Partially Observable Markov Decision Process in Reinforcement Learning

Shvechikov Pavel

National Research University Higher School of Economics,
Yandex School of Data Analysis

April 6, 2018

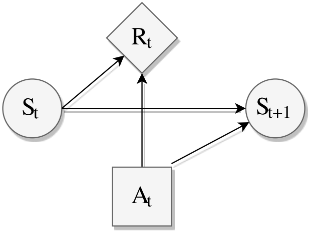
Overview

What is MDP?

Definition of Markov Decision Process

MDP is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$, where

1. $\mathcal{S}$ – set of states of the world
2. $\mathcal{A}$ – set of actions
3. $\mathcal{P} : \mathcal{S} \times \mathcal{A} \mapsto \triangle(\mathcal{S})$ – state-transition function, giving us $p(s_{t+1} \mid s_t, a_t)$
4. $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ – reward function, giving us $\mathbb{E}_R \left[R(s_t, a_t) \mid s_t, a_t \right]$.

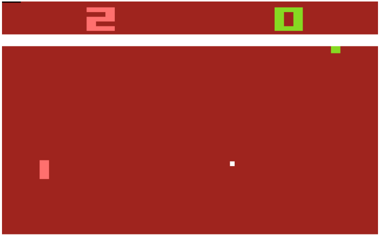


Markov property

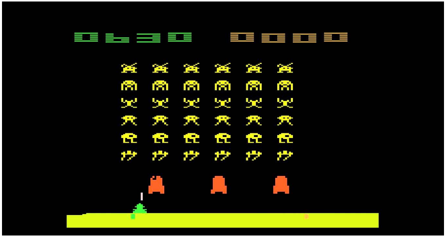
$$p(r_t, s_{t+1} \mid s_0, a_0, r_0, \ldots, s_t, a_t) = p(r_t, s_{t+1} \mid s_t, a_t)$$

(next state, expected reward) depend on (previous state, action)

MDP problems are closer than they seem



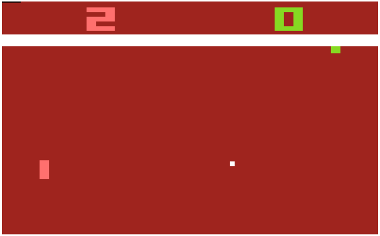
Pong



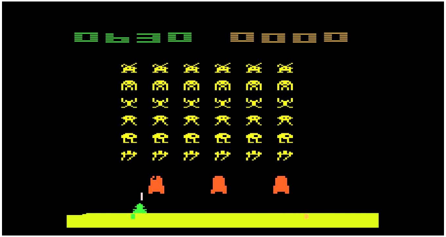
Space invaders

What is a state here?

MDP problems are closer than they seem



Pong



Space invaders

What is a state here?

128 bytes of unobserved Atari simulator RAM

Sources of uncertainty

Typically autonomous agent's state is composed of

- measurement of environment
- measurment of agent itself

In real system there is even more uncertainty:

1. imperfect self-sensing (position, torque, velocity, etc.)
2. imperfect environment perception
3. incomplete observation of environment

How to incorporate uncertainty into decision making?

POMDP is a powerful mathematical abstraction

- Industrial applications
  - Machine maintenance (Shani et al., 2009)
  - Wireless networking (Pajarinen et al., 2013)
  - Wind farms managing (Memarzadeh et al., 2014)
  - Aircraft collision avoidance (Bai et al., 2012)
  - Choosing sellers in E-marketplaces (Irissappane et al., 2016)
- Assistive care
  - Assistant for patients with dementia (Hoey et al., 2010)
  - Home assistants (Pineau et al., 2003)
- Robotics
  - Grasping with a robotic arm (Hsiao et al., 2007)
  - Navigating an office (Spaan et al., 2005)
- Spoken dialog systems
  - Uncertainty in voice recognition (Young et al., 2013)

Outline

POMDP's place in a model world

Markov Models		Do we have control over the state transitions?	
		NO	**YES**
Are the states completely observable?	**YES**	**Markov Chain**	**MDP** Markov Decision Process
	NO	**HMM** Hidden Markov Model	**POMDP** Partially Observable Markov Decision Process

