Bayesian Learning for literal POMDP in Communicative IPOMDPs

Sarit Adhikari
University of Illinois at Chicago
Chicago, USA
sadhik6@uic.edu

Piotr Gmytrasiewicz
University of Illinois at Chicago
Chicago, USA
piotr@uic.edu

ABSTRACT

We propose a Bayesian learning methodology for addressing the problem of communication for the literal listener, a POMDP agent at the bottom of the cognitive hierarchy in the Communicative Interactive Partially Observable Markov Decision Process (CIPOMDP) framework. The agent models and learns the uncertainty in message distribution by recording the counts of messages received at each state during interaction with the environment and the unknown sender. We first propose parameterization of message distribution and then the update procedure which is integrated alongside the usual POMDP belief update. We provide an approximate update procedure based on particle filtering and show the results on benchmark multi-agent POMDP domains.

KEYWORDS

Bayesian Learning; Multi-Agent Systems; Planning

ACM Reference Format:
Sarit Adhikari and Piotr Gmytrasiewicz. 2023. Bayesian Learning for literal POMDP in Communicative IPOMDPs. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 6 pages.

1 INTRODUCTION

The Communicative IPOMDP (CIPOMDP) framework [7] is a principled approach to modeling interaction and communication between agents in a multi-agent setting. It extends the Interactive POMDP (IPOMDP) framework [6], by allowing agents to exchange messages with each other. Communication is treated as a type of action in CIPOMDPs, and decisions about communication are made using decision-theoretic planning and the Bellman optimality principle. In CIPOMDPs, agents update their beliefs based on their actions, observations, and messages they receive from and send to other agents using Bayes' theorem. This approach allows for the consideration of agents who are not necessarily cooperative during communication and are only guided by their own self-interest.

In the CIPOMDP framework, the interactive state space (IS) includes nested models of the beliefs of different agents. When an agent updates their beliefs based on the Bayesian update, this update process is applied to all levels of the agent's nested theories of mind. These simulations are a feature of game-theoretic pragmatics, which is a way of formalizing how agents communicate and make decisions in multi-agent settings. The recursive process of updating beliefs terminates at the level of "flat" POMDP models, which represent agents who do not have explicit models of other agents. In the CIPOMDP framework, it is assumed that an agent who does not have an explicit model of any other agents can still participate in the exchange of messages. This agent, referred to as a "literal listener," can incorporate the content of a message into its beliefs about the physical state without considering the beliefs or intentions of the sender when interpreting the message. It rather takes the message at face value and updates its beliefs accordingly.

Thus, one of the challenges in the CIPOMDP literature is integrating incoming messages into the belief update process. In the CIPOMDP framework, level-0 POMDP agents do not have the ability to make use of incoming messages, which can limit the usefulness of the CIPOMDP framework. Without the ability to incorporate incoming messages, level-1 CIPOMDP agents may not have the incentive to send particular messages, which would eliminate the purpose of communication and consequently the overall benefits of the CIPOMDP framework. It is important to find ways to effectively incorporate incoming messages into the belief update process in order to fully realize the potential of the CIPOMDP framework for modeling the communication between agents.

Among previous works, [1] deals with the problem by assuming a fixed discretized message distribution under the gullibility assumption and demonstrates the scenarios in cooperative and deceptive cases. [9] offers workaround to the problem by defining a function which maps a message received from higher level agent to its action, replacing the fixed distribution ascribed by the level-0 agent to sender's actions.

We proceed to define the principled approach to learning message distribution by keeping track of the counts of each message in each state. This frequency-based model is a part of the POMDP state space and hence can be updated using the Bayesian approach. We first formalize the notion of "literal POMDP", derive the belief update and provide a modified particle filtering algorithm for approximating the posterior. We finally report the performance of the algorithm in recovering the true message-generating distribution.

We define the listener to be literal in the sense that the listener's interpretation of the received message does not take the intention of the sender into account, owing to the lack of an opponent model. Rather the listener learns the crude model of the frequency of messages in each state. This is in contrast to a strategic listener, which infers the most likely epistemic state that would have triggered the speaker to say what he actually said and not something else [4]. In the context of the theory of mind agent (Figure 1), as we ascend the cognitive hierarchy, the agents with strategy level l>0, can attribute strategic interpretation to the received message due to the sophistication of opponent modeling [7]. The strategic listeners are also explored in pragmatic reasoning literature like [3] which shows that the listeners use Bayesian inference to recover speakers' intended referents in a referential communication game.

