly falls into a poor local optimum and never recovers. The poor performance of MFAC in the MAgent games, compared to the other baselines, is consistent with previous work [6, 32]. Faceoff in the FOR case (Figure 1(c)) shows that POMFQ wins more than 50% of the games against others ($p < 0.01$). An ablation study in Appendix E shows that performance improves with increase in viewing distance.

In the PDO setting, we use both the FOR variant of POMFQ algorithm (no $\overline{\lambda}$ parameter) and the PDO variant of POMFQ algorithm (Algorithm 2). We differentiate these two algorithms in the legends of Figures 1(b) and 1(d). The FOR variant loses out to the PDO algorithm that explicitly tracks the $\overline{\lambda}$ parameter ($p < 0.02$). If an algorithm bases decisions only on $\tilde{a}$, as in Algorithm 1, the agents do not know how risk seeking or risk averse their actions should be (when agents nearby are not visible). In this game, agents can choose to make an attack (risk seeking) or a move (risk averse). The additional parameter $\overline{\lambda}$ helps agents understand the uncertainty in not seeing some agents when making decisions. The PDO algorithm takes a lead over the other algorithms from roughly 900 episodes ($p < 0.04$) and maintains the lead until the end ($p < 0.03$). In faceoff, PDO wins more than 50% (500) of the games against all other algorithms as seen in Figure 1(d) ($p < 0.01$).

The second game, Battle-Gathering, is similar to the Multibattle game where a set of two groups of 50 agents are fighting against each other to win a battle, but with an addition of food resources scattered in the environment. All the agents are given an additional reward when capturing food, in addition to killing the competition (as in Multibattle). The training and faceoff are conducted similar to Multibattle game. Figure 2(a) shows that the POMFQ algorithm dominates the other three algorithms from about 900 episodes ($p < 0.03$) till the end ($p < 0.01$). In the comparative battles (Figure 2(c)), POMFQ has a clear lead over other algorithms ($p < 0.01$). MFQ and IL are similar in performance and MFAC loses to all other algorithms. We also observe this in the PDO domain train ($p < 0.02$, Figure 2(b)) and test experiments ($p < 0.01$, Figure 2(d)).

The third game is a type of Predator-Prey domain, where there are two groups — predators and prey. There are a total of 20 predator agents and 40 prey agents in our domain. The predators are stronger than the prey and have an objective of killing the prey. The prey are faster than the predator and try to escape from the predators. The training is conducted and rewards are plotted using the same procedure as in the Multibattle domain. Training performances are in Figures 3(a) and 3(c). The standard deviation of the performance in this game is considerably higher than the previous two games because we have two completely different groups that are trying to outperform each other in the environment. At different points in training, one team may have a higher performance than the other, and this lead can change over time. In the first setting, Figure 3(a), we can see that the POMFQ (FOR) shows a small lead over other baseline algorithms (at the end of training, $p < 0.1$). In the direct faceoff (Figure 3(c)), POMFQ wins more games than the other algorithms ($p < 0.01$), In the PDO setting too, the POMFQ-PDO algorithm has an edge over the others during the training phase ($p < 0.4$) and the testing ($p < 0.01$)(Figure 3(b) and 3(d)). As the p-values for the training suggest, POMFQ can be seen to have a better performance, but the results are not statistically significant. The faceoff results, on the other hand, are statistically significant ($p < 0.01$). We have run training for 2000 games and faceoff for 1000 games in the last two domains.



