Information State Embedding in Partially Observable Cooperative Multi-Agent Reinforcement Learning

Weichao Mao, Kaiqing Zhang, Erik Miehling, Tamer Başar

***Abstract*— Multi-agent reinforcement learning (MARL) under partial observability has long been considered challenging, primarily due to the requirement for each agent to maintain a belief over all other agents' local histories – a domain that generally grows exponentially over time. In this work, we investigate a partially observable MARL problem in which agents are cooperative. To enable the development of tractable algorithms, we introduce the concept of an *information state embedding* that serves to compress agents' histories. We quantify how the compression error influences the resulting value functions for decentralized control. Furthermore, we propose an instance of the embedding based on recurrent neural networks (RNNs). The embedding is then used as an approximate information state, and can be fed into any MARL algorithm. The proposed *embed-then-learn* pipeline opens the black-box of existing (partially observable) MARL algorithms, allowing us to establish some theoretical guarantees (error bounds of value functions) while still achieving competitive performance with many end-to-end approaches.**

I. Introduction

Multi-agent reinforcement learning (MARL) is a prominent and widely applicable paradigm for modeling multi-agent sequential decision making under uncertainty, with applications in a wide range of domains including robotics [1], cyber-physical systems [2], and finance [3]. Many practical problems require agents to make decisions based on only a partial view of the environment and the (private) information of other agents, e.g., intention of pedestrians in autonomous driving settings, or location of surveillance targets in military drone swarm applications. This partial information generally precludes agents from making optimal decisions due to the requirement that each agent maintains a belief over all other agents' local histories – a domain that, in general, grows exponentially in time [4].

We consider a partially observable setting, but restrict attention to problems in which the agents are cooperative, that is, they share the same objective.[1] Even under this simpler (cooperative) setting, the primary challenge still exists: due to the lack of explicit communication, agents possess noisy and asymmetric information about the environment, yet their rewards depend on the joint actions of all agents. As in the general (non-cooperative) setting, this informational coupling requires agents to maintain a growing amount of information.

Many approaches have been proposed to address this challenge; we roughly categorize them into two classes: *concurrent learning* and *centralized learning*. In concurrent learning approaches, agents learn and update their own control policies simultaneously. However, since reward and state processes are coupled, the environment becomes non-stationary from the perspective of each agent, and hence concurrent solutions do not converge in general [5]. On the other hand, centralized learning approaches, as the name suggests, reformulate the problem from the perspective of a centralized (or virtual) coordinator [6]. Despite its popularity [7], [8], the centralized approach suffers from high computational complexity. A centralized algorithm needs to assign an action to each possible history sequence of the agents, and the cardinality of such sequences grows exponentially over time. In fact, decentralized partially observable Markov decision processes (Dec-POMDPs), an instance of the general decentralized control model, are known to be NEXP-complete [9].

In this paper, we propose to address the computational issues in the centralized approach by extracting a summary, termed an *information state embedding*, from the history space, and then learning control policies in the compressed embedding space that possess some quantifiable performance guarantee. This procedure, which we term the *embed-then-learn* pipeline, is depicted in Fig. 1. In the first stage, an embedding scheme with a quantifiable compression error is extracted from the history space, where our metric of compression error, to be defined in Section III, favors an embedding with higher predictive ability. In the second stage, we learn a policy in this compressed embedding space. We prove how the embedding error propagates over time, and our theoretical results provide an overall performance bound of the policy in terms of the compression error. Therefore, in this paradigm, the cooperative MARL problem can be reduced to one of finding an information state embedding with a small compression error.

We also introduce an empirical instance of the information state embedding, and demonstrate how to extract an embedding from data. Despite the empirical results that *end-to-end* learning[2] generally leads to state-of-the-art performance [10], our approach breaks a partially observable MARL problem into two stages: embedding followed by learning. It provides

Research supported in part by the US Army Research Laboratory (ARL) Cooperative Agreement W911NF-17-2-0196, and in part by the Office of Naval Research (ONR) MURI Grant N00014-16-1-2710. Authors are with the Coordinated Science Laboratory, University of Illinois, Urbana, IL 61801, U.S.A. Email addresses: {weichao2, kzhang66, miehling, basar1}@illinois.edu.

[1] This is in contrast with the more challenging general case of non-cooperative agents in which agents may act strategically to achieve their (individual) goals.

[2] In deep learning, end-to-end learning generally means training one single neural network that takes in the raw data as input and outputs the ultimate goal task.

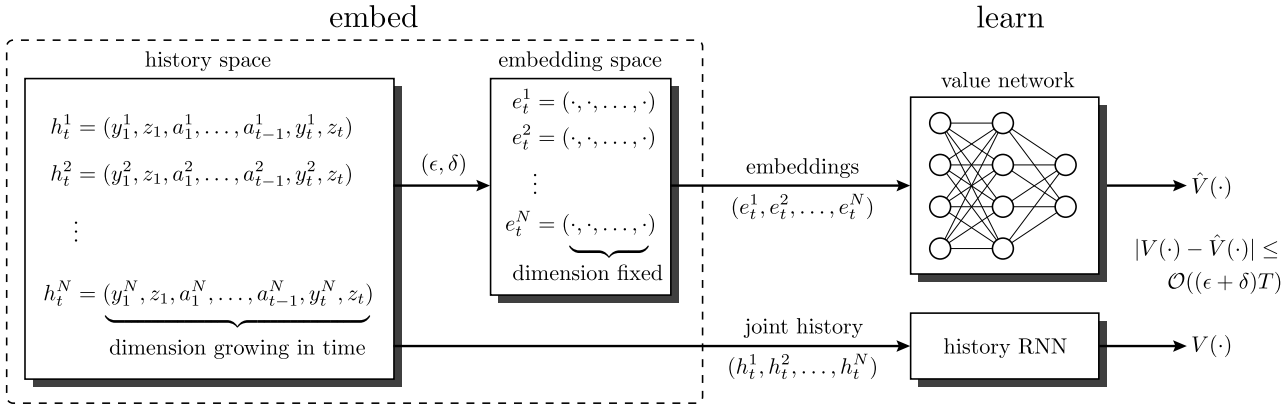

Fig. 1: The *embed-then-learn* pipeline. First, the *embed* stage extracts a compact summary, given ϵ and δ tolerances (as discussed in Definition 3), from the history space, followed by the *learn* stage in which embedding-based policies are learned. Note that the lower path (the history-based policies) is not actually carried out; it is only illustrated to describe the bound.

a new angle that also enjoys fair performance, while allowing for a more theoretical foundation. This approach opens the *black-box* in end-to-end approaches, and makes an initial step toward understanding their great empirical successes.