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom. © 2023 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

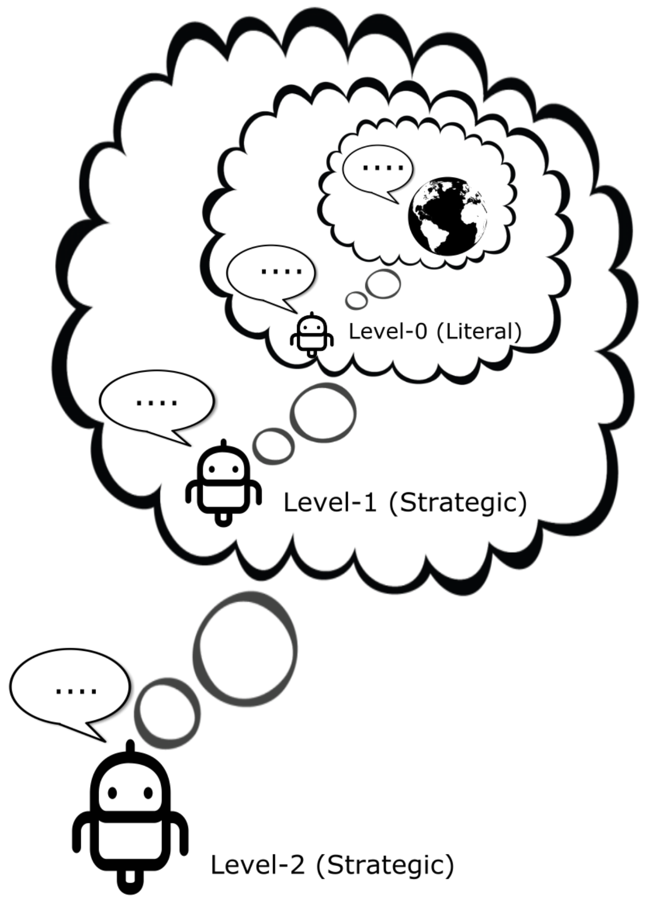

Figure 1: The cognitive hierarchy for the level-2 CIPOMDP agent. For levels $l > 0$ the agent can model the intentional choice of message from the sender. The recursion terminates at level $l = 0$, where the literal agent assumes the message is being generated from fixed distribution from the environment

2 BACKGROUND

2.1 Partially Observable Markov Decision Processes (POMDPs)

POMDPs provide the principled decision-making framework for modeling the agent acting in partially observable and non-deterministic environment. POMDP can be defined as a tuple

$$\langle S, A, T, R, \Omega, O \rangle$$

where,

- S is a state space
- A(s) ⊆ A is a set of actions applicable to set of states s ∈ S
- T is a transition function $T : S \times A \times S' \to \Delta(S)$ gives probability of transitioning to state s' given state s and action a $P(s' \mid s, a)$
- R is a reward function $R : S \times A \to \mathbb{R}$
- Ω is a set of observations
- O is an observation function $O : S \times A \times S' \to \Delta(\Omega)$ gives probability of receiving observation ω while transitioning to state s' after taking action a $P(\omega \mid s, a, s')$

In sequential decision-making, the agent alternates between taking action and receiving observation. Since the POMDP agent cannot observe the state directly, it maintains belief over the possible physical states of the world, which summarizes the action-observation history.

In general, POMDPs do not take part in the exchange of messages and cannot keep track of the presence of other agents in the environment. We propose the literal-POMDP which can keep track of the received messages from the unknown sender and hence learn the message distribution.

2.2 Communicative Interactive POMDPs

CIPOMDP [7] [5] is the bayesian decision-theoretic framework for a self-interested agent to communicate and interact with other agents in the partially observable and stochastic environment. A finitely nested communicative interactive POMDP of agent i in an environment with agent j, is defined as:

$$CIPOMDP_i = \langle IS_{i,l}, A_i, \mathbb{M}, \Omega_i, T_i, O_i, R_i \rangle \quad (1)$$

where $IS_{i,l}$ is a set of interactive states, defined as $IS_{i,l} = S \times M_{j,k}, l \geqslant 1$, where S is the set of physical states and $M_{j,k}$ is the set of possible models of agent j, l is the strategy (nesting) level, and $k < l$.

2.3 Dirichlet Distribution

The Dirichlet distribution is a multivariate distribution over the simplex, a set of non-negative values that sum to 1. It is commonly used as a prior distribution in Bayesian analysis, particularly in the context of categorical data.

