Learning Internal State Models in Partially Observable Environments

Andrea Baisero
College of Computer and Information Science
Northeastern University
Boston, MA 02115
baisero.a@husky.neu.edu

Christopher Amato
College of Computer and Information Science
Northeastern University
Boston, MA 02115
c.amato@northeastern.edu

One of the key unsolved challenges of reinforcement learning (RL) in partially observable Markov decision processes (POMDPs) concerns the task of learning how to summarize histories of events into an *internal state* (i-state) representation which is compact, cumulative, recursively computable, and ultimately useful for optimal decision making. Recurrent models such as RNNs and LSTMs are a natural parameterizations for the i-state dynamics (i-dynamics) [1], however a key aspect remains not completely answered: how do we train the i-state model, and which loss do we use?

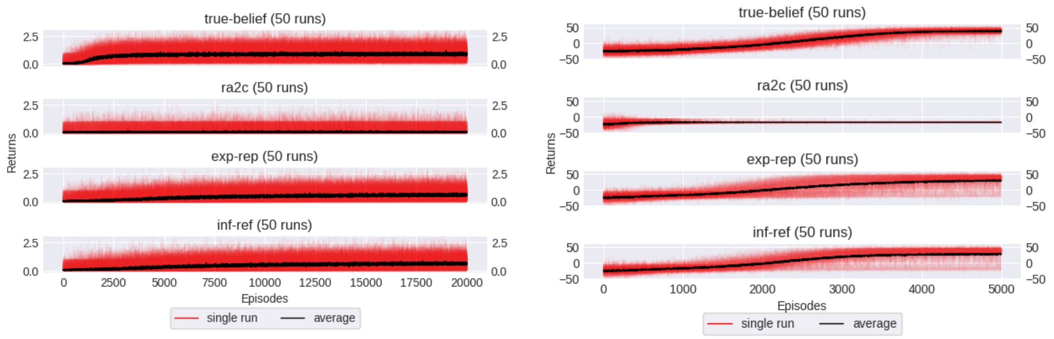
In our approach, the i-state representation is trained not on the performance of the agent, but on its ability to predict the next observations and rewards. We propose two online learning methods; one based on the explicit memorization of past experiences, and another powered by accumulated statistics and exact inference. I-states are fully observable proxies for the unobservable true belief-state, hence a *good* i-state representation can be seen as a tool to reduce POMDP learning tasks into completely observable Markov decision process (MDP) learning tasks. We exploit this by focusing exclusively on learning predictive i-state representations, while using advantage actor critic (A2C) [2] to learn the complementary policy which completes the acting agent. Preliminary empirical results show that our methods either match or outperform learning the i-state representation and policy jointly using a recurrent equivalent [1] of A2C (RA2C).

Related Work The RL literature is rich in approaches for learning meaningful i-state representations in POMDPs. Finite state controllers (FSCs) [3–5] and regionalized policy parameterizations (RPRs) [6] learn discrete representations in a model-free fashion by maximizing the agent's overall performance. Similarly, recurrent policy gradient (RPG) [1] applies policy gradient methods to jointly learn a policy and a RNN/LSTM i-dynamics. Predictive state representations (PSRs) [7–10] are compact state representations based on predictive criteria, and serve as an alternative to standard belief-state [11] representations. Deep variational RL (DVRL) [12] is a model-based approach which learns model dynamics and uses them to compute corresponding belief-states. In contrast to DVRL, our approaches learn an i-state representation which is used as a proxy for belief-MDP dynamics [11], rather than trying to learn the POMDP latent dynamics.

Predictive Internal-State Models We complement the i-state representation with predictive models of observations (o-model) and rewards (r-model). Ideally, all the relevant parameters θ (initial i-state, i-dynamics, o-model, and r-model) would be trained to match, for all observable histories h and actions a, the environment's true posterior predictive distributions of observations $\Pr(o \mid h, a)$ and rewards $\Pr(r \mid h, a)$. Being that these target distributions are not directly available in the general learning scenario, we propose two alternative training methods to achieve the same predictive goal.