Related Work. In the seminal work [11], it was proved that a decentralized problem can be converted to a centralized standard form and hence be solved via a dynamic program. Following this paradigm, the common information approach [6] formulates the decentralized control problem as a single agent POMDP based on the common information of all the agents. A similar work [7] transforms the Dec-POMDP into a continuous-state MDP, and again demonstrates that standard techniques from POMDPs are applicable. More recently, [12] has introduced a sufficient information approach that investigates the sufficient conditions under which an information state is optimal for decision making. However, this approach does not offer a constructive algorithm of the sufficient information state. In fact, it is generally difficult, in multi-agent settings, to determine whether an information state that is more compact than the entire history exists. Our work extends this approach, in that we *learn* or *extract* a compact information state *from data* by approximately satisfying these sufficient conditions for optimal decision-making. Perhaps most related to our work is [13], where the authors also studied error propagation for approximations in Dec-POMDPs. Our work differs and complements [13] in that [13] approximates the Bellman backup operator and the transition matrix, while we focus on the approximation of the information state.

Learning a state representation that is more compact than an explicit history is also of great interest even in single agent partially observable systems. When the underlying system has an inherent state, as in POMDPs, it is helpful to directly learn a generative model of the system [14], [15]. When an inherent system state is not available, it is still possible to learn an internal state representation that captures the system dynamics. Two well-known approaches are predictive state representation (PSR) [16] and causal state representation learning [17]. Interested readers are referred to [18] for a detailed survey of state representation learning in control. A recent work [19] points out that PSR is not sufficient for RL problems, and proposes to extend PSR by introducing a set of sufficient conditions for performance evaluation. In comparison, we generalize their analysis to the multi-agent setting, where our work can be regarded as learning a state representation in a *decentralized* partially observable system.

There is also no lack of empirical/heuristic approaches for MARL under partial observability. In concurrent learning, a learning scheme is proposed in [20] where agents take turns to learn the optimal response to others' policies. The authors in [5] extend single-agent RL algorithms to the multi-agent setting and empirically evaluate their performance. Another work [21] studies multi-task MARL and introduces a multi-agent extension of experience replay [22]. A centralized learning algorithm, termed oSARSA, is proposed in [8] to solve a centralized continuous-state MDP formulated from the MARL problem. For a more detailed survey, please refer to [23]. Compared with those more end-to-end solutions, our embed-then-learn pipeline enables more theoretical analysis.

Contribution. We summarize our contributions as follows: **1)** We address the computational issue in partially observable MARL, by compressing each agent's local history to an information state embedding that lies in a fixed domain. **2)** Given the compression error of an embedding, we prove that the value function defined on the embedding space has bounded approximation error. **3)** We propose to learn an embedding instance using RNNs, and empirically evaluate its performance on some benchmark Dec-POMDP tasks.

Notation. We use uppercase letters (e.g., X) to denote random variables, lowercase letters (e.g., x) for their corresponding realizations, and calligraphic letters (e.g., $\mathcal{X}$) for the corresponding spaces where the random variables reside. Subscripts denote time indices, whereas superscripts index agents. For $a < b$, the notation $a : b$ denotes $\{a, a+1, \ldots, b\}$, e.g., $X_{a:b}$ is a shorthand for $\{X_a, X_{a+1}, \ldots, X_b\}$, with a similar convention for superscripts. Finally, $\Delta(\mathcal{X})$ denotes

the set of all probability measures over $\mathcal{X}$.

II. Model and Preliminaries

We adopt a model similar to the one developed in [12], and for completeness we state it here. Consider a team of N non-strategic, i.e., cooperative, agents indexed by $i \in \mathcal{I} = \{1, 2, \dots, N\}$ over a finite horizon $\mathcal{T} = \{1, \dots T\}$. The state of the environment $X_t \in \mathcal{X}_t$, the private observation $Y_t^i \in \mathcal{Y}_t^i$ of agent i, and the common observation[3] of all agents $Z_t \in \mathcal{Z}_t$, follow the following dynamics:

$$\begin{aligned} X_{t+1} &= f_t\left(X_t, A_t^{1:N}, W_t^x\right), \\ Y_{t+1}^i &= l_{t+1}^i\left(X_{t+1}, A_t^{1:N}, W_{t+1}^i\right), \\ Z_{t+1} &= l_{t+1}^c\left(X_{t+1}, A_t^{1:N}, W_{t+1}^c\right), \end{aligned}$$

where X_1, W_t^x, W_t^i and W_t^c are independent random variables for all $t \in \mathcal{T}, i \in \mathcal{I}$. We also assume $\mathcal{X}_t, \mathcal{Y}_t^i, \mathcal{A}_t^i, \mathcal{Z}_t$ are all finite sets. The initial state X_1 follows a fixed distribution $f_X \in \Delta(\mathcal{X}_1)$. Note that the Dec-POMDP model of [25] considers the special case where the agents share no common information, i.e., $\mathcal{Z}_t$ is empty.

Define common information $C_t = \{Z_{1:t}\} \in \mathcal{C}_t$ as the aggregate of common observations from time 1 to t. Similarly, let $P_t^i = \{Y_{1:t}^i, A_{1:t-1}^i\} \backslash C_t \in \mathcal{P}_t^i$ denote each agent's *private information*, assumed to be unknown to the other agents $j \neq i$, where $\mathcal{P}_t^i$ is the space of agent i's private information at time t. Define the joint history $H_t = \{Y_{1:t}^{1:N}, A_{1:t-1}^{1:N}, Z_{1:t}\} \in \mathcal{H}_t$ to be the collection of all agents' actions and observations up to and including time t. Accordingly, define each agent's *local history* $H_t^i = \left\{Y_{1:t}^i, A_{1:t-1}^i, Z_{1:t}\right\} \in \mathcal{H}_t^i$ to be agent i's information at time t. Under the assumption of perfect recall, each agent's strategy $g_{1:T}^i$ maps its local history to a distribution over its actions, i.e., $g_t^i : \mathcal{H}_t^i \to \Delta(\mathcal{A}_t^i)$. We also refer to g_t^i as agent i's control policy at t and $g_{1:T}^i$ as agent i's control law.

Remark 1. We consider stochastic policies rather than deterministic policies throughout the paper. To understand the reason, consider the single agent case, a standard POMDP. Even though deterministic policies are known to be optimal in POMDPs, their existence relies on knowing the belief state with certainty. In any learning approach, the belief state may not be known with certainty at any time. Any approximation to the true belief state may give rise to stochastic policies that outperform deterministic policies (an extreme case is when the action is based on only the most recent observation as in [26]). Since a Dec-POMDP can be translated to an equivalent (centralized/single-agent) POMDP [6], the requirement to consider stochastic policies when only an approximate belief state is available carries over to the multi-agent setting as well.

At each time t, all the agents receive a joint reward $R_t(X_t, A_t^{1:N})$. The agents' joint objective is to maximize the total reward over the entire horizon: $R(g) = \mathbb{E}_g\left[\sum_{t=1}^T R_t(X_t, A_t^{1:N})\right]$, where the expectation is taken with respect to the probability distribution on the states induced by the joint control law $g \triangleq g_{1:T}^{1:N}$.