The probability density function (PDF) of the Dirichlet distribution with parameters $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_k)$ is given by:

$$f(\boldsymbol{x}; \boldsymbol{\alpha}) = \frac{\Gamma(\sum_{i=1}^{k} \alpha_i)}{\prod_{i=1}^{k} \Gamma(\alpha_i)} \prod_{i=1}^{k} x_i^{\alpha_i - 1}$$

where $\boldsymbol{x} = (x_1, x_2, \ldots, x_k)$ is a k-dimensional vector of values on the simplex, and $\Gamma(\cdot)$ is the gamma function.

The mean of the Dirichlet distribution is
$\boldsymbol{\mu} = (\frac{\alpha_1}{\sum_{i=1}^{k} \alpha_i}, \frac{\alpha_2}{\sum_{i=1}^{k} \alpha_i}, \ldots, \frac{\alpha_k}{\sum_{i=1}^{k} \alpha_i})$

The Dirichlet distribution is a conjugate prior of the categorical distribution, meaning that the posterior distribution is also a Dirichlet distribution. This makes it a convenient choice for the Bayesian analysis of categorical data like discretized message distribution. We show in the next section how the Dirichlet distribution can be used to model uncertainty over the message distribution. We draw inspiration from the works like [11] and [2] where Dirichlet is used to model uncertainty over the parameters of the POMDP.

3 APPROACH

3.1 Parameterization - Message distribution as a categorical distribution

IN CIPOMDP, the messages are probability distributions. We proceed by discretizing the probability ascribed to a state into finite bins. Let's say message space is discretized into 3 regions and augmented with nil, resulting in 4 messages $|M| = 4$. For notational convenience, let's suppose messages are represented as indices 1, 2, 3, and 4.

Table 1: Parameterization of Message Distribution

		message					
		1	2	3	4		
state	+	ϕ_{+1}	ϕ_{+2}	ϕ_{+3}	ϕ_{+4}	ϕ_{+}	ϕ
	-	ϕ_{-1}	ϕ_{-2}	ϕ_{-3}	ϕ_{-4}	ϕ_{-}	

Table 1 shows the state-message matrix Φ parameterizing message distribution. For each state, we need a set of parameters ϕ , which encodes how likely a message is in particular state. For 2-state problem, let ϕ_+ and ϕ_- denote the parameter set for states '+' and '-' respectively.

Then, the likelihood of receiving a particular message given a state s and parameter matrix ϕ is defined as

$$Pr(msg =' 1'|s =' +', \phi) = \phi_{+1}$$

For each $s \in S$,

$$msg|\phi_s = (msg_1, msg_2, ...msg_N) \sim Cat(|M|, \phi_s)$$

$$\sum_{\phi \in \phi_+} \phi = \sum_{\phi \in \phi_-} \phi = 1$$
$$\forall \phi \in \Phi, 0 \leqslant \phi \leqslant 1$$

3.2 Example Update - finite set of values for ϕ

For illustration, let's consider the simple scenario with two states, two observations, and two messages each taking a value in a set $\{`+', `-'\}$. Message generation is such that $Pr(msg = `+' |s = `+') = Pr(msg = `-' |s = `-') = \phi$
, and observation function is such that $Pr(obs = `+' |s = `+') = Pr(obs = `-' |s = `-') = 0.85$

Let $\Phi = \{0, 0.5, 1\}$ for both the states. In general, set Φ can be different for different states. The agent starts with uniform distribution over both state space S and parameter space Φ . $Pr(s) = (0.5, 0.5)$ and $Pr(\phi|s) = (0.33, 0.33, 0.33)$.

The belief update proceeds as follows

$$Pr(s = `+', \phi = 1|obs = `+', msg = `+', b)$$
$$\propto Pr(obs = `+' |s = `+')Pr(msg = `+' |s = `+', \phi = 1)$$
$$Pr(s = `+', \phi = 1)$$
$$\propto 0.85 * 1 * 0.33 * 0.5$$

Table 2: Belief update over s and ϕ

s	ϕ	Calculation	posterior
+	0	0.85 * 0 * 0.33 * 0.5	0
+	0.5	0.85 * 0.5 * 0.33 * 0.5	0.283
+	1	0.85 * 1 * 0.33 * 0.5	0.566
-	0	0.15 * 1 * 0.33 * 0.5	0.1
-	0.5	0.15 * 0.5 * 0.33 * 0.5	0.05
-	1	0.15 * 0 * 0.33 * 0.5	0

Table 2 shows posterior over s and ϕ after receiving observation $`+'$ and message $`+'$. Posterior marginal over physical states is (0.85, 0.15) and over Φ is (0.1, 0.33, 0.567). i.e. higher probability to ascribed to $\phi = 1$, as the message and observation agreed with each other.