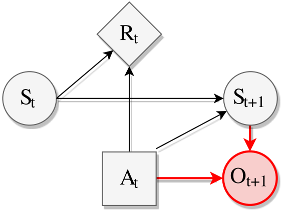
POMDP's siblings

POMDP model

Definition

Partially Observed Markov Decision Process is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \Omega, \mathcal{O} \rangle$

1. $\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}$ are the same as in MDP
2. Ω – finite *set of observations*
3. $\mathcal{O} : \mathcal{S} \times \mathcal{A} \mapsto \triangle(\Omega)$ – *observation function*, which gives $\forall (s, a) \in \mathcal{S}, \mathcal{A}$, a probability distribution over Ω, i.e. $p(o \mid s_{t+1}, a_t) \quad \forall o \in \Omega$

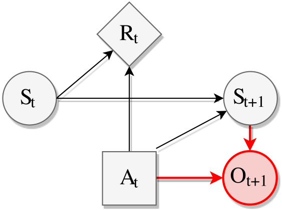


POMDP model

Definition

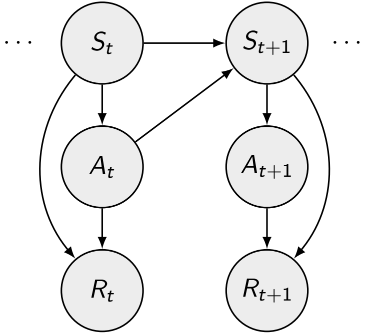
Partially Observed Markov Decision Process is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \Omega, \mathcal{O} \rangle$

1. $\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}$ are the same as in MDP
2. Ω – finite *set of observations*
3. $\mathcal{O} : \mathcal{S} \times \mathcal{A} \mapsto \triangle(\Omega)$ – *observation function*, which gives $\forall (s, a) \in \mathcal{S}, \mathcal{A}$, a probability distribution over Ω, i.e. $p(o \mid s_{t+1}, a_t) \quad \forall o \in \Omega$

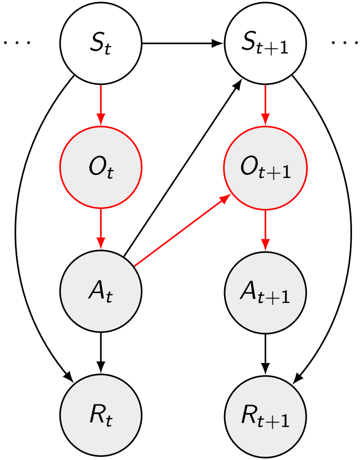


What if we ignore the partial observability?

Reactive (adapted) policies



MDP



POMDP

Adapted policies (Singh et al., 1994)

Adapted policy

is a mapping $\pi\colon \Omega \to \triangle(\mathcal{A})$

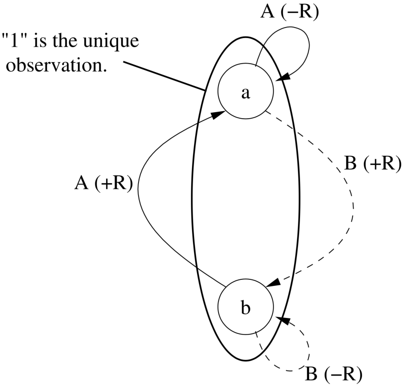
Stationary adapted π's in POMDP:

1. **deterministic** π can be arbitrarily bad compared to the best stochastic π
2. **stochastic** π can be arbitrarily bad compared to the optimal π in underlying MDP

Adapted policies (Singh et al., 1994)

Adapted policy

is a mapping $\pi : \Omega \to \triangle(\mathcal{A})$



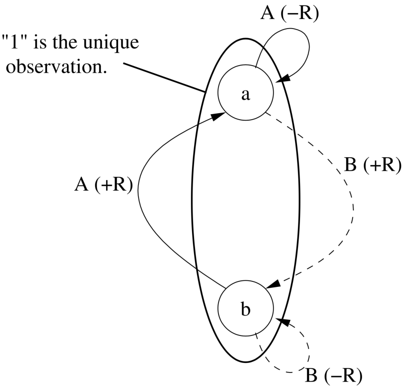
Stationary adapted π's in POMDP:

1. **deterministic** π can be arbitrarily bad compared to the best stochastic π
2. **stochastic** π can be arbitrarily bad compared to the optimal π in underlying MDP

Adapted policies (Singh et al., 1994)

Adapted policy

is a mapping $\pi : \Omega \to \triangle(\mathcal{A})$



Stationary adapted π's in POMDP:

1. **deterministic** π can be arbitrarily bad compared to the best stochastic π
2. **stochastic** π can be arbitrarily bad compared to the optimal π in underlying MDP

What maximum return is achievable for a nonstationary policy?