Our first approach (*exp-rep*) is based on storing past experiences into an *experience replay* buffer. Given a batch of sample experiences, we train observation and reward predictions to minimize cross-entropy and mean-squared-error losses respectively, in a supervised fashion. To facilitate an efficient learning procedure, we use 2 priority queues (size 512 each) to maintain a population of histories with the highest observation and reward losses encountered so far. In this approach, the true target predictive distributions are approximated by the memorized (sparse) sample instances.

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.



(a) Hallway POMDP (20k episodes shown)

(b) Shopping POMDP (5k episodes shown)

Figure 1: For each method, we perform 50 independent runs of 20k episodes and 50 time-steps each. The graphics show each individual run's returns (in red) and the average run's return (in black).

Our second approach (*inf-ref*) is based on accumulating statistics—which we call the *inferential-reference* model ρ—from past experiences. The reference model ρ defines, for each individual history h and action a, separate statistical models $\rho_{h,a}$ (Bayesian or frequentist) which describe generative processes of the resulting observations o and rewards r. As experience is gathered in the form of history-action-observation-reward tuples by interacting with the environment, the respective reference statistics are updated via exact Bayesian or frequentist inference, and then the parameters θ are optimized to minimize a loss $\mathcal{L}(\theta; \rho)$ dependent on the updated reference ρ (see appendix A for more details). Ideally, parameters θ would be trained using the statistics associated with all encountered histories and actions; in practice, we use 2 priority queues (size 512 each) to focus the training efforts on the histories associated with the highest measured losses. In this approach, the true target predictive distributions are approximated by the accumulated inferential statistics.

Evaluation We evaluate our methods on 2 discrete partially observable environments: the *hallway* domain [13], and a novel *shopping* domain (appendix B). In each environment, we compare the performance of 4 methods, each of which uses the same A2C variant to train the policy model: a) *true-belief* uses the true (generally unavailable) belief-state—we treat this as a soft upper-bound showcasing the kind of performance which can be expected when good i-dynamics have been learned; b) *ra2c* uses recurrent A2C to train both i-dynamics and policy, using the agent's performance as objective; c) *exp-rep* trains the i-dynamics using the experience replay method outlined above; and d) *inf-ref* trains the i-dynamics using the inferential reference method outlined above. All i-dynamics are modeled as 2-layer LSTMs with *tanh* activations and as many hidden units as true environment states, while all non-recurrent models (policy, critic, o-model, r-model) as single-hidden-layer feedforward neural networks with *leaky-ReLU* activations.

Our empirical results (fig. 1) show that, on average, both proposed methods outperform the RA2C strategy, and are able to learn i-state representations as useful as the true belief-state. In the hallway domain (fig. 1a), while RA2C is wholly unable to improve upon the initial policy, the proposed methods slowly converge to achieve a performance only marginally worse than that of the soft upper-bound. In the shopping domain (fig. 1b), while RA2C consistently converges to uninteresting local optima, the proposed methods are able (about 88% of the time) to achieve the same performance as true-belief. Most runs follow one of two distinct trends, either completely succeeding or completely failing to learn useful i-dynamics, with few intermediate results. This suggests the presence of some stochastic latent condition which largely influences whether a useful i-state representation will be learned. We believe this condition to be related to a combination of unfavorable random initialization and insufficient exploration, causing the policy model to converge before the i-dynamics.

Conclusions Learning useful state representations is a fundamental necessity for agents operating under partial observability. We have proposed two methods to train i-dynamics based on their predictive abilities, rather than the performance of the resulting agent. Both methods decouple the task of learning the i-state dynamics from that of learning a policy, and preliminary results indicate the viability of this approach. Future work will focus on facilitating exploration, and scaling the methods to handle larger domains and tasks which require more long-term memory.

References

[1] Daan Wierstra, Alexander Förster, Jan Peters, and Jürgen Schmidhuber. Recurrent policy gradients. *Logic Journal of the IGPL*, 18(5):620–634, 2010.

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

[3] Jonathan Baxter and Peter L. Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research (JAIR-01)*, 15:319–350, 2001.

[4] Jonathan Baxter, Peter L. Bartlett, and Lex Weaver. Experiments with infinite-horizon, policy-gradient estimation. *Journal of Artificial Intelligence Research (JAIR-01)*, 15:351–381, 2001.