It has been shown in [6] that this decentralized control problem can be formulated as a centralized POMDP from the perspective of a virtual coordinator. The coordinator only has access to the common information C_t, not the agents' private information $P_t^{1:N}$. The centralized state $(X_t, P_t^{1:N}) \in \mathcal{X}_t \times \mathcal{P}_t^{1:N}$ corresponds to the environment state and agents' private information in the decentralized problem. The centralized actions, termed prescriptions (to be defined in Definition 2), are mappings from each agent's private information to a local control action, i.e., $\gamma_t^i : \mathcal{P}_t^i \to \Delta(\mathcal{A}_t^i)$. The information state in the centralized POMDP is a belief over the environment state and all the agents' private information, conditional on the common information and joint strategies (see e.g., Lemma 1 in [6]). For optimal decision making, the coordinator needs to maintain a belief over all the agents' private information, the domain of which grows exponentially over time. Therefore, the centralized approach is generally intractable. In the following, we review relevant definitions and structural results from the literature.

Definition 1 (Sufficient private and common information [12])**.** We say S_t^i, which is a function of P_t^i and C_t, denoted by $\zeta_t^i\left(P_t^i, C_t; g\right), i \in \mathcal{N}, t \in \mathcal{T}$, is sufficient private information if it satisfies the following conditions:

(a) Recursive update:

$$S_t^i = \phi_t^i\left(S_{t-1}^i, H_t^i \backslash H_{t-1}^i; g\right), \forall t \in \mathcal{T} \backslash \{1\},$$

(b) Sufficient to predict future observations:

$$\begin{aligned} &\mathbb{P}^g\{S_{t+1}^{1:N} = s_{t+1}^{1:N}, Z_{t+1} = z_{t+1} | P_t^{1:N}, C_t, A_t^{1:N}\} \\ &\quad = \mathbb{P}^g\{S_{t+1}^{1:N} = s_{t+1}^{1:N}, Z_{t+1} = z_{t+1} | S_t^{1:N}, C_t, A_t^{1:N}\}, \end{aligned}$$

$\forall\{s_{t+1}^{1:N}, z_{t+1}\} \in \mathcal{S}_{t+1}^{1:N} \times \mathcal{Z}_{t+1}$ and $\forall g \in \mathcal{G}$,

(c) Sufficient to predict future reward: For any $\widetilde{g}$ (strategies defined on the sufficient information space),

$$\mathbb{E}^{\widetilde{g}}\left[R_t \mid C_t, P_t^i, A_t^{1:N}\right] = \mathbb{E}^{\widetilde{g}}\left[R_t \mid C_t, S_t^i, A_t^{1:N}\right],$$

(d) Sufficient to predict others' private information:

$$\mathbb{P}^{\widetilde{g}}\left\{S_t^{-i} = s_t^{-i} | P_t^i, C_t\right\} = \mathbb{P}^{\widetilde{g}}\left\{S_t^{-i} = s_t^{-i} | S_t^i, C_t\right\},$$

$\forall s_t^{-i} \in \mathcal{S}_t^{-i}$ and $\forall \widetilde{g} \in \widetilde{\mathcal{G}}$.

Sufficient common information Π_t is defined as the conditional distribution over the environment state X_t and the joint private information $S_t^{1:N}$ of all the agents, given the current common information C_t and previous strategies $g_{1:t-1}$:

$$\Pi_t = \mathbb{P}^{g_{1:t-1}}\left(X_t, S_t^{1:N} \mid C_t\right).$$

Due to the above definition, an agent can make decisions using only its sufficient private information and common information. Such a decision making strategy $\sigma_t^i : \Delta(\mathcal{X}_t \times \mathcal{S}_t^{1:N}) \times \mathcal{S}_t^i \to \Delta(\mathcal{A}_t^i)$ is termed a sufficient information based (SIB) strategy for agent i at time t.

Let $\eta^{g_{1:t-1}} : \mathcal{C}_t \to \Delta(\mathcal{X}_t \times \mathcal{S}_t^{1:N})$ denote the mapping from common information to sufficient common information,

[3]This definition of common observations can be equivalently considered as the innovations defined in the partial history sharing setting [24], [6], where each agent sends a subset of its local history to a shared memory according to a predefined sharing protocol.

i.e., $\Pi_t = \eta^{g_{1:t-1}}(C_t)$. The authors of [12] showed that the SIB belief Π_t can be updated recursively via Bayes' rule, using only the previous common belief Π_{t-1} and the new common observation Z_t. That is, there exists a mapping $\psi_t^{\sigma_{t-1}} : \Delta(\mathcal{X}_{t-1} \times \mathcal{S}_{t-1}^{1:N}) \times \mathcal{Z}_t \to \Delta(\mathcal{X}_t \times \mathcal{S}_t^{1:N})$, such that $\Pi_t = \psi_t^{\sigma_{t-1}}(\Pi_{t-1}, Z_t)$. When the belief Π_t is fixed, the SIB strategy $\sigma_t^i(\Pi_t, S_t^i)$ only depends on S_t^i. This induces another function, termed prescription.

Definition 2 (Prescriptions [6])**.** A *prescription* $\gamma_t^i : \mathcal{S}_t^i \to \Delta(\mathcal{A}_t^i)$ is a mapping from agent i's sufficient private information to a distribution over actions at time t.

According to Theorem 3 in [12], given perfect information of the system dynamics (i.e., f_t, l_t^i, and l_t^c), the optimal planning solution to the decentralized control problem can be found via the following dynamic program:

$$V_{T+1}(\pi_{T+1}) = 0, \quad \forall \pi_{T+1} \in \mathit{\Pi}_{T+1}, \tag{1}$$

and at every $t \in \mathcal{T}$,

$$\begin{aligned} V_t(\pi_t) &= \max_{\gamma_t^{1:N}: \mathcal{S}_t^{1:N} \to \Delta(\mathcal{A}_t^{1:N})} Q_t(\pi_t, \gamma_t^{1:N}), \forall \pi_t \in \mathit{\Pi}_t, \\ Q_t(\pi_t, \gamma_t^{1:N}) &= \mathbb{E}\Big[R_t(X_t, \gamma_t^{1:N}(S_t^{1:N})) \\ &\quad + V_{t+1}(\psi_{t+1}^{\gamma_t}(\pi_t, Z_{t+1}))\Big|\Pi_t = \pi_t\Big], \forall \pi_t \in \mathit{\Pi}_t. \end{aligned} \tag{2}$$

In a learning problem, the system model is unknown to the agents. Agents must infer this information through interaction with the environment. The remainder of the paper is devoted to solving the learning problem.

III. Information State Embedding

The definition of sufficient information (see Definition 1) characterizes a compression of history that is sufficient for optimal decision making. However, it does not offer an explicit way to construct such an information state, nor to learn it from data. In this section, we first define the notion of an *information state embedding*, an approximate version of sufficient information, and analyze the approximation error it introduces. We further provide an explicit algorithm to learn this information state embedding from data.

A. Information State Embedding: Definition

Since a Dec-POMDP can be formulated as a centralized POMDP [6], it might seem reasonable to apply (single-agent) state representation techniques, e.g., [19], directly to the centralized problem to learn an appropriate information state embedding. However, this is not very helpful because the "state" in the centralized POMDP is derived from the common information in the decentralized problem, and hence single-agent state representation techniques would only compress common information. However, the intractability of the decentralized problem arises from the exponential growth of private information. To address the computational bottleneck in the decentralized problems, what we really need is a compact embedding of the agents' private information.