Sufficient information process (Striebel, 1965)

Complete information state I_t^C at time t

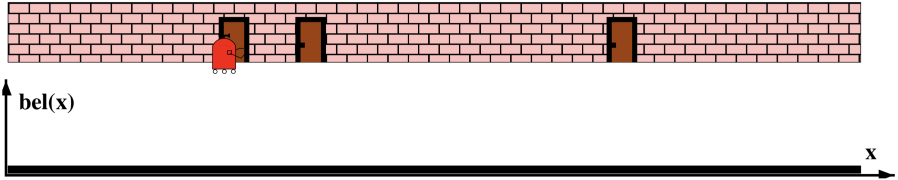
$$I_t^C = \langle \rho(s_0), o_0, a_0, \ldots, a_{t-1}, o_t \rangle$$

where $\rho(s_0)$ is a distribution over initial states

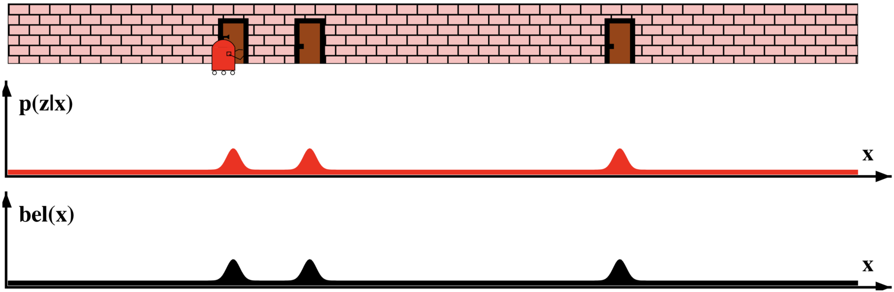
A sequence $\{I_t\}$ defines a **sufficient information process** when

1. $I_t = \tau(I_{t-1}, o_t, a_{t-1})$
   - I can be updated incrementally
2. $P(s_t | I_t) = P(s_t | I_t^C)$
   - I_t does not lose information about states
3. $P(o_t | I_{t-1}, a_{t-1}) = P(o_t | I_{t-1}^C, a_{t-1})$
   - I_t does not lose information about next observation

