Reinforcement Learning: Policy Updates & Policy Gradients Quiz

Ethan B. Mehta
ethanbmehta@berkeley.edu

Sean Lin
seanlin2000@berkeley.edu

Jaiveer Singh
j.singh@berkeley.edu

1 MDP Questions

1.1 Q1

What are the 5 components of an MDP? Describe each one.

Answer:

- States: States represent the status of all the elements in the environment, encapsulating all the information that may change from time to time.
- Actions: Actions are the controllable choices an Agent can make when in a State, in order to initiate a Transition to a subsequent State.
- Transition Function: The Transition Function is a probability distribution that identifies how likely it is for the Agent to be transferred to a specific next state after the Agent takes a specific action from a specific starting state.
- Reward Function: The Reward Function gives the mapping from (state, action, next state) to rewards.
- Discount Factor: A discount factor multiplicatively reduces the magnitude of future rewards, incentivizing earlier rewards over later rewards and providing a convergence guarantee.

1.2 Q2

How is a Q-State different than a State? In what special cases would the Q-States and States be equivalent? *Hint: consider the size of the State and Action Spaces.*

Answer: Q-States represent a pair of (state, action), while State simply encapsulates the status of the environment at a specific timestep. In the case where there is exactly 1 Action that can be taken from every State, the Q-States and States are functionally equivalent.

1.3 Q3

Compute the size of the state space for the game of Monopoly with 2 players, 8 properties in total, 4 corner spaces (non-property locations). Each property can be owned by exactly 1 player or no player. Each player's token can be on one of the properties, or one of the corner spaces.

Answer:

$$(\#\text{ of places for a player token})^{\#\text{ of players}} \times \#\text{ of property ownership states}^{\#\text{ of properties}} = (8+4)^2 \times (2+1)^4 = 11664$$

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

2 Offline Learning

2.1 Q4

Mortimer and Ellis are debating the formulation of the Bellman Equation.

1. Mortimer: The Bellman Equation is
$$V^*(s) = \max_{a \in actions} \sum_{s' \in states} T(s, a, s')[R(s, a, s') + \gamma \times V^*(s')]$$
, because the optimal value of a state is the expected value of the Reward that an Agent starting from state s and behaving optimally afterwards will receive over its entire lifetime.
2. Ellis: I agree with your reasoning, but the Bellman Equation is
$$V^*(s) = \max_{a \in actions} Q(s, a)$$

Who is right? Why?

Answer: Both Mortimer and Ellis are correct. Their formulations are equivalent. Ellis has substituted in for the Q-Value.

2.2 Q5

Anant, Jennifer, and Jitendra are sharing their favorite method of learning policies.

1. Anant: My favorite method of learning policy is value iteration followed by policy extraction. I know that I always need my values to converge before I can figure out a policy.
2. Jennifer: My favorite method of learning policy is policy iteration. I don't like waiting for my values to converge because all I really need is a policy. If I do policy iteration, I can get the optimal policy without doing any value iteration!
3. Jitendra: My favorite method of learning policy is Q-Learning. I really like MDPs, so with Q-Learning I can learn the underlying MDP's transition functions and reward functions and then extract the optimal policy.

Why are they all wrong?

Answer:

1. Anant's mistake is that the policy often converges before the precise values converge. He doesn't need to wait for value iteration to fully converge before extracting the optimal policy.
2. Jennifer's mistake is that policy iteration still requires value iteration, since the per-policy values need to be updated each time the policy is updated.
3. Jitendra's mistake is that Q-Learning is a model-free approach, and so this method does not try to learn the underlying MDP at all.

2.3 Q6

Why is the discount factor useful? What is the range of useful discount factors?

Answer: The discount factor provides a convergence guarantee for the Bellman Equation by ensuring that future rewards are worse than equal-magnitude present rewards by a multiplicative factor. In order to be useful, $0 < \gamma < 1$, to avoid blowing up the future rewards or diminishing or inverting them entirely.

3 Online Learning

3.1 Q7

How does the Direct Evaluation algorithm incorporate information about the specific trajectory of States and Actions that the Agent takes? Is this a strength or a weakness of the Direct Evaluation method?

Answer: Direct Evaluation does not take into account the specific trajectory of States and Action, only the ultimate rewards achieved through each State and the counts of visiting each State. This is a major weakness of Direct Evaluation; it does not learn at every Transition, and can only update at the end of a series of episodes.

3.2 Q8

Considering Temporal Difference Learning, write the equations for each of the following. **Be sure to define all constants and Greek letters you reference.**

- Constructing a sample after a Transition has taken place
- Incorporating the sample into the estimated Values

Answer:

$$\text{sample} = R(s, a, s') + \gamma \times V^{\pi}(s')$$
$$V^{\pi}(s) \leftarrow (1 - \alpha) \times V^{\pi}(s) + \alpha \times \text{sample}$$

γ is the discount factor, and α is the learning rate.

3.3 Q9

Jethro tried to develop a new RL algorithm called J-Learning. The pseudocode of the J-Learning algorithm follows:

1. Initialize all Q-Values as 0
2. Repeat for the number of exploration episodes:
   (a) While the current State is not terminal:
      i. Use the current policy to determine the best Action
      ii. Transition to the new State using the calculated Action
      iii. Construct a sample