Let $\hat{S}_t^i$ denote a compression of agent i's sufficient private information S_t^i at time t, where the detailed properties of this compression will be clear later. This compression mapping is denoted by $\alpha_t^i : \mathcal{S}_t^i \to \hat{\mathcal{S}}_t^i$, and we assume that α_t^i is injective[4] for all $i \in \mathcal{I}, t \in \mathcal{T}$. It is worth noting that the injectivity assumption is also inherent in the mathematical definition of embeddings in topological spaces (see e.g., [27]).

Remark 2. We note that the injectivity assumption does not contradict the fact that α_t^i is a compression. The injective property restricts the cardinalities of the domain and the co-domain, yet compression concerns dimensionality. For computational reasons, we assume throughout the paper that $\hat{\mathcal{S}}^i$ has a fixed domain.[5] Given its private information embedding, an agent makes decisions using its *embedded strategy* defined as $\hat{g}_t^i : \hat{\mathcal{S}}_t^i \times \mathcal{C}_t \to \Delta(\mathcal{A}_t^i)$.

Define the compressed common belief $\hat{\Pi}_t$ to be the conditional distribution over the current environment state X_t and the joint private information embeddings $\hat{S}_t^{1:N}$ of all the agents, given the current common information C_t and previous embedded strategies $\hat{g}_{1:t-1}$, i.e., $\hat{\Pi}_t = \mathbb{P}^{\hat{g}_{1:t-1}}\left(X_t, \hat{S}_t^{1:N} \mid C_t\right)$. Define the embedded Bayes' rule $\hat{\psi}_t^{\hat{\gamma}_{t-1}} : \Delta(\mathcal{X}_{t-1} \times \hat{\mathcal{S}}_{t-1}) \times \mathcal{Z}_t \to \Delta(\mathcal{X}_t \times \hat{\mathcal{S}}_t)$ analogously. Following Definition 2, we define the common information compression mapping $\hat{\eta}^{\hat{g}_{1:t-1}} : \mathcal{C}_t \to \Delta(\mathcal{X}_t \times \hat{\mathcal{S}}_t^{1:N})$ and embedded prescriptions $\hat{\gamma}_t^i : \hat{\mathcal{S}}_t^i \to \Delta(\mathcal{A}_t^i)$ accordingly. Analogous to (2), we can also define a dynamic program based on our embedded information:

$$\hat{V}_{T+1}(\hat{\pi}_{T+1}) = 0, \quad \forall \hat{\pi}_{T+1} \in \hat{\mathit{\Pi}}_{T+1}, \tag{3}$$

and for every $\hat{\pi}_t$ at every $t \in \mathcal{T}$,

$$\begin{aligned} \hat{V}_t(\hat{\pi}_t) &= \max_{\hat{\gamma}_t^{1:N}} \hat{Q}_t(\hat{\pi}_t, \hat{\gamma}_t^{1:N}), \\ \hat{Q}_t(\hat{\pi}_t, \hat{\gamma}_t^{1:N}) &= \mathbb{E}\Big[R_t(X_t, \hat{\gamma}_t^{1:N}(\hat{S}_t^{1:N})) \\ &\quad + \hat{V}_{t+1}(\hat{\psi}_{t+1}^{\hat{\gamma}_t}(\hat{\pi}_t, Z_{t+1})) \mid \hat{\Pi}_t = \hat{\pi}_t\Big]. \end{aligned} \tag{4}$$

To quantify the performance of any such information-embedding, we formally define an (ϵ, δ)-information state embedding as follows.

Definition 3 ((ϵ, δ)-information state embedding)**.** We call $\{\hat{S}_t^{1:N}, \hat{\Pi}_t\}$ an (ϵ, δ)-*information state embedding* if it satisfies the following two conditions:

(a) Approximately sufficient to predict future rewards: For any $t \in \mathcal{T}$ and any realization of sufficient private information $s_t^{1:N}$, common information c_t, and actions $a_t^{1:N}$:

$$\begin{aligned} \Big|\mathbb{E}[R_t(X_t, a_t^{1:N}) \mid \pi_t, s_t^{1:N}, a_t^{1:N}] \\ - \mathbb{E}[R_t(X_t, a_t^{1:N}) \mid \hat{\pi}_t, \hat{s}_t^{1:N}, a_t^{1:N}]\Big| \leq \epsilon. \end{aligned}$$

(b) Approximately sufficient to predict future beliefs: For

[4] A function $f : X \to Y$ is injective if $\forall a, b \in X, f(a) = f(b) \Rightarrow a = b$.

[5] A fixed domain is not a theoretical requirement in our analysis, it is just desirable from a computational perspective.

any Borel subset $B \subseteq \Delta(\mathcal{X}_{t+1} \times \hat{\mathcal{S}}_{t+1})$, define

$$\mu_t(B; a_t^{1:N}) = \mathbb{P}\left(\hat{\Pi}_{t+1} \in B \mid \pi_t, s_t^{1:N}, a_t^{1:N}\right),$$
$$\nu_t(B; a_t^{1:N}) = \mathbb{P}\left(\hat{\Pi}_{t+1} \in B \mid \hat{\pi}_t, \hat{s}_t^{1:N}, a_t^{1:N}\right).$$

Then

$$\mathcal{K}\left(\mu_t(a_t^{1:N}), \nu_t(a_t^{1:N})\right) \leq \delta,$$

where $\mathcal{K}$ denotes the Wasserstein or Kantorovich-Rubinstein distance between two distributions.

By Kantorovich-Rubinstein duality [28], Definition 3 suggests $\left|\int f d\mu_t - \int f d\nu_t\right| \leq \delta$ for any Lipschitz continuous function f with Lipschitz constant $\|f\|_{Lip} \leq 1$ (with respect to the Euclidean metric). To obtain an error bound on the value function, we make the following assumption:

Assumption 1 (Lipschitz continuity of value functions)**.** Value functions $\hat{V}_t : \Delta(\mathcal{X}_t \times \hat{\mathcal{S}}_t^{1:N}) \to \mathbb{R}$ are Lipschitz continuous for all $t \in \mathcal{T}$ with Lipschitz constant upper bound L_V, i.e., $\|\hat{V}(\hat{\pi}_t) - \hat{V}(\hat{\pi}_t')\|_2 \leq L_V \|\hat{\pi}_t - \hat{\pi}_t'\|_2$.

We note that Lipschitz continuity over the compressed common information space is a mild assumption. This is because by centralizing the problem as a single agent POMDP, the value function is piecewise linear convex over the belief state [29], which is Lipschitz continuous over the (non-compressed) common information space.