3.3 Updating Distribution over continuous ϕ_s

In this section, we relax the assumption that ϕ can take a finite set of values. Particularly, if the distribution over ϕ_s is represented by a Dirichlet distribution, then it serves as a conjugate prior to the message distribution parameterized as in the section 3.1. We need a separate Dirichlet for modeling message distribution in each state. Let's consider state '+',

The concentration hyperparameter for Dirichlet distribution is denoted as

$$\alpha_+ = (\alpha_{+,1}, \alpha_{+,2}, \alpha_{+,3}, \alpha_{+,4})$$

The distribution over ϕ_+ given the concentration hyperparameters is denoted as

$$\phi_+|\alpha_+ = (\phi_{+1}, \phi_{+2}, \phi_{+3}, \phi_{+4}) \sim Dir(|M|, \alpha)$$

$c = (c_1, c_2, c_3....c_{|M|})$ is a vector recording the number of occurrences of message m_i in a stream of messages denoted by $msgV$

$$c_i = \sum_{j=1}^{N} [msgV[j] = m_i]$$

Updated distribution over ϕ_+ is given by

$$\phi_+|msgV, \alpha_+ \sim Dir(|M|, c_1 + \alpha_{+,1}, ..., c_k + \alpha_{+,k})$$

3.4 Literal POMDP

Definition 3.1. We define literal POMDP as a tuple

$$\langle \bar{S}, A, T, R, \Omega, O \rangle$$

where A, T, R , Ω, and O are as defined in POMDP (Section 2.1). Literal POMDP augments the state space S with the count vector space C. Each element of $c \in C$ corresponds to receiving the message $m_{i,r}$ while on state s. Thus $\bar{s} = \langle s, c \rangle$

3.4.1 Belief Update for literal POMDP.

Proposition 1: The belief update for literal POMDP is defined as

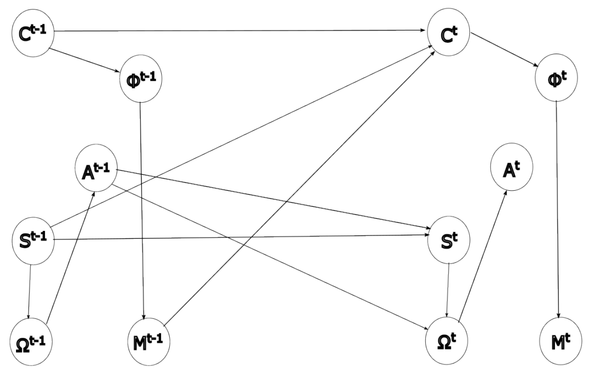

Figure 2: A dynamic bayesian network showing two time slices for literal POMDP

$$
\begin{aligned}
&Pr(s^t, c^t|o^t, m^t_r, a^{t-1}, b^{t-1}) \\
=&\eta' Pr(o^t|s^t, a^{t-1}) \sum_{s^{t-1}} \sum_{c^{t-1}} \boldsymbol{I}_{c'}(\boldsymbol{v}(s^{t-1}, c^{t-1}, m^t_r)) \\
&Pr(s^t|s^{t-1}, a^{t-1}) Pr(m^t_r|s_{t-1}, c^{t-1}, a^{t-1}) b^{t-1}(s^{t-1}, c^{t-1}) \quad (2)
\end{aligned}
$$

where η' is the normalizing constant.

$$\eta' = \frac{1}{Pr(o^t|m^t_r, a_{t-1}, b^{t-1}) Pr(m^t_r|a^{t-1}, b^{t-1})}$$

Analogous to POMDP belief update, which results in updated distribution over the physical states s^t after each action a^{t-1} and observation o^t pair, the belief update for literal POMDP [1] results in updated distribution over the physical states and count vectors $\langle s^t, c^t \rangle$, after action a^{t-1}, observation o^t and a message m^t_r triplet. This is analogous to how a BA-POMDP models and updates the uncertainty over transition and observation functions [10]. The bayesian network in figure 2, represents the belief update and encodes the conditional independence assumptions made during the derivation.

c is a 2 dimensional vector of size $|M| \times |S|$ recording counts of received messages in each state. $\boldsymbol{I}_{c^t}(\hat{c^t})$ is a indicator function returning 1 if $c^t = \hat{c^t}$ and 0 otherwise. $\boldsymbol{v}(s^{t-1}, c^{t-1}, m^t_r)$ is a update function which returns new count $\hat{c^t}$ given previous count c^{t-1}, state s^{t-1} and message m^t_r.

