6.882 Progress Report: Bayesian Reinforcement Learning

Vickie Ye and Alexandr Wang

1 Introduction

In reinforcement learning, the learning procedure consists of two parallel processes: estimating parameters of the surrounding environment and learning the optimal policy of highest long-term reward. In this project, we wanted to examine both of these processes with a Bayesian approach. In this half, we used a Bayesian framework to estimate the parameters of the learner's environment in a Markov decision process (MDP). We followed Strens (2000) in this section. For the next half of the project, we will use Gaussian process temporal difference (GPTD) learning as a Bayesian solution to the policy evalution. We will follow Engel et al. (2005) in this section.

2 Bayesian MDP

MDPs are commonly used to learn the policy for the system with a set of states S, a set of actions A, a reward function $R(S, A)$, and a transition function $T(s, a, s') = P(X^{(t+1)} = s'|X^{(t)} = s, Y^{(t)} = a)$. To learn the optimal long-term-reward policy, we define a quality function with discount factor γ, $Q = \sum_{t=0}^{\infty} \gamma^t R^(t)$, which we approximate for each state-action pair as

$$Q(s, a) = \mathbb{E}[R(s, a)] + \gamma \sum_{s'} T(s, a, s') \max_{a'} Q(s', a')$$

The quantities we need to estimate are then expected returns $\mathbb{E}[R(s, a)]$ and transition probabilities $T(s, a, s')$ to make our updates to Q.

2.1 Models for Transition Probabilities and Expected Return

We represent the transition probabilities for each state-action pair (s, a) as a multinomial distribution over the N states of the system,

$$\pi(s, a) = (T(s, a, s_0), ..., T(s, a, s_{N-1}))$$

In Strens (2000), the author considers a Dirichlet prior and a second "sparse" uniform prior. For our experiments we used a Dirichlet prior with $\alpha_i = 1$. Our updated posterior for each $\pi(s, a)$ is then

$$\pi^{(t)} \sim \text{Dirichlet}(\alpha^{(t)}|\mathbf{n}^{(t)}), \alpha_i^{(t)} = \alpha_i + n_i^{(t)}$$

Although a sparse prior might have its benefits, for the problems described in the paper, which we reproduced here, we did not feel the need to enforce sparseness into our transition matrix. Then in estimating Q at each time step, we sampled $\pi(s, a)$ for from our posterior.

We represent the expected return for each state-action pair (s, a) as Gaussian- distributed with mean μ and precision τ. We use a $\text{Ga}(\beta, \rho)$ prior for τ and a $\mathcal{N}(\mu_0, c_0\tau)$ prior for μ. Then our updated posteriors are

$$\tau \sim \text{Ga}\Big(\beta + \frac{n}{2}, \rho + \frac{1}{2}\sum_i (x_i - \bar{x})^2 + \frac{nc_0(\bar{x} - \mu_0)^2}{2(n + c_0)}\Big),$$

$$\mu \sim \mathcal{N}\Big(\frac{n\bar{x} + c_0\mu_0}{n + c_0}, (n + c_0)\tau\Big)$$

Then in estimating Q at each time step, we sampled μ and τ, which we then used to sample $\mathbb{E}[R(s, a)]$, for each (s, a).

For comparison (and debugging) we also used the modes of our posterior distributions as estimates for $\pi(s, a)$ and $\mathbb{E}[R(s, a)]$.

2.2 Testing Problems

We used three toy problems to test our implementation. In the "Chain" problem (Figure 1a), there are five states and two possible actions, with 0.2 probability of slipping (performing the opposite action intended). The optimal policy is to explore to state 5, and stay there until slipping; the agent should try to get back to state 5 as much as possible.

In the "Loop" problem (Figure 1a), there are nine states and two possible actions, with no possibility of slipping. The optimal policy is to stay in the left loop and receive reward of 2 for every 5 actions. Because this problem has very little distinguishing the two actions in terms of short-term reward, effective estimation of Q parameters is important to optimal behavior.

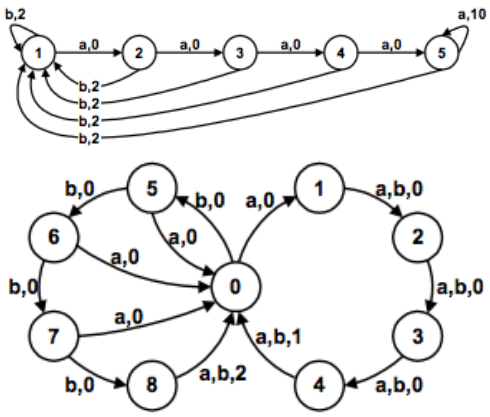

Figure 1: The "Chain" and "Loop" toy problems used in testing.

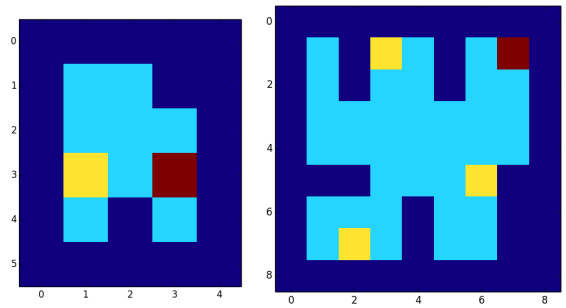

Figure 2: The "Maze" flag-collecting toy problem used in testing. Yellow squares are flags; red is the goal.

In the "Maze" problem (Figure 2), the agent explores a maze to find flags. The agent can either move north, west, south, or east, and receives only its current position as its state (it has no physical sensors or observations of walls, besides the its new state). Each time the agent reaches the goal, it receives a reward equal to the number of flags collected and is transported back to the beginning of the maze. All other state-action pairs receive zero reward. In the maze problem, we also have a probability of slipping (performing an action other than the one specified) of 0.2.

2.3 Preliminary Results

References

Engel, Y., Mannor, S., and Meir, R. (2005). Reinforcement learning with gaussian processes. In *International Conference on Machine Learning*.

Strens, M. (2000). A bayesian framework for reinforcement learning. In *International Conference on Machine Learning*.