Corollary 1. Combining Definition 3(b) and Assumption 1, we know that for any realization $c_t, s_t^{1:N}$ and $a_t^{1:N}$, $\forall t \in \mathcal{T}$:

$$\Big|\mathbb{E}[\hat{V}(\hat{\Pi}_{t+1}) \mid \pi_t, s_t^{1:N}, a_t^{1:N}] - \mathbb{E}[\hat{V}(\hat{\Pi}_{t+1}) \mid \hat{\pi}_t, \hat{s}_t^{1:N}, a_t^{1:N}]\Big| \leq L_V \delta.$$

B. Information State Embedding: Error Analysis

Next, we extend the approximation error analysis in [19] to the multi-agent setting. Our main result is that, by compressing the exponentially growing history to an (ϵ, δ)-information state embedding, the error of value functions over the entire horizon is bounded as stated in the theorem below:

Theorem 1. For any $t \in \mathcal{T}$ and any realization c_t and $p_t^{1:N}$, let ${\gamma_t^*}^{1:N}$ and ${\hat{\gamma}_t^*}^{1:N}$ denote the optimal prescriptions in the two dynamic programming solutions (2) and (4), respectively. Then, we have:

$$\left|Q_t(\pi_t, {\gamma_t^*}^{1:N}) - \hat{Q}_t(\hat{\pi}_t, {\hat{\gamma}_t^*}^{1:N})\right| \leq (T-t+1)(\epsilon + L_V\delta),$$
$$\left|V_t(\pi_t) - \hat{V}_t(\hat{\pi}_t)\right| \leq (T-t+1)(\epsilon + L_V\delta).$$

Proof. We prove by backward induction. As the basis of induction, Theorem 1 holds at time $T+1$ by construction. Suppose Theorem 1 holds at time $t+1$, $t \leq T$; then, for time t, we define an auxiliary set of prescriptions ${\hat{\gamma}_t^0}^{1:N}$ that produces exactly the same action distribution as ${\gamma_t^*}^{1:N}$. Specifically, for any $t \in \mathcal{T}, i \in \mathcal{I}$, and for any realization of S_t^i, this definition implies $\mathbb{P}\left(a_t^i \mid \hat{s}_t^i, \hat{\gamma}_t^{0\,i}\right) = \mathbb{P}\left(a_t^i \mid s_t^i, {\gamma_t^*}^i\right), \forall a_t^i \in \mathcal{A}_t^i$. This is possible because our private information embedding process is assumed to be injective. The existence of such an oracle ${\hat{\gamma}_t^0}^{1:N}$ suggests that, given only the embedded information, it is always possible to recover the optimal action distributions that are produced by the complete information. Nevertheless, our embedded-information-based dynamic program ends up generating a different set of prescriptions ${\hat{\gamma}_t^*}^{1:N}$, and hence the oracle ${\hat{\gamma}_t^0}^{1:N}$ is only used for analysis purposes. Together with Definition 3(a), the following holds for every realization c_t and $s_t^{1:N}$:

$$\Big|\mathbb{E}[R_t(X_t, {\gamma_t^*}^{1:N}(s_t^{1:N})) \mid \pi_t, s_t^{1:N}, {\gamma_t^*}^{1:N}] - \mathbb{E}[R_t(X_t, {\hat{\gamma}_t^0}^{1:N}(\hat{s}_t^{1:N})) \mid \hat{\pi}_t, \hat{s}_t^{1:N}, {\hat{\gamma}_t^0}^{1:N}]\Big| \leq \epsilon, \tag{5}$$

where $\gamma_t^{1:N}(s_t^{1:N})$ is a shorthand for $\{\gamma_t^1(s_t^1), \ldots, \gamma_t^N(s_t^N)\}$. Similarly, by combining the definition of ${\hat{\gamma}_t^0}^{1:N}$ with Remark 1, we also have:

$$\Big|\mathbb{E}[\hat{V}(\hat{\Pi}_{t+1}) \mid \pi_t, s_t^{1:N}, {\gamma_t^*}^{1:N}] - \mathbb{E}[\hat{V}(\hat{\Pi}_{t+1}) \mid \hat{\pi}_t, \hat{s}_t^{1:N}, {\hat{\gamma}_t^0}^{1:N}]\Big| \leq L_V\delta. \tag{6}$$

To see this, notice that for any Borel subset B of $\Delta(\mathcal{X}_{t+1} \times \hat{\mathcal{S}}_{t+1})$, we have:

$$\begin{aligned}
&\mu_t(B; {\gamma_t^*}^{1:N}) \triangleq \mathbb{P}\left(\hat{\Pi}_{t+1} \in B \mid \pi_t, s_t^{1:N}, {\gamma_t^*}^{1:N}\right) \\
&= \sum_{a_t^{1:N} \in \mathcal{A}_t^{1:N}} \mathbb{P}\left(\hat{\Pi}_{t+1} \in B \mid \pi_t, s_t^{1:N}, a_t^{1:N}\right) \mathbb{P}\left(a_t^{1:N} \mid s_t^{1:N}, {\gamma_t^*}^{1:N}\right) \\
&= \sum_{a_t^{1:N} \in \mathcal{A}_t^{1:N}} \mu_t(B) \mathbb{P}\left(a_t^{1:N} \mid s_t^{1:N}, {\gamma_t^*}^{1:N}\right),
\end{aligned}$$

and also note that by Kantorovich-Rubinstein duality,

$$\begin{aligned}
&\mathcal{K}\left(\mu_t(B; {\gamma_t^*}^{1:N}), \nu_t(B; {\hat{\gamma}_t^0}^{1:N})\right) \\
&= \sup_{\|f\|_{\text{Lip}} \leq 1} \left|\int f d\mu_t({\gamma_t^*}^{1:N}) - \int f d\nu_t({\hat{\gamma}_t^0}^{1:N})\right| \\
&= \sup_{\|f\|_{\text{Lip}} \leq 1} \Big| \sum_{a_t^{1:N} \in \mathcal{A}_t^{1:N}} \mathbb{P}\left(a_t^{1:N} \mid s_t^{1:N}, {\gamma_t^*}^{1:N}\right) \int f d\mu_t(a_t^{1:N}) \\
&\qquad - \sum_{a_t^{1:N} \in \mathcal{A}_t^{1:N}} \mathbb{P}\left(a_t^{1:N} \mid \hat{s}_t^{1:N}, {\hat{\gamma}_t^0}^{1:N}\right) \int f d\nu_t(a_t^{1:N})\Big| \qquad (7) \\
&\leq \sup_{\|f\|_{\text{Lip}} \leq 1} \sum_{a_t^{1:N} \in \mathcal{A}_t^{1:N}} \mathbb{P}\left(a_t^{1:N} \mid s_t^{1:N}, {\gamma_t^*}^{1:N}\right) \\
&\qquad \times \left|\int f d\mu_t(a_t^{1:N}) - \int f d\nu_t(a_t^{1:N})\right|. \qquad (8)
\end{aligned}$$