Table 3 shows the analogous terms in belief update between literal POMDP and non-literal CIPOMDP

Proof of proposition 1:

Table 3: Comparison of terms between literal POMDP and CIPOMDP

Literal POMDP	CIPOMDP
$O_i(s^t, a^{t-1}_i, o^t_i)$	$O_i(s^t, a^{t-1}, o^t_i)$
$b^{t-1}_i(s^{t-1}, c^{t-1})$	$b^{t-1}_i(is^{t-1})$
$Pr(m^t_{i,r}\|s^{t-1}, c^{t-1}, a^{t-1}_i)$	$Pr(m^t_{i,r}, a^{t-1}_j\|\theta^{t-1}_j)$
$T_i(s^{t-1}, a^{t-1}_i, s^t)$	$T_i(s^{t-1}, a^{t-1}, s^t)$
$\boldsymbol{I}_{c^t}(\boldsymbol{v}(s^{t-1}, c^{t-1}, m^t_r))$	$\tau_{\theta^t_j}(b^{t-1}_j, a^{t-1}_j, \mathrm{m}^{t-1}_{i,s}, o^t_j, m^t_{i,r}, b^t_j)$

$$
\begin{aligned}
&Pr(s^t, c^t|o^t, m^t_r, a^{t-1}, b^{t-1}) \\
=&\frac{Pr(o^t|s^t, c^t, m^t_r, a^{t-1}, b^{t-1}) Pr(s^t, c^t|m^t_r, a^{t-1}, b^{t-1})}{Pr(o^t|m^t_r, a^{t-1}, b^{t-1})} \\
=&\frac{Pr(o^t|s^t, a^{t-1} \sum_{s^{t-1}} \sum_{c^{t-1}} Pr(s^t, s^{t-1}, c^{t-1}, c^t|m^t_r, a_{t-1}, b_{t-1})}{Pr(o^t|m^t_r, a^{t-1}, b^{t-1})} \\
=&\eta Pr(o^t|s^t, a^{t-1}) \sum_{s^{t-1}} \sum_{c^{t-1}} Pr(c^t|s^t, s^{t-1}, c^{t-1}, m^t_r, a^{t-1}, b^{t-1}) \\
&Pr(s^t, s^{t-1}, c^{t-1}|m^t_r, a^{t-1}, b^{t-1}) \\
=&\eta Pr(o^t|s^t, a^{t-1}) \sum_{s^{t-1}} \sum_{c^{t-1}} Pr(c^t|s^{t-1}, c^{t-1}, m^t_r) \\
&Pr(s^t|s^{t-1}, c^{t-1}, m^t_r, a^{t-1}, b^{t-1}) Pr(s^{t-1}, c^{t-1}|m^t_r, a^{t-1}, b^{t-1}) \\
=&\eta' Pr(o^t|s^t, a^{t-1}) \sum_{s^{t-1}} \sum_{c^{t-1}} Pr(c^t|s^{t-1}, c^{t-1}, m^t_r) \\
&Pr(s^t|s^{t-1}, a^{t-1}) Pr(m^t_r|s^{t-1}, c^{t-1}, a^{t-1}, b^{t-1}) \\
&Pr(s^{t-1}, c^{t-1}|a^{t-1}, b^{t-1}) \\
=&\eta' Pr(o^t|s^t, a^{t-1}) \sum_{s^{t-1}} \sum_{c^{t-1}} \boldsymbol{I}_{c^t}(\boldsymbol{v}(s^{t-1}, c^{t-1}, m^t_r)) \\
&Pr(s^t|s^{t-1}, a^{t-1}) Pr(m^t_r|s^{t-1}, c^{t-1}, a^{t-1}) b^{t-1}(s^{t-1}, c^{t-1})
\end{aligned}
$$

3.5 Algorithm

Since the size of the space of count vectors C is enormous, we cannot explicitly enumerate over each $c \in C$ as we do for $s \in S$. In fact, only a small subset of C is possible starting from initial count c_0 and for each message received m^t_r in subsequent time steps. Thus we can construct $\hat{s^t} = \langle s^t, c^t \rangle$ dynamically as we proceed with the belief update. Algorithm 1 shows the exact belief update procedure for literal POMDP. Line 4 copies the count vector from the previous time step and line 5 constructs the new count vector using the received message and previous state. This updated count vector will be part of the state space in the next time-step. Line 7 creates the new state dynamically and assigns the belief defined by equation 2.