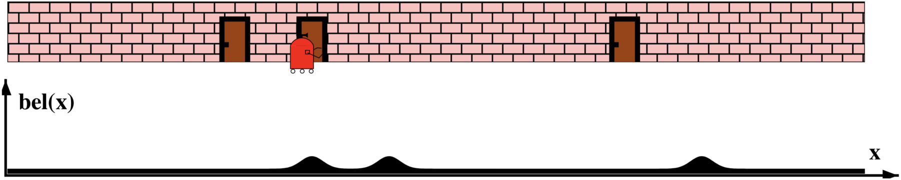
Information tracking



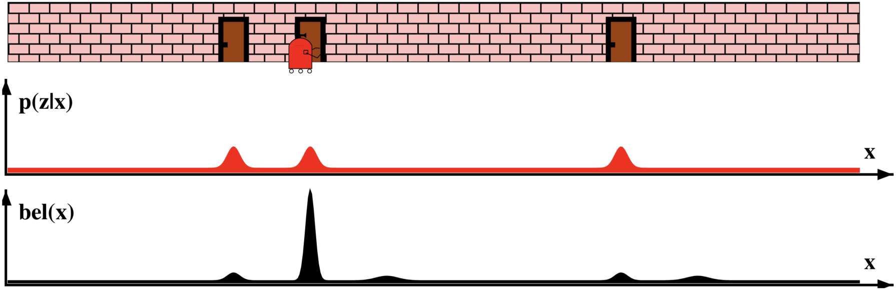
Information tracking



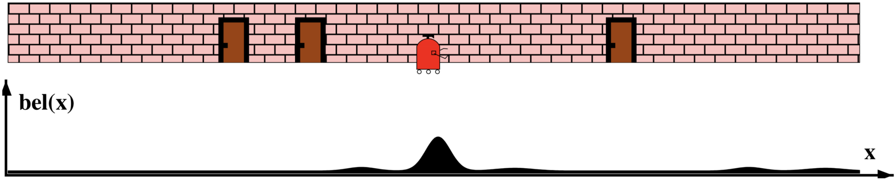
Information tracking



Information tracking



Information tracking



Belief states and their updates (Bayes filter)

Belief state – distribution over state space

$$b_t(s) \triangleq P(s_t = s \mid I_t^C)$$

POMDP is MDP over properly updated beliefs (Astrom, 1965):

$$b'(s') = p(s' \mid o', a, b) = \frac{p(s', o' \mid a, b)}{p(o' \mid a, b)}$$

$$= \frac{p(o' \mid s', a) \cdot p(s' \mid a, b)}{\sum_{s''} p(o' \mid s'', a) \cdot p(s'' \mid a, b)}$$

$$\propto p(o' \mid s', a) \sum_s p(s' \mid a, s) \cdot b(s)$$

From POMDP to MDP over beliefs

Bellman optimality equation for $V^*(s_t)$

$$V^*(s) = \max_a \left[\mathcal{R}(s, a) + \gamma \sum_{s'} p(s' \mid s, a) V^*(s') \right]$$

Bellman optimality equation for $V^*(b_t)$

$$V^*(b) = \max_a \left[\mathcal{R}(b, a) + \gamma \sum_{s'} p(b' \mid b, a) V^*(b') \right]$$

$$p(b' \mid a, b) = \sum_{o', s', s} p(b' \mid a, b, o') p(o' \mid s', a) p(s' \mid s, a) b(s)$$

$$p(b' \mid a, b, o') = \mathbb{I}(b' = \mathrm{BayesFilter}(o', a, b))$$

$$\mathcal{R}(b, a) = \sum_s b(s) \mathcal{R}(s, a)$$

Reasoning about state uncertainty

Bad news: belief updating can be computed exactly only for

1. discrete low-demensional state-spaces
2. linear-Gaussian dynamics (leading to Kalman filter), i.e.
   - $s' \sim \mathcal{N}(s' \mid T_s s + T_a a,\ \Sigma_s)$
   - $o' \sim \mathcal{N}(o' \mid O_s s' + O_a a, \Sigma_o)$
   - $R(s, a) = s^\top R_s s + a^\top R_a a$

Reasoning about state uncertainty

Bad news: belief updating can be computed exactly only for

1. discrete low-demensional state-spaces
2. linear-Gaussian dynamics (leading to Kalman filter), i.e.
   - $s' \sim \mathcal{N}(s' \mid T_s s + T_a a,\ \Sigma_s)$
   - $o' \sim \mathcal{N}(o' \mid O_s s' + O_a a, \Sigma_o)$
   - $R(s, a) = s^\top R_s s + a^\top R_a a$

What if

1. states are of a complex nature? (i.e. images)
2. state transition function is non-linear and unknown?

Possible options

1. Use advanced tracking techniques
   - Deep Variational Bayes Filter (Karl et al., 2016)
2. Just forget all the math and use LSTM / GRU
   - DRQN (Hausknecht et al., 2015), DARQN (Zhu et al., 2017), RDPG (Heess et al., 2015)
3. Preserve information with predictive state representations
   - Recurrent Predictive State Policy (Hefny et al., 2018)
4. Use human-like differentiable memory
   - Neural Map (Parisotto et al., 2017), MERLIN (Wayne et al., 2018)

Outline

1. What is wrong with MDP?
   - MDP reminder
2. POMDP details
   - Definitions
   - Adapted policies
   - Sufficient information process
3. Approximate Learning in POMDPs
   - Deep Recurrent Q-Learning
   - MERLIN

Deep Recurrent Q-Learning (DRQN)

Q-learning: $Q(s_t, a_t) = \mathbb{E}_{r,s'|s_t,a_t}\left[r + \gamma \max_{a'} Q(s', a')\right]$