Equality (7) holds because the probability measures are finite and the coefficients are non-negative. Inequality (8) follows from the triangle inequality and the fact that $\mathbb{P}\left(a_t^{1:N} \mid \hat{s}_t^{1:N}, {\hat{\gamma}_t^0}^{1:N}\right) = \mathbb{P}\left(a_t^{1:N} \mid s_t^{1:N}, {\gamma_t^*}^{1:N}\right), \forall a_t^{1:N} \in \mathcal{A}_t^{1:N}$. Since the supremum of summation is no larger than

the summation of suprema:

$$\begin{aligned}
&\mathcal{K}\left(\mu_t(B;\gamma_t^{*\,1:N}),\nu_t(B;\hat{\gamma}_t^{0\,1:N})\right)\\
\le&\sum_{a_t^{1:N}\in\mathcal{A}_t^{1:N}}\mathbb{P}\left(a_t^{1:N}\mid s_t^{1:N},\gamma_t^{*\,1:N}\right)\\
&\quad\times\sup_{\|f\|_{\text{Lip}}\le 1}\left|\int f d\mu_t(a_t^{1:N})-\int f d\nu_t(a_t^{1:N})\right|\\
\le&\sum_{a_t^{1:N}\in\mathcal{A}_t^{1:N}}\mathbb{P}\left(a_t^{1:N}\mid s_t^{1:N},\gamma_t^{*\,1:N}\right)\mathcal{K}\left(\mu_t(a_t^{1:N}),\nu_t^{1:N}(a_t^{1:N})\right) \quad (9)\\
\le&\,\delta. \quad (10)
\end{aligned}$$

Inequality (9) holds because Kantorovich-Rubinstein duality implies that $\mathcal{K}(\mu,\nu)$ is the upper bound of any function with Lipschitz constant no larger than 1, and hence holds for the specific function f. Finally, (10) is due to Definition 3(b) and the fact that $\sum_{a_t^{1:N}\in\mathcal{A}_t^{1:N}}\mathbb{P}\left(a_t^{1:N}\mid s_t^{1:N},\gamma_t^{*\,1:N}\right)=1$.

Using this oracle $\hat{\gamma}_t^{0\,1:N}$, for any realization of sufficient private information $s_t^{1:N}$ and common information c_t at time t, let $\pi_t=\eta(c_t)$ and $\hat{\pi}_t=\hat{\eta}(c_t)$; then, we have:

$$\begin{aligned}
&Q_t(\pi_t,\gamma_t^{*\,1:N}) \quad (11)\\
=&\mathbb{E}\Big[R_t(X_t,\gamma_t^{*\,1:N}(s_t^{1:N}))\\
&\qquad+V_{t+1}(\psi_{t+1}^{\gamma_t^*}(\pi_t,Z_{t+1}))\Big|\pi_t,s_t^{1:N},\gamma_t^{*\,1:N}\Big] \quad (12)\\
=&\mathbb{E}\Big[R_t(X_t,\gamma_t^{*\,1:N}(s_t^{1:N}))+V_{t+1}(\Pi_{t+1})\mid\pi_t,s_t^{1:N},\gamma_t^{*\,1:N}\Big] \quad (13)\\
\le&\mathbb{E}\Big[R_t(X_t,\gamma_t^{*\,1:N}(s_t^{1:N}))+\hat{V}_{t+1}(\hat{\Pi}_{t+1})\mid\pi_t,s_t^{1:N},\gamma_t^{*\,1:N}\Big]\\
&\qquad+(T-t)(\epsilon+L_V\delta). \quad (14)
\end{aligned}$$

Equality (12) is by the definition of Q_t in the dynamic programming. Equality (13) is by the definition of ψ. Inequality (14) comes from our induction hypothesis. Using the results from Equations (5) and (6), we then have:

$$\begin{aligned}
&Q_t(\pi_t,\gamma_t^{*\,1:N}) \quad (15)\\
\le&\left(\mathbb{E}\Big[R_t(X_t,\hat{\gamma}_t^{0\,1:N}(\hat{s}_t^{1:N}))\mid\hat{\pi}_t,\hat{s}_t^{1:N},\hat{\gamma}_t^{0\,1:N}\Big]+\epsilon\right)\\
&\qquad+\left(\mathbb{E}\Big[\hat{V}_{t+1}(\hat{\Pi}_{t+1})\mid\hat{\pi}_t,\hat{s}_t^{1:N},\hat{\gamma}_t^{0\,1:N}\Big]+L_V\delta\right)\\
&\qquad+(T-t)(\epsilon+L_V\delta) \quad (16)\\
=&\hat{Q}_t(\hat{\pi}_t,\hat{\gamma}_t^{0\,1:N})+(T-t+1)(\epsilon+L_V\delta) \quad (17)\\
\le&\hat{Q}_t(\hat{\pi}_t,\hat{\gamma}_t^{*\,1:N})+(T-t+1)(\epsilon+L_V\delta). \quad (18)
\end{aligned}$$

Equality (17) follows from the definition of $\hat{Q}_t(\hat{\pi}_t,\hat{\gamma}_t^{0\,1:N})$. Inequality (18) holds because $\hat{\gamma}_t^{*\,1:N}$ is optimal to the embedded dynamic program, and hence its $\hat{Q}_t$ value is no smaller than that of $\hat{\gamma}_t^{0\,1:N}$. □

Our result suggests that the error of carrying out dynamic programming using only the embedded information is upper bounded linearly in time from dynamic programming with the full information. To obtain a small value error upper bound, embedding schemes with small compression errors (ϵ and δ) should be designed.

C. Learning an Information State Embedding

In this subsection, we introduce an empirical instance of the information state embedding based on Recurrent Neural Networks (RNNs), and demonstrate how to learn such an embedding from data. A theoretical upper bound for the (ϵ,δ) values of this embedding is generally unclear, as it requires quantification of the expressiveness of the RNNs, which is beyond the scope of this paper. Instead, the embedding is introduced to demonstrate the feasibility of the embed-then-learn framework. We evaluate its performance empirically in the next section.

The recurrent neural network embedding (RNN-E) uses an RNN to compress the history. In our implementation, each agent uses an LSTM network [30] (a variant of RNN) that maps its local history to a fixed-size vector at each time step. We treat the fixed-size hidden state of the LSTM network as our information state embedding. Recursive update is inherent in the structure of LSTM: $(\hat{s}_{t+1}^i,\hat{\pi}_{t+1})=\text{lstm}_i\left(\hat{s}_t^i,\hat{\pi}_t,y_{t+1}^i,a_t^i,z_{t+1},\text{Cell}_t^i;W_i\right)$, where Cell_t^i is the cell state of the LSTM network that keeps a selective memory of history, and W_i is the network parameters to be learned from data.

Once we have an embedding, we use deep Q-networks (DQNs) [22] to learn a policy. Following the embed-then-learn procedure, as illustrated in Fig. 1, we feed the embedding into a DQN to get the Q-value for each candidate action. In the single agent setting, a similar network structure, named DRQN [10], concatenates LSTM and DQN, and adopts an end-to-end structure where LSTM directly outputs the Q-values. In contrast, we extract an embedding first so that we can theoretically bound the value function given an upper bound on the embedding error.

Similar to the problem faced by other MARL algorithms, the environment is non-stationary from each agent's perspective since agents learn and update policies concurrently. To address this issue, common training schemes in the literature include centralized training and execution, concurrent learning, and parameter sharing [5]. In our simulations, we adopt parameter sharing as it has demonstrated better performance in [5]. In parameter sharing, homogeneous agents share the same network parameter values, which leads to more efficient training and partly addresses the non-stationarity issue in concurrent learning. Heterogeneous policies are still possible because agents feed unique agent IDs and different local observations into the network. However, as a standard training scheme in the literature, parameter sharing does slightly break the assumption of decentralization, as it requires either centralized learning (but still fully decentralized execution), or periodic gradients sharing among agents (which is still a weaker assumption than real-time sharing of local observations).