Algorithm 2 provides the particle filtering approach [8] to approximate the belief update for literal POMDP. Here the beliefs are represented as a set of particles and each particle represents a pair of state and a count vector. The particles for the next time steps are generated by first sampling from the transition dynamic and then assigning importance weight to the particles. Finally, particles are resampled based on the importance weights. This extension differs from the typical particle filtering in the sense that weights

[1]subscript i denoting the agent is omitted

are assigned not only according to observation likelihood but also according to message likelihood as shown in lines 6-8.

Algorithm 1 Exact Belief Update for literal POMDP

```
Input: $S, A, T, O, o^t, m^t, a^{t-1}, b^{t-1}$
Output: $b^t$
 1: $\hat{b^t} \leftarrow \{\}$
 2: for $\hat{s}^{t-1}, prob \leftarrow b^{t-1}$ do
 3:     $s^{t-1}, c^{t-1} = \hat{s}^{t-1}$
 4:     $c^t \leftarrow copy(c^{t-1})$
 5:     $c^t \leftarrow c^t + \delta^{m^t}_{s^{t-1}}$
 6:     for $s^t \leftarrow S$ do
 7:         $\hat{b^t}((s^t, c^t)) \overset{+}{\leftarrow} O(s^t, a^{t-1}, o^t)$
 8:         $b^{t-1}((s^{t-1}, c^{t-1}))T(s^{t-1}, a^{t-1}, s^t)$
 9:         $((c^{t-1}(s^{t-1}, m_t) + 1)/(\sum_m c^{t-1}(s^{t-1}, m)$
10:         $+|M|))$
11:     end for
12: end for
13: $b^t \leftarrow Normalize(\hat{b^t})$
```

Algorithm 2 Approximate PF Belief Update for literal POMDP

```
Input: $S, A, T, O, o^t, m^t, a^{t-1}, b^{t-1}$
Output: $b^t$
 1: $\hat{b^t_{temp}} \leftarrow \{\}$
 2: $\hat{W^t} \leftarrow \{\}$
 3: for $\hat{s}^{t-1} \leftarrow b^{t-1}$ do
 4:     $s^{t-1}, c^{t-1} = \hat{s}^{t-1}$
 5:     $s^t \sim T(s^t|s^{t-1}, a^{t-1})$
 6:     $w^t_m \leftarrow ((c^{t-1}(s^{t-1}, m_t) + 1)/(\sum_m c^{t-1}(s^{t-1}, m) + |M|))$
 7:     $w^t_o \leftarrow O(s^t, a^{t-1}, o^t)$
 8:     $W^t \overset{\cup}{\leftarrow} (w^t_o \cdot w^t_m)$
 9:     $c^t \leftarrow copy(c^{t-1})$
10:     $c^t \leftarrow c^t + \delta^{m^t}_{s^{t-1}}$
11:     $\hat{s^t} \leftarrow (s^t, c^t)$
12:     $\hat{b^t_{temp}} \overset{\cup}{\leftarrow} (\hat{s^t})$
13:     $W^t \leftarrow Normalize(W^t)$
14:     $b^t \leftarrow Resample(\hat{b^t_{temp}}, W^t)$
15: end for
```

4 EXPERIMENTS

4.1 Multi-Agent Tiger Game

In this benchmark, the agent is situated in front of two doors labeled "left" and "right." The state of the system is characterized by the presence of a tiger behind one of the doors and a pot of gold behind the other, but the agent lacks knowledge of the exact location. The state space is defined as S=TL, TR, where TL represents the presence of the tiger behind the left door and TR represents the presence of the tiger behind the right door. The agent has three possible actions: opening the right door (OR), opening the left door (OL), or listening (L).