Problem: we don't know s_t

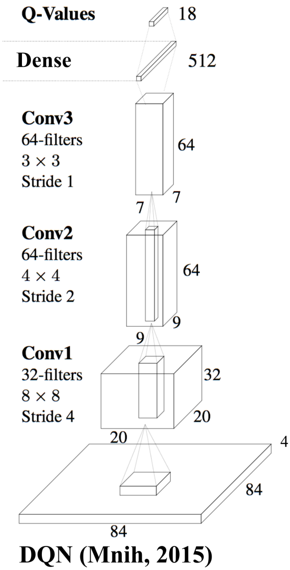
DRQN solution: (Hausknecht et al., 2015)

1. equip agent with memory h_t
2. approximate $Q(s_t, a_t)$ with $Q(o_t, h_{t-1}, a_t)$
3. eliminate dependence on o_t by modelling $h_t = LSTM(o_t, h_{t-1})$

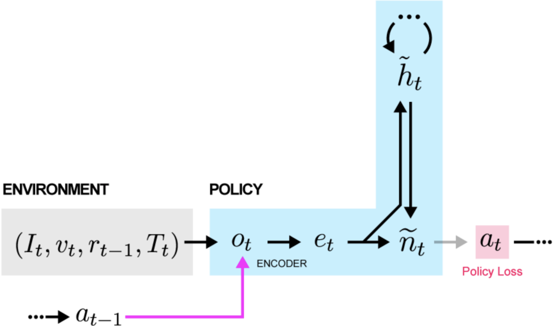
Benefits:

1. simple approximate POMDP solver with *one frame* input
2. need only to model $Q(h_t, a_t)$
3. minor changes to vanilla DQN architecture

DRQN: architecture



RNN-like memory

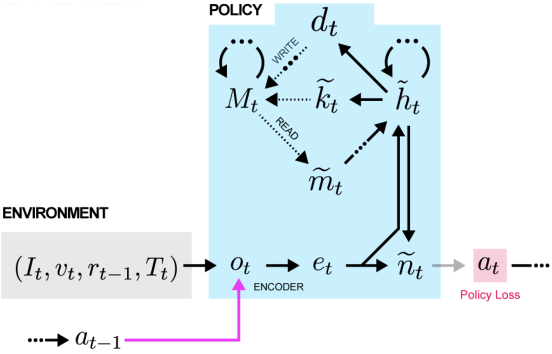


1. I_t – input image
2. v_t – agent's velocity
3. r_{t-1} – previous reward
4. T_t – text instructions

Drawbacks:

1. truncated BPTT
2. sparse reward signal

DNC-like memory



Sensory data can instead be encoded and stored without trial and error in a temporally local manner.

MERLIN (Wayne et al., 2018): design principles

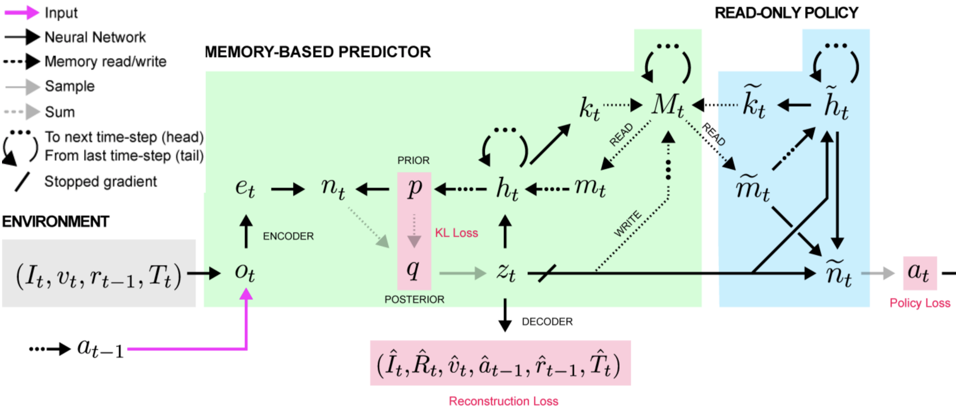
Neuroscience motivation

1. predictive sensory coding
   - brain continually generates models of the world
   - based on context and information from memory
   - to predict future sensory input
2. hippocampal representation theory (Gluck et al., 1993)
   - representations pass through a compressive bottleneck
   - then reconstruct input stimuli **together** with task reward
3. temporal context model