(a) FOR - Train (b) PDO - Train

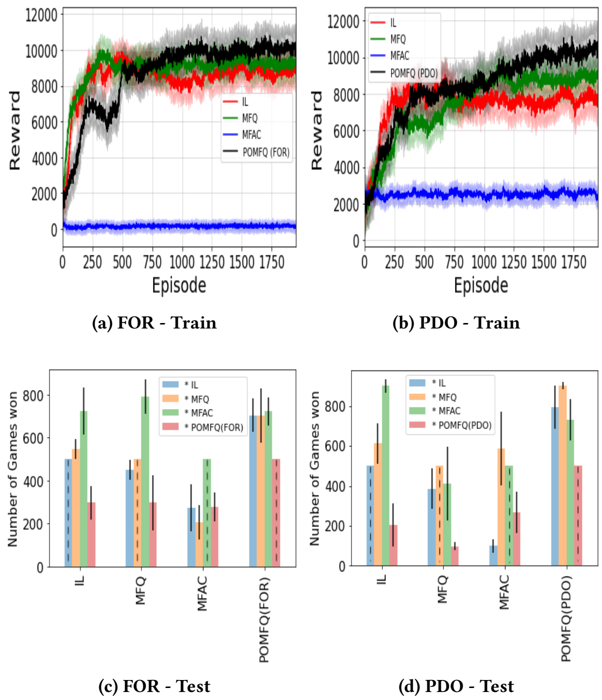
(c) FOR - Test (d) PDO - Test

Figure 1: Multibattle results. The * in the legend of test plots denotes the opponent. For example, first orange bar (from the left) in the bar plots is result for IL. vs MFQ. The dashed lines indicate bars that we set for symmetry. We do not run faceoff experiments between the same algorithm.

In three semantically different domains, we have shown that in the partially observable case, the MFQ and MFAC algorithms using frequentist strategies do not provide good performances. Also, the frequentist strategies (MFQ and MFAC) have worse performance in the harder PDO domain compared to the FOR domain. Sometimes, they also lose out to a simpler algorithm that does not even track the mean field parameter (IL). The FOR and PDO algorithms gives the best performance across both settings, as evidenced by the training and the test results. The training results clearly show that POMFQ never falls into a very poor local optimum like MFAC often does. The test results show that in a direct face-off, POMFQ outperforms all other algorithms. The p-values indicate that our results are statistically significant. Additionally, in Appendix B, we provide comparisons of the POMFQ - FOR and PDO algorithms with two more baselines, recurrent versions of IL and MFQ, in the same three MAgent domains and the results show that POMFQ has clear advantages compared to these recurrent baselines as well.

8 RELATED WORK

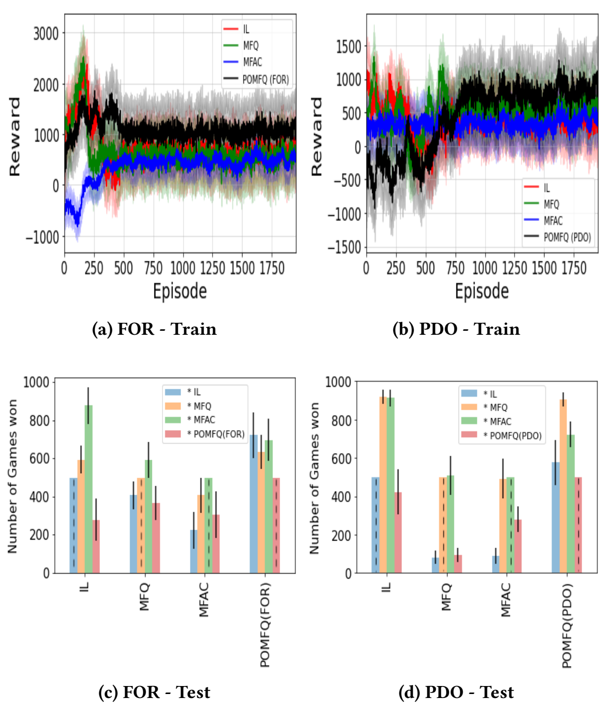
Mean field games were introduced by Lasry and Lions [14], extending mean field theory [20, 22] to the stochastic games framework. The stochastic games formulation was obtained by extending MDPs to MARL environments [10, 15]. Recent research has actively used the mean field games construct in a MARL setting, allowing



(a) FOR - Train (b) PDO - Train

(c) FOR - Test (d) PDO - Test

Figure 2: Training and faceoff results of Battle-Gathering game.