The transition function T describes the change in state after an action is taken. If an agent opens a door, the state resets to either TR or TL with equal probability. However, if the agent listens, the state changes according to transition noise. The observation function O is used to determine the accuracy of the agent's observation after each action. The accuracy of observations is only guaranteed if the agent listens, as the growl of the tiger can be heard coming from either the left (GL) or the right (GR) door with a predefined sensor accuracy. On the other hand, if the agent opens the doors, the growls have an equal chance of coming from either the left or right door, rendering the observations completely uninformative.

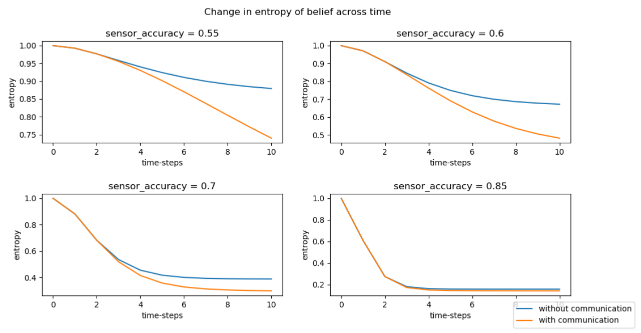

Figure 3: Comparison of entropy reduction for belief distribution in communication vs no communication scenarios. Here the sensor accuracy is varied while transition noise is held constant

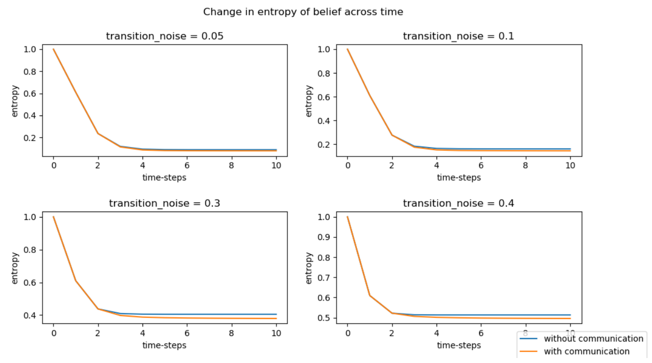

Figure 4: Comparison of entropy reduction for belief distribution in communication vs no communication scenarios. Here the sensor accuracy is held constant while the transition noise is varied

5 RESULTS

In this set of experiments, we fix the policy of the agent to take listen action in each time step and assume it receives the same message in each time step. We compare the reduction in entropy of the belief distribution, between literal POMDP (with the communication), and POMDP (without communication). The results are reported for different settings of sensor accuracy (Figure 3), and transition noise

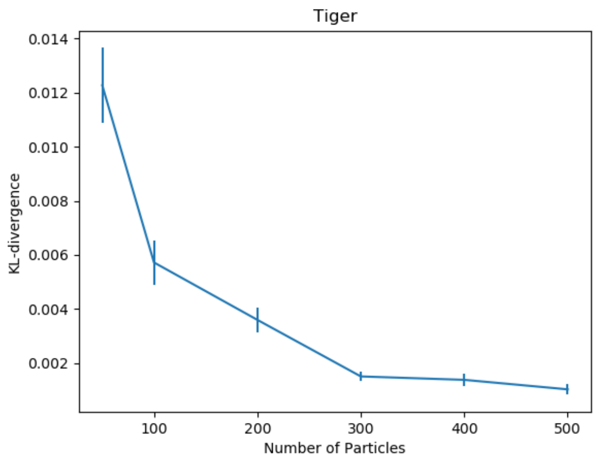

Figure 5: Performance of particle filtering algorithm increases as we increase the number of particles

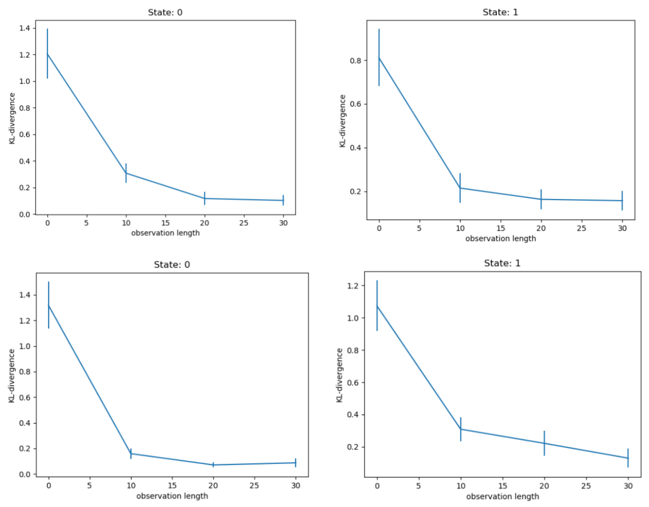

Figure 6: The plot showing the difference between actual message distribution and the one learned by the literal POMDP agent. After sufficient observations, the learned distribution closely approximates the ground truth. The first row corresponds to the exact belief update while the second row corresponds to the approximate belief update with particle filtering

(Figure 4). In each case, the literal POMDP does better than POMDP in reducing the entropy of the belief distribution. The approach can be beneficial for the cooperative scenarios but might make the agent susceptible to bait and switch strategy from the interacting higher level agent. The study of the latter is left for the future work.