IV. Numerical Results

In this section, we evaluate our embedding scheme on several benchmark problems in the Dec-POMDP literature [31]: Grid3x3corners [32], Dectiger [33], and Boxpushing [34].

We first introduce two heuristics that can also serve to compress the history, but generally do not satisfy our definition of an embedding (more specifically, they are not injective). These two compression heuristics are followed by a DQN to learn a policy in exactly the same way as RNN-E.

We will use these two heuristics as baselines to evaluate the performance of our embedding.

Finite memory compression. The first heuristic compression, finite memory compression (FMC), simply maintains a fixed memory, or window, of the local history as an approximate information state.[6] Specifically, each agent maintains a one-hot encoded vector of a fixed window of its most recent actions and observations, and its decision only depends on this fixed memory. This compression can be updated recursively via $\hat{s}^i_{t+1} = \left(\hat{s}^i_t \backslash \{y^i_{t-M+1}, a^i_{t-M}, z_{t-M+1}\}\right) \cup \{y^i_{t+1}, a^i_t, z_{t+1}\}$, where M is the length of the fixed window. This compression can be regarded as a simplification of [20], where the authors define each complete history sequence to be an information state and perform Q-learning [36] on such an information state space. Their method does not scale well to longer horizons due to the explosion of the new state space; in contrast, FMC is more scalable as its size is fixed, but it comes at a price of losing long-term memory.

Principal component analysis compression. The second heuristic compression, principal component analysis compression (PCAC), uses PCA [37] to reduce the local history to a fixed-size feature vector. PCA is a simple and well-established algorithm for dimensionality reduction. Given a specified dimensionality, PCA keeps the largest variance during compression. Note that this goal differs from our intention of maintaining the predictive capability for decision-making purposes. PCAC cannot be defined a priori and needs to be first trained on a data set. In our implementation, we generate this training set by uniformly randomizing history sequences of the same length. We also note that PCA can be implemented recursively [38] to handle sequential data.

We compare the performance of our RNN-based embedding with the above two compression heuristics, the state-of-the-art planning solution FB-HSVI [7], which requires a complete model of the environment, and a learning solution oSARSA [8]. We refer to the performances of FB-HSVI and oSARSA, as reported by their authors. The authors limited the running time of their algorithms to 10^5 episodes and 5 hours, but these stopping criteria are not the binding constraints for our solutions, as our algorithm takes significantly less time and fewer episodes to converge.

For RNN-E, we use a one-layer LSTM network with a hidden layer of size 10 as the embedding network. The inputs to the LSTM are one-hot encoded actions and observations, together with the embedding from the last step. Our DQN is a two-layer fully connected network, where the input is the embedding/compression. The DQN hidden layer has 10 neurons, and the output size is equal to the size of action space. All activations are Rectified Linear Unit (ReLU) [39] functions.

We adopt the ϵ-greedy approach for policy exploration, with ϵ decreasing linearly from 0.8 to 0 over the total 40000 episodes by default. We use a buffer of size 4000 for experience replay, and for DQN error estimation we draw a batch of 400 samples from the buffer. We use Mean Squared Error loss and the Adam optimizer, with a learning rate of 10^{-2} by default for both the embedding network and DQN. We perform back-propagation after each episode, and update the target network of DQN every 100 episodes. The size of the compressed information state for FMC and PCAC is set to $M = 4$. For more efficient training, we only train our networks with a horizon length of 10 and then test on different horizons, which surprisingly achieves comparable performance as training and testing on each possible horizon length separately. We average the performance over 2000 testing episodes in each run, and all results are averaged over 10 runs. The performances of the five algorithms over different lengths of horizon T are shown in TABLE I.

	Parameter Sharing			Centralized Solutions	
T	RNN-E	FMC	PCAC	oSARSA	FB-HSVI
Grid3x3corners					
6	0.86	0.83	0.26	1.49	1.49
7	1.41	1.30	0.51	2.19	2.19
8	1.94	1.93	0.72	2.95	2.96
9	2.69	2.53	1.01	3.80	3.80
10	3.47	3.25	1.30	4.69	4.68
Dectiger					
3	4.58	4.89	0.06	5.19	5.19
4	2.97	3.78	1.00	4.80	4.80
5	1.46	2.02	0.65	6.99	7.02
6	2.50	2.95	0.71	2.34	10.38
7	0.85	1.89	0.53	2.25	9.99
Boxpushing					
3	12.63	64.92	17.81	65.27	66.08
4	65.06	76.83	17.76	98.16	98.59
5	81.51	94.22	34.28	107.64	107.72
6	91.00	97.03	34.65	120.26	120.67
7	106.76	143.53	34.23	155.21	156.42

TABLE I: Performance on classic benchmark problems

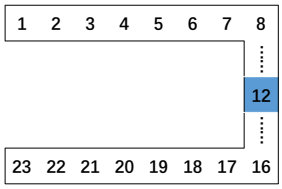

Fig. 2: A Dec-POMDP example requiring long-term memory

We can see that RNN-E and FMC achieve high rewards over different horizons. FMC performs better on shorter horizon problems Dectiger and Boxpushing, but RNN-E outperforms FMC on Grid3x3corners, where horizons are longer and long-term memories are necessary. Although FMC achieves good performance in the three examples, we note that it has a very limited scope of application, because it is easy to construct examples where short-term memories are not sufficient for decision making. For example, consider a two-agent Dec-POMDP problem as illustrated in Fig. 2, which can be regarded as a modification of Grid3x3corners.

[6]Although finite-memory decision making has been well studied in the single agent setting [35], it is still generally open how the truncation of history influences the performance in the multi-agent setting.

Agent 1 starts from state 1 in the maze, and Agent 2 starts from state 23. The goal for the two agents is to meet at the destination state 12 as soon as possible (i.e., they receive a time-discounted unit reward when both of them are in state 12, and no reward otherwise). Candidate actions include moving one step in any of the four directions. They always receive the same observation no matter what states they are in and what actions they take. If an agent runs into a wall, it stays where it is. For each agent, we can see that it is sufficient to only count how many times it has been going right, and the optimal strategy is to switch from going right to going down / up when the count reaches 7. Now suppose Agent 1 only has a finite memory of length 4. Then this agent performs poorly because it cannot distinguish states 5, 6, 7, and 8. If it decides to deterministically go right, it will get stuck in state 8 forever. If it has some probability of going down, then it wastes time in states 5, 6, and 7. Therefore, finite memory agents obtain low rewards in this example. On the other hand, RNN-E is able to summarize the entire history rather than only keeping a short-term memory, but it comes at a price that RNNs are generally difficult to train mostly due to the vanishing and exploding gradient problems.

PCAC does not perform well on the three tasks. We believe the reason is that PCA is designed to keep the largest possible variance of data, which is generally not the same as the most predictive information of the history as required by Definition 3. The oSARSA algorithm performs comparably well as the planning solution FB-HSVI, and generally outperforms our solutions. This is because oSARSA relies on centralized learning, which is a much stronger assumption than the parameter sharing assumption that our solutions rely on. The centralized scheme of oSARSA also incurs heavy computation, as it requires to solve a mixed-integer linear program at each step.