[5] Douglas Aberdeen and Jonathan Baxter. Scaling internal-state policy-gradient methods for POMDPs. In *Proceedings of the 19th International Conference on Machine Learning (ICML-02)*, pages 3–10, 2002.

[6] Hui Li, Xuejun Liao, and Lawrence Carin. Multi-task reinforcement learning in partially observable stochastic environments. *Journal of Machine Learning Research (JMLR-09)*, 10 (May):1131–1186, 2009.

[7] Michael L. Littman and Richard S. Sutton. Predictive representations of state. In *Advances in Neural Information Processing Systems (NIPS-02)*, pages 1555–1561, 2002.

[8] Satinder P. Singh, Michael L. Littman, Nicholas K. Jong, David Pardoe, and Peter Stone. Learning predictive state representations. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 712–719, 2003.

[9] Britton Wolfe, Michael R. James, and Satinder Singh. Learning predictive state representations in dynamical systems without reset. In *Proceedings of the 22nd International Conference on Machine Learning (ICML-05)*, pages 980–987. ACM, 2005.

[10] Michael Bowling, Peter McCracken, Michael James, James Neufeld, and Dana Wilkinson. Learning predictive state representations using non-blind policies. In *Proceedings of the 23rd International Conference on Machine Learning (ICML-06)*, pages 129–136. ACM, 2006.

[11] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1):99–134, 1998.

[12] Maximilian Igl, Luisa Zintgraf, Tuan Anh Le, Frank Wood, and Shimon Whiteson. Deep Variational Reinforcement Learning for POMDPs. *arXiv preprint arXiv:1806.02426*, 2018.

[13] Michael L. Littman, Anthony R. Cassandra, and Leslie Pack Kaelbling. Learning policies for partially observable environments: Scaling up. In *Machine Learning Proceedings 1995*, pages 362–370. Elsevier, 1995.

A Inferential Reference Models

We denote the space of probability (mass or density) distributions over a set $\mathcal{X}$ as $\Delta(\mathcal{X})$.

State, action and observation spaces are respectively denoted by $\mathcal{S}$, $\mathcal{A}$, and $\mathcal{O}$, and the space of observable histories is defined as $\mathcal{H} \doteq (\mathcal{A} \times \mathcal{O})^*$. A deterministic i-state representation $\langle \mathcal{N}, n_0, \phi \rangle$ is composed of an i-state space $\mathcal{N}$, an initial i-state $n_0 \in \mathcal{N}$, and i-dynamics $\phi \colon \mathcal{N} \times \mathcal{A} \times \mathcal{O} \to \mathcal{N}$. A *policy* $\pi \colon \mathcal{N} \to \Delta(\mathcal{A})$ selects actions stochastically from given i-states, and complements the i-state representation to complete the acting agent. The o-model $m_o \colon \mathcal{N} \times \mathcal{A} \to \Delta(\mathcal{O})$ returns a distribution over the observation space, and the r-model $m_r \colon \mathcal{N} \times \mathcal{A} \to \mathbb{R}$ estimates the expected reward as a single scalar. The overall reference model $\rho = \{\rho_{h,a}\}_{h,a}$ is composed of an independent reference for each history and action, $\rho_{h,a}$, each modeling the next observation and reward.

Observation Reference Model and Loss The Dirichlet-categorical conjugate pair represents a natural choice to model discrete observations:

$$\omega_{h,a} \sim \text{Dirichlet}(\{\alpha_{h,a,o'}\}_{o'})\,, \tag{1}$$

$$o_{h,a} \sim \text{Categorical}(\omega_{h,a})\,. \tag{2}$$

We initialize the hyper-parameters so as to embed no prior knowledge, i.e. $\alpha_{h,a,o} = 1$. To perform inference, we exploit conjugacy and update the model hyper-parameters by incrementing the counts associated to each encountered history-action-observation tuple $\langle h, a, o \rangle$,

$$\omega_{h,a} \mid o \sim \text{Dirichlet}(\{\tilde{\alpha}_{h,a,o'}\}_{o'})\,, \tag{3}$$

$$\tilde{\alpha}_{h,a,o'} \mid o = \alpha_{h,a,o'} + \mathbb{I}\left[o = o'\right]\,. \tag{4}$$