(a) FOR - Train (b) PDO - Train

(c) FOR - Test (d) PDO - Test

Figure 3: Training and faceoff results of Predator-Prey game.

tractable solutions in environments in which many agents participate. Model-based solutions have also been tried in this setting [12], but the model is specific to the application domain and these methods do not generalize well. Subramanian and Mahajan [23] analyze the problem using a stationary mean field. In contrast to our approach, this paper needs strict assumptions regarding this stationarity, which do not hold in practice. Mguni et al. [16] approaches this problem using the fictitious play technique. They provide strong theoretical properties for their algorithms, but only in the finite time horizon case. These results do not directly hold for infinite horizons. Additionally, strict assumptions on the reward function and fictitious property [1] assumption makes their algorithms less generally applicable. In fictitious play, each agent assumes that its opponents are playing stationary strategies. Thus, the response of each agent is a best response to the empirical frequency of their opponents. Another work by the same authors [17] introduces an algorithm and provides theoretical analysis for the mean field learning problem in cooperative environments. Our methods, on the other hand, work for both cooperative and competitive domains. Along the same lines, the work by Elie et al. [2, 3] contributes fictitious play based techniques to solve mean field games with general theoretical properties based on quantifying the errors accumulated at each time step. However, the strict assumptions on the reward function in addition to the fictitious play assumption is also present in the work by Elie et al. In our work, the agents do not make the fictitious play assumption for best responses. Yang et al. [32] do not have the limitations of other works noted here, but it assumes the global state is observable for all agents and a local action is taken from it. This has been relaxed by us.

9 CONCLUSION

This paper considers many agent RL problems where the exact cumulative metrics regarding the mean field behaviour is not available and only local information is available. We used two variants of this problem and provided practical algorithms that work in both settings. We empirically showed that our approach is better than previous methods that used a simple aggregate of neighbourhood agents to estimate the mean field action. We theoretically showed that POMFQ stays close to the Nash Q under common assumptions.

In future work, we would like to relax some assumptions about the Bayesian approach using conjugate priors and make our analysis more generally applicable. Additionally, different observation distributions could allow the direction of view to determine the "viewable" agents, such as when agents in front of another agent are more likely to be seen than agents behind it.

10 ACKNOWLEDGEMENTS

Resources used in preparing this research were provided by the province of Ontario and the government of Canada through CIFAR, NSERC and companies sponsoring the Vector Institute. Part of this work has taken place in the Intelligent Robot Learning Lab at the University of Alberta, which is supported in part by research grants from the Alberta Machine Intelligence Institute (Amii), CIFAR, and NSERC.

REFERENCES

[1] Ulrich Berger. 2007. Brown's original fictitious play. *Journal of Economic Theory* 135, 1 (2007), 572–578.

[2] Romuald Elie, Julien Pérolat, Mathieu Laurière, Matthieu Geist, and Olivier Pietquin. 2019. Approximate fictitious play for mean field games. *arXiv preprint arXiv:1907.02633* (2019).

[3] Romuald Elie, Julien Perolat, Mathieu Laurière, Matthieu Geist, and Olivier Pietquin. 2020. On the Convergence of Model Free Learning in Mean Field Games. In *AAAI Conference on Artificial Intelligence (AAAI 2020)*.

[4] Jakob Foerster, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. 2018. Learning with opponent-learning awareness. In *Proceedings of the 17th International Conference on Autonomous Agents and Multi-Agent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 122–130.

[5] Sriram Ganapathi Subramanian. [n.d.]. Partially Observable Mean Field Reinforcement Learning. https://github.com/Sriram94/pomfrl

[6] Sriram Ganapathi Subramanian, Pascal Poupart, Matthew E. Taylor, and Nidhi Hegde. 2020. Multi Type Mean Field Reinforcement Learning. In *Proceedings of the Autonomous Agents and Multi Agent Systems (AAMAS 2020)*. IFAAMAS, Auckland, New Zealand.

[7] Xin Guo, Anran Hu, Renyuan Xu, and Junzi Zhang. 2019. Learning mean-field games. In *Advances in Neural Information Processing Systems*. 4967–4977.

[8] Matthew Hausknecht and Peter Stone. 2015. Deep recurrent Q-learning for partially observable MDPs. In *2015 AAAI Fall Symposium Series*.