V. Concluding Remarks and Future Directions

In this paper, we have introduced the concept of information state embedding for partially observable cooperative MARL. We have theoretically analyzed how the compression error of the embedding influences the value functions. We have also proposed an instance of embedding based on RNNs, and empirically evaluated its performance on partially observable MARL benchmarks. An interesting future direction would be to theoretically analyze the compression errors of the embedding/compression strategies we have used, which helps close the loop of our theoretical analysis. It would also be interesting to design other empirical embeddings that explicitly reduce this compression error.

References

[1] Y. Duan, B. X. Cui, and X. H. Xu, "A multi-agent reinforcement learning approach to robot soccer," *Artificial Intelligence Review*, vol. 38, no. 3, pp. 193–211, 2012.

[2] S. Wang, J. Wan, D. Zhang, D. Li, and C. Zhang, "Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination," *Computer Networks*, vol. 101, pp. 158–168, 2016.

[3] J. W. Lee, J. Park, O. Jangmin, J. Lee, and E. Hong, "A multiagent approach to Q-learning for daily stock trading," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 37, no. 6, pp. 864–877, 2007.

[4] A. Nayyar, A. Mahajan, and D. Teneketzis, "The common-information approach to decentralized stochastic control," in *Information and Control in Networks*. Springer, 2014, pp. 123–156.

[5] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *AAMAS*, 2017.

[6] A. Nayyar, A. Mahajan, and D. Teneketzis, "Decentralized stochastic control with partial history sharing: A common information approach," *IEEE Trans. Automatic Control*, vol. 58, no. 7, pp. 1644–1658, 2013.

[7] J. S. Dibangoye, C. Amato, O. Buffet, and F. Charpillet, "Optimally solving Dec-POMDPs as continuous-state MDPs," *Journal of Artificial Intelligence Research*, vol. 55, pp. 443–497, 2016.

[8] J. S. Dibangoye and O. Buffet, "Learning to act in decentralized partially observable MDPs," in *Proc. ICML*, 2018, pp. 1233–1242.

[9] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of Markov decision processes," *Math. Operations Research*, vol. 27, no. 4, pp. 819–840, 2002.

[10] M. Hausknecht and P. Stone, "Deep recurrent q-learning for partially observable mdps," *arXiv preprint arXiv:1507.06527*, 2015.

[11] H. S. Witsenhausen, "A standard form for sequential stochastic control," *Mathematical Systems Theory*, vol. 7, no. 1, pp. 5–11, 1973.

[12] H. Tavafoghi, Y. Ouyang, and D. Teneketzis, "A unified approach to dynamic decision problems with asymmetric information-part I: Non-strategic agents," *arXiv preprint arXiv:1812.01130*, 2018.

[13] J. S. Dibangoye, O. Buffet, and F. Charpillet, "Error-bounded approximations for infinite-horizon discounted decentralized POMDPs," in *Proc. ECML/PKDD*, 2014, pp. 338–353.

[14] X. Ma, P. Karkus, D. Hsu, and W. S. Lee, "PF-LSTM: Belief state particle filter for LSTM," in *NeurIPS RLPO Workshop*, 2018.

[15] P. Moreno, J. Humplik, G. Papamakarios, B. A. Pires, L. Buesing, N. Heess, and T. Weber, "Neural belief states for partially observed domains," in *NeurIPS RLPO Workshop*, 2018.

[16] M. L. Littman and R. S. Sutton, "Predictive representations of state," in *Proc. NeurIPS*, 2002, pp. 1555–1561.

[17] A. Zhang, Z. C. Lipton, L. Pineda, K. Azizzadenesheli, A. Anandkumar, L. Itti, J. Pineau, and T. Furlanello, "Learning causal state representations of partially observable environments," *arXiv preprint arXiv:1906.10437*, 2019.

[18] T. Lesort, N. Díaz-Rodríguez, J.-F. Goudou, and D. Filliat, "State representation learning for control: An overview," *Neural Networks*, vol. 108, pp. 379–392, 2018.

[19] J. Subramanian and A. Mahajan, "Approximate information state for partially observed systems," in *Proc. CDC*, 2019, pp. 1629–1636.

[20] B. Banerjee, J. Lyle, L. Kraemer, and R. Yellamraju, "Sample bounded distributed reinforcement learning for decentralized POMDPs," in *Proc. AAAI*, 2012, pp. 1256–1262.

[21] S. Omidshafiei, J. Pazis, C. Amato, J. P. How, and J. Vian, "Deep decentralized multi-task multi-agent reinforcement learning under partial observability," in *Proc. ICML*, 2017, pp. 2681–2690.

[22] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[23] K. Zhang, Z. Yang, and T. Başar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms," *arXiv preprint arXiv:1911.10635*, 2019.

[24] K. Zhang, E. Miehling, and T. Başar, "Online planning for decentralized stochastic control with partial history sharing," in *Prof. ACC*, 2019, pp. 3544–3550.

[25] F. A. Oliehoek, C. Amato *et al.*, *A concise introduction to decentralized POMDPs*. Springer, 2016, vol. 1.

[26] S. P. Singh, T. S. Jaakkola, and M. I. Jordan, "Learning without state-estimation in partially observable Markovian decision processes," in *Proc. ICML*, 1994, pp. 284–292.

[27] H. P. Sankappanavar and S. Burris, "A course in universal algebra," *Graduate Texts Math*, vol. 78, 1981.

[28] D. A. Edwards, "On the Kantorovich–Rubinstein theorem," *Expositiones Mathematicae*, vol. 29, no. 4, pp. 387–398, 2011.

[29] E. J. Sondik, "The optimal control of partially observable Markov processes," *Ph.D. Thesis*, 1971.

[30] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[31] M. Spaan, C. Amato, F. Oliehoek, and S. Witwicki, "Masplan." [Online]. Available: http://masplan.org

[32] C. Amato, J. S. Dibangoye, and S. Zilberstein, "Incremental policy generation for finite-horizon Dec-POMDPs," in *ICAPS*, 2009.
[33] R. Nair, M. Tambe, M. Yokoo, D. Pynadath, and S. Marsella, "Taming decentralized POMDPs: towards efficient policy computation for multiagent settings," in *Proc. IJCAI*, 2003, pp. 705–711.
[34] S. Seuken and S. Zilberstein, "Improved memory-bounded dynamic programming for decentralized POMDPs," in *Proc. UAI*, 2007, pp. 344–351.
[35] C. C. White III and W. T. Scherer, "Finite-memory suboptimal design for partially observed Markov decision processes," *Operations Research*, vol. 42, no. 3, pp. 439–455, 1994.
[36] C. J. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, 1992.
[37] K. Pearson, "On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.
[38] W. Li, H. H. Yue, S. Valle-Cervantes, and S. J. Qin, "Recursive PCA for adaptive process monitoring," *Journal of Process Control*, vol. 10, no. 5, pp. 471–486, 2000.
[39] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. AISTATS*, 2011, pp. 315–323.