Under the hood:

1. Variational Autoencoders
2. Differentiable Neural Computer
3. Recurrent Asynchronous Advantage Actor Critic (A3C)
4. 57 pages long paper of 24 authors from DeepMind

MERLIN (Wayne et al., 2018): architecture



Variational Autoencoder

Each component of

$$o_t = (I_t, v_t, a_{t-1}, r_{t-1}, T_t) \in \mathbb{R}^{10\,000}$$

is independently encoded into $e_t \in \mathbb{R}^{100}$ by

(6 ResNet blocks, MLP, None, None, 1-layer LSTM)

Decoding process uses same architectures.

Return predictor decoder

- MLP: $V^\pi(z_t, \log \pi_t(a | M_{t-1}, z_{\leq t}))$ is regressed on $\hat{G}_t$
- MLP: $A(z_t, a_t)$ is regressed on $\hat{G}_t - V^\pi(\cdot, \cdot)$

Memory – simplified DNC model

- Memory is a tensor M_t with dimensions $(N_{mem},\ 2|z|)$
- Each step we write vector $[z_t, (1-\gamma)\sum_{t'>t}\gamma^{t'-t}z_{t'}]$ to M_{t-1}
- denote $m_t = [m_t^1, ..., m_t^K]$ for readout of K read heads

Reading from memory

- MBP LSTM: $[z_t, a_t, m_{t-1}] \ \rightarrow\ h_t^1$
- Policy LSTM: $[z_t] \ \rightarrow\ h_t^2$
- $\text{Linear}([h_t^1, h_t^2]) \rightarrow i_t = [k_t^1, ..., k_t^K, \beta_t^1, .., \beta_t^K]$
- $c_t^{ij} = \text{cosine}(k_t^i,\ M_{t-1}[j, \cdot])$
- $w_t^i = \text{Softmax}(\beta_t^1 c_t^{i1}, ..., \beta_t^K c_t^{iN_{mem}})$
- readout memory $m_t^i = M_{t-1}^\top w_t^i$

Memory – simplified DNC model

- Memory is a tensor M_t with dimensions $(N_{mem},\ 2|z|)$
- Each step we write vector $[z_t, (1-\gamma)\sum_{t'>t}\gamma^{t'-t}z_{t'}]$ to M_{t-1}
- denote $m_t = [m_t^1, ..., m_t^K]$ for readout of K read heads

Writing to memory is performed *after* reading:

- $v_t^{wr}[i] = \delta_{it}$
- $v_t^{ret} = \gamma v_{t-1}^{ret} + (1-\gamma)v_{t-1}^{wr}$
- $M_t = M_{t-1} + v_t^{wr}[z_t, 0]^\top + v_t^{ret}[0, z_t]^\top$
- When $t > N_{mem}$, select the cell with lowest utility

$$u_{t+k}[k] = u_t[k] + \sum_i w_{t+1}^i[k]$$

Latent space

- Prior, MLP:

$$[h_{t-1}, m_{t-1}] \;\rightarrow\; \mu_t^{prior}, \log \sigma_t^{prior}$$

- Concatenate all information from this timestep

$$n_t = [e_t, h_{t-1}, m_{t-1}, \mu_t^{prior}, \log \sigma_t^{prior}]$$

- Posterior

$$[\mu_t^{post}, \log \sigma_t^{post}] = \mathrm{MLP}(n_t) + [\mu_t^{prior}, \log \sigma_t^{prior}]$$

z_t is a sample from posterior factorized Gaussian

Loss function

Variational Lower Bound:

$$\log p(o_{\leq t}, r_{\leq t}) \geq \sum_{\tau=0}^{t} \mathbb{E}_{q(z_{<\tau} \mid o_{<\tau})} \left[\mathrm{DataTerm} - \mathrm{KL} \right]$$

$$\mathrm{DataTerm} = \mathbb{E}_{q(z_{\tau} \mid z_{<\tau}, o_{\leq\tau})} \left[\log p(o_{\tau}, r_{\tau} \mid z_t) \right]$$

$$\mathrm{KL} = D_{KL} \left(q(z_{\tau} \mid z_{<\tau}, o_{\leq\tau} \mid\mid p(z_{\tau} \mid z_{<\tau}, a_{\leq\tau}) \right)$$

Where $p(o_{\tau}, r_{\tau} \mid z_t)$ is a linear combination of 6 decoding losses

Experiments

The most interesting experiments

1. Goal oriented navigation in a maze (3-7 rooms) (video)
2. Arbitrary Visuomotor Mapping (video)
3. T-maze (video)

Thank you!