[9] Pablo Hernandez-Leal, Bilal Kartal, and Matthew E Taylor. 2019. A survey and critique of multiagent deep reinforcement learning. *Autonomous Agents and Multi-Agent Systems* 33, 6 (2019), 750–797.

[10] Junling Hu and Michael P Wellman. 2003. Nash Q-learning for general-sum stochastic games. *Journal of machine learning research* 4, Nov (2003), 1039–1069.

[11] Peter Karkus, David Hsu, and Wee Sun Lee. 2017. QMDP-net: Deep learning for planning under partial observability. In *Advances in Neural Information Processing Systems*. 4694–4704.

[12] Arman C Kizilkale and Peter E Caines. 2012. Mean field stochastic adaptive control. *IEEE Trans. Automat. Control* 58, 4 (2012), 905–920.

[13] Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Pérolat, David Silver, and Thore Graepel. 2017. A unified game-theoretic approach to multiagent reinforcement learning. In *Advances in Neural Information Processing Systems*. 4190–4203.

[14] Jean-Michel Lasry and Pierre-Louis Lions. 2007. Mean field games. *Japanese journal of mathematics* 2, 1 (2007), 229–260.

[15] Michael L Littman. 1994. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*. Elsevier, 157–163.

[16] David Mguni, Joel Jennings, and Enrique Munoz de Cote. 2018. Decentralised learning in systems with many, many strategic agents. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[17] David Mguni, Joel Jennings, Emilio Sison, Sergio Valcarcel Macua, Sofia Ceppi, and Enrique Munoz de Cote. 2019. Coordinating the Crowd: Inducing Desirable Equilibria in Non-Cooperative Systems. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 386–394.

[18] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.

[19] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. 2016. Deep exploration via bootstrapped DQN. In *Advances in neural information processing systems*. 4026–4034.

[20] P Ring. 1996. Relativistic mean field theory in finite nuclei. *Progress in Particle and Nuclear Physics* 37 (1996), 193–263.

[21] Satinder Singh, Tommi Jaakkola, Michael L Littman, and Csaba Szepesvári. 2000. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine learning* 38, 3 (2000), 287–308.

[22] H Eugene Stanley and Guenter Ahlers. 1973. Introduction to phase transitions and critical phenomena. *Physics Today* 26 (1973), 71.

[23] Jayakumar Subramanian and Aditya Mahajan. 2019. Reinforcement learning in stationary mean-field games. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 251–259.

[24] Sriram Ganapathi Subramanian, Matthew E. Taylor, Mark Crowley, and Pascal Poupart. 2020. Partially Observable Mean Field Reinforcement Learning. (2020). arXiv:2012.15791 [cs.MA]

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

[26] Csaba Szepesvári and Michael L Littman. 1999. A unified analysis of value-function-based reinforcement-learning algorithms. *Neural Computation* 11, 8 (1999), 2017–2060.

[27] Ming Tan. 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*. 330–337.

[28] William R Thompson. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 25, 3/4 (1933), 285–294.

[29] John N Tsitsiklis. 1994. Asynchronous stochastic approximation and Q-learning. *Machine learning* 16, 3 (1994), 185–202.

[30] Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning* 8, 3-4 (1992), 279–292.

[31] Jiachen Yang, Xiaojing Ye, Rakshit Trivedi, Huan Xu, and Hongyuan Zha. 2018. Learning Deep Mean Field Games for Modeling Large Population Behavior. In *International Conference on Learning Representations (ICLR)*.

[32] Yaodong Yang, Rui Luo, Minne Li, Ming Zhou, Weinan Zhang, and Jun Wang. 2018. Mean Field Multi-Agent Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, Stockholmsmässan, Stockholm Sweden, 5567–5576.

[33] Lianmin Zheng, Jiacheng Yang, Han Cai, Ming Zhou, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Magent: A many-agent reinforcement learning platform for artificial collective intelligence. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[34] Pengfei Zhu, Xin Li, Pascal Poupart, and Guanghui Miao. 2017. On improving deep reinforcement learning for POMDPs. *arXiv preprint arXiv:1704.07978* (2017).