Next, in figure 5, we show the performance of the particle filtering approach. The difference with the exact belief update is made in terms of KL divergence. As we increase the number of particles, we get closer to the posterior obtained from the exact belief update. The results are averaged across 100 runs.

Finally, we investigate whether the agent is able to learn the true message distribution after sufficient interaction with the environment. In each run, the message distribution is randomly generated for all states. The agent interacts with the environment for 30 time-steps. After sufficient interaction, the literal POMDP is able to approximate the true message distribution. The results are reported for 10 episodes. Figure 6 shows the comparison for the exact algorithm vs the particle filter algorithm in two states of the tiger game.

6 CONCLUSION AND FUTURE WORK

We defined the principled approach to learning message distribution for literal POMDP, lying at the bottom of the cognitive hierarchy in CIPOMDPs. We empirically showed that the agent can approximate the true message distribution after sufficient observations. The particle filtering-based update method can be integrated to monte-carlo tree-based solution approaches for CIPOMDPs. This allows us to study how the agents higher in the cognitive hierarchy would behave if they model the opponent as a literal POMDP, as opposed to the strategic agent.

REFERENCES

[1] Sarit Adhikari and Piotr Gmytrasiewicz. 2021. Point Based Solution Method for Communicative IPOMDPs. In *Multi-Agent Systems*, Ariel Rosenfeld and Nimrod Talmon (Eds.). Springer International Publishing, Cham, 245–263.

[2] Finale Dosh-Velez. 2009. The infinite partially observable Markov decision process. In *NIPS*.

[3] Michael C. Frank and Noah D. Goodman. 2012. Predicting pragmatic reasoning in language games. *Science* 336 (2012), 998–998.

[4] Michael Franke and Gerhardt Jager. 2016. Probabilistic pragmatics, or why Bayes' rule is probably important for pragmatics. *Zeitschrift fur Sprachwissenschaft* 35 (2016), 3–44.

[5] Piotr Gmytrasiewicz and Sarit Adhikari. 2019. Optimal Sequential Planning for Communicative Actions: A Bayesian Approach. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems* (Montreal QC, Canada) *(AAMAS '19)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1985–1987.

[6] Piotr Gmytrasiewicz and Prashant Doshi. 2005. A Framework for Sequential Planning in Multiagent Settings. *Journal of Artificial Intelligence Research* 24 (2005), 49–79. http://jair.org/contents/v24.html.

[7] Piotr J. Gmytrasiewicz. 2020. How to Do Things with Words: A Bayesian Approach. *J. Artif. Intell. Res.* 68 (2020), 753–776. https://doi.org/10.1613/jair.1.11951

[8] Neil J Gordon, David J Salmond, and Adrian FM Smith. 1993. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE proceedings F (radar and signal processing)*, Vol. 140. IET, 107–113.

[9] Anirudh Kakarlapudi, Gayathri Anil, Adam Eck, Prashant Doshi, and Leen-Kiat Soh. 2022. Decision-theoretic planning with communication in open multiagent systems. In *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence (Proceedings of Machine Learning Research, Vol. 180)*, James Cussens and Kun Zhang (Eds.). PMLR, 938–948. https://proceedings.mlr.press/v180/kakarlapudi22a.html

[10] Sammie Katt, Frans A. Oliehoek, and Christopher Amato. 2017. Learning in POMDPs with Monte Carlo Tree Search. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 1819–1827. https://proceedings.mlr.press/v70/katt17a.html

[11] Stephane Ross, Brahim Chaib-draa, and Joelle Pineau. 2007. Bayes-Adaptive POMDPs. In *Proceedings of the 20th International Conference on Neural Information Processing Systems* (Vancouver, British Columbia, Canada) *(NIPS'07)*. Curran Associates Inc., Red Hook, NY, USA, 1225–1232.