References I

- Astrom, Karl J (1965). "Optimal control of Markov processes with incomplete state information". In: *Journal of mathematical analysis and applications* 10.1, pp. 174–205.
- Bai, Haoyu et al. (2012). "Unmanned aircraft collision avoidance using continuous-state POMDPs". In: *Robotics: Science and Systems VII* 1, pp. 1–8.
- Gluck, Mark A et al. (1993). "Hippocampal mediation of stimulus representation: A computational theory". In: *Hippocampus* 3.4, pp. 491–516.
- Hausknecht, Matthew et al. (2015). "Deep recurrent q-learning for partially observable mdps". In: *arXiv preprint arXiv:1507.06527*.
- Heess, Nicolas et al. (2015). "Memory-based control with recurrent neural networks". In: *arXiv preprint arXiv:1512.04455*.

References II

- Hefny, Ahmed et al. (2018). "Recurrent Predictive State Policy Networks". In: *arXiv preprint arXiv:1803.01489*.
- Hoey, Jesse et al. (2010). "Automated handwashing assistance for persons with dementia using video and a partially observable Markov decision process". In: *Computer Vision and Image Understanding* 114.5, pp. 503–519.
- Hsiao, Kaijen et al. (2007). "Grasping pomdps". In: *Robotics and Automation, 2007 IEEE International Conference on*. IEEE, pp. 4685–4692.
- Irissappane, Athirai Aravazhi et al. (2016). "A Scalable Framework to Choose Sellers in E-Marketplaces Using POMDPs.". In: *AAAI*, pp. 158–164.
- Karl, Maximilian et al. (2016). "Deep variational bayes filters: Unsupervised learning of state space models from raw data". In: *arXiv preprint arXiv:1605.06432*.

References III

- Memarzadeh, Milad et al. (2014). "Optimal planning and learning in uncertain environments for the management of wind farms". In: *Journal of Computing in Civil Engineering* 29.5, p. 04014076.
- Pajarinen, Joni et al. (2013). "Planning under uncertainty for large-scale problems with applications to wireless networking". In:
- Parisotto, Emilio et al. (2017). "Neural Map: Structured Memory for Deep Reinforcement Learning". In: *arXiv preprint arXiv:1702.08360*.
- Pineau, Joelle et al. (2003). "Towards robotic assistants in nursing homes: Challenges and results". In: *Robotics and autonomous systems* 42.3, pp. 271–281.
- Shani, Guy et al. (2009). "Improving existing fault recovery policies". In: *Advances in Neural Information Processing Systems*, pp. 1642–1650.

References IV

- Singh, Satinder P et al. (1994). "Learning without state-estimation in partially observable Markovian decision processes". In: *Machine Learning Proceedings 1994*. Elsevier, pp. 284–292.
- Spaan, Matthijs TJ et al. (2005). "Perseus: Randomized point-based value iteration for POMDPs". In: *Journal of artificial intelligence research* 24, pp. 195–220.
- Striebel, Charlotte (1965). "Sufficient statistics in the optimum control of stochastic systems". In: *Journal of Mathematical Analysis and Applications* 12.3, pp. 576–592.
- Wayne, Greg et al. (2018). "Unsupervised Predictive Memory in a Goal-Directed Agent". In: *arXiv preprint arXiv:1803.10760*.
- Young, Steve et al. (2013). "Pomdp-based statistical spoken dialog systems: A review". In: *Proceedings of the IEEE* 101.5, pp. 1160–1179.

References V

Zhu, Pengfei et al. (2017). "On Improving Deep Reinforcement Learning for POMDPs". In: *arXiv preprint arXiv:1704.07978*.