Because the o-model m_o outputs distributions $x \in \Delta(\mathcal{O})$ over the discrete observation space, we can score the predictiveness of the learned model by means of the neg-log-likelihood of x with respect to the reference Dirichlet distribution,

$$\begin{aligned}\mathcal{L}_o(\theta; \rho, \langle h, a \rangle) &= -\log \text{Dirichlet}(x; \{\alpha_{h,a,o'}\}_{o'})\big|_{x=m_o(\phi(n_0,h),a)} \\ &= -\log\left(\frac{1}{B(\{\alpha_{h,a,o'}\}_{o'})}\prod_{o'} x_{o'}^{\alpha_{h,a,o'}-1}\right) \\ &= \log B(\{\alpha_{h,a,o'}\}_{o'}) + \sum_{o'}\left(\alpha_{h,a,o'}-1\right)\left(-\log x_{o'}\right) \end{aligned} \tag{5}$$

$$\nabla_\theta \mathcal{L}_o(\theta; \rho, \langle h, a \rangle) = \sum_{o'}\left(\alpha_{h,a,o'}-1\right)\nabla_\theta\left(-\log x_{o'}\right) \tag{6}$$

Note that this loss has the desired effect whereby histories which have been observed more often (and are consequently associated a lower overall posterior uncertainty) influence the learning procedure more heavily in light of the higher accumulated counts $\{\alpha_{h,a,o'}\}_{o'}$.

Reward Reference Model and Loss Defining a useful hierarchical model over the continuous reward space requires more domain-dependent assumptions to be made, e.g. concerning the parameterization of a distribution in the continuous space of reals. To make our method domain-agnostic, we employ a simpler frequentist approach. For each history-action pair $\langle h, a \rangle$, we only keep track of the empirical average of the received rewards. Again, we initialize the model statistics so as to contain no prior knowledge, i.e. $\mu_{h,a} = 0$ and $\nu_{h,a} = 0$. Upon observing a new reward, the parameters are updated to represent the new cumulative average of rewards,

$$\tilde{\mu}_{h,a} \mid r = \frac{\mu_{h,a}\nu_{h,a} + r}{\nu_{h,a} + 1}\,, \tag{7}$$

$$\tilde{\nu}_{h,a} \mid r = \nu_{h,a} + 1\,. \tag{8}$$

Finally, the predictiveness of the r-model m_r is scored as the squared difference between its output $x \in \mathbb{R}$ and the reference empirical average,

$$\mathcal{L}_r(\theta; \rho, \langle h, a \rangle) = \left(x - \mu_{h,a}\right)^2\Big|_{x=m_r(\phi(n_0,h),a)}\,. \tag{9}$$

B The Shopping Domain

We define a novel discrete partially observable domain, called the *shopping* domain, parameterized by the width and height of a grid $w > 1, h > 1$; in our evaluation, we use $w = h = 2$. The agent is sent into a $w \times h$ grid-world store to purchase an item, but has forgotten which item was requested.

The domain state encodes both the agent and item locations in the grid, hence $|\mathcal{S}| = (wh)^2$. The action-space $\mathcal{A} = \{\texttt{up}, \texttt{down}, \texttt{left}, \texttt{right}, \texttt{query}, \texttt{buy}\}$, $|\mathcal{A}| = 6$, includes the standard grid-world movements, a `query` action to ask for the location of the item, and a `buy` action to purchase the item in the current cell. Observations are the individual locations in the grid, $|\mathcal{O}| = wh$.

The dynamics of the environment are as follows: The agent enters the grid-world always in the same cell, whereas the item's location is randomly sampled. All movements are deterministic, and querying leaves the agent stationary. Buying the wrong item also leaves the agent stationary, while buying the correct item resets the environment. The agent receives deterministic observations of its location for all actions except `query`; an observation of the item's position is received instead. Movements receive a reward of -1, querying -2, buying the wrong item -5 and buying the correct item 10.

The optimal policy for the shopping domain requires the agent to know when a reset has happened, `query` after each reset to receive the position of the item, move onto that location, and finally `buy`.