Review on: *Deep Reinforcement Learning with POMDPs* (http://cs229.stanford.edu/proj2015/363_report.pdf)

by **Jilan Samiuddin**

July 24, 2020

Problem Motivation

The motivation behind this project was to integrate Deep Q-Network (DQN) with RL for POMDPs without having to be limited in 2D state-space unlike previous literature. The novel approach learns policies by mapping belief states into optimal policies via DQN. Furthermore, the DQN can also use the action-observation history to find optimal policies.

Problem Formulation

A POMDP model is the simple concept that in real world an agent does not have the full information of its state, rather it uses its observations and rewards (that it gains in the process) to make decisions. Thus a POMDP model is formally presented by the tuple (S,A,T,R,Z,O) where S,A,T,R are states, actions, transition and rewards like that of an MDP, and Z,O are observations and observation model rrespectively. The belief state b in a POMDP is updated using Bayes theorem. The optimal value function is piecewise-linear convex function of the form $V(b) = \max_{\alpha \in \Gamma}(\alpha . b)$ where Γ is a set of alpha-vectors presenting the POMDP policy.
A neural network is used to estimate the POMDP value function over the belief space, which in general becomes intractable using just Q-learning method. In contrast to MDP, where the Q-values are parameterized by state and action, in POMDP it is parameterized by either the belief and the action $Q(b, a \in A|\theta)$, or the action-observation history h and the action $Q(h,a|\theta)$, where θ is the weight (and biases) of the neural network. The loss function is thus defined as

$$\mathcal{L}(b,a|\theta_i) = \Big(r + \gamma \max_a Q(b',a|\theta_i) - Q(b,a|\theta_i)\Big)^2 .$$

Method

Random batches of experience replay tuples (b,a,r,b') are used to train the network. The random batch strategy helps break the correlation between the state transitions and thus allowing the neural network learn more effectively. A second network is used as the target network that the primary network uses to update θ. An adaptive learning technique called the RMSProp is used to adjust the network parameters. The architecture of the setup is shown below. The architecture allows usage of fully observable states as well as the belief state and outputs utility for each action available to the agent. Such models are known as mixed observability MDPs (MDOMPs).

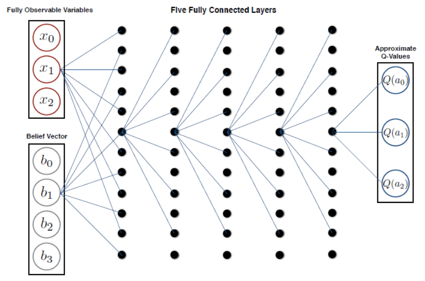

Figure 1: Five layer fully connected network that maps the concatenated fully observable variable and belief vectors to Q-values

Simulation and results

The framework was tested on two benchmark problems. First the Tiger problem (POMDP structure) in which there are two doors – behind one door is a tiger (with a penalty of -100) and a prize behind the other (a reward of $+10$). The agent does not know behind which door is the tiger and thus there are two states – the tiger is behind the left door or behind the right door. The agent can either listen, open the left door or open the right door. However, listening gives a noisy observation of the tiger's position. The value function obtained for the Tiger problem through the described technique is compared to that of obtained by SARSOP – a popular state-of-the-art solver. It can be seen that the policy obtained by the DQN, after converging, resembles the optimal policy from SARSOP, however, when the DQN did not converge, the policy did not work out when the belief about the tiger being on the left is concrete. The next problem is the Rock sample problem (MOMDP structure with 12000 states, 13 actions and 2 observations) and is also solved using the DQN with satisfactory results.

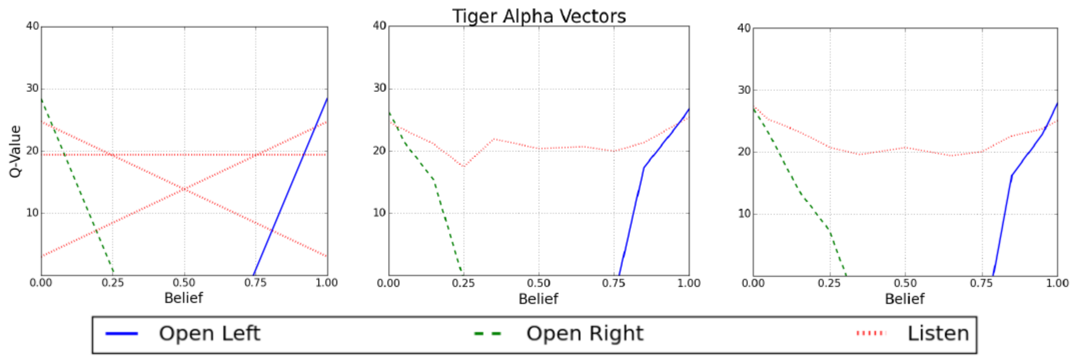

Figure 2: The value function surfaces for the Tiger problem for the SARSOP alpha-vectors (left), DQN converged policy (middle), and DQN non-converged policy (right)

The DQN converged to near-optimal policy afte 5 minutes and 6 hours for the Tiger and the Rock sample problems respectively, whereas, it took only 1 second and 5 minutes for SARSOP to do the same task. Even though the DQN is pretty slow compared to SARSOP, the DQN did not require explicit knowledge of the model unlike SARSOP.
For both the belief approach and the history approach, the policies obtained were evaluated and compared to the SARSOP policies and the POMDP RL approach from [1] which is shown in the Table below.

	Belief DQN	History DQN	SARSOP	POMDP RL
Tiger	1.08 ± 0.1	1.07 ± 0.1	1.12 ± 0.1	1.06 ± 0.1
Rock Sample	1.65 ± 0.1	1.43 ± 0.2	1.75 ± 0.1	1.14 ± 0.2

Table: Average rewards per time-step for the Tiger and the Rock Sample problems using the SARSOP, DQN and POMDP RL policies. The rewards ere averaged over 100 trials.

Suggested future work

Use policy gradient technique instead of value iteration which would provide faster convergence.

Reference

[1] J. Baxter and P. L. Bartlett, \Reinforcement learning in pomdp's via direct gradient ascent," in *In Proc. 17th International Conf. on Machine Learning*. Morgan Kaufmann, 2000, pp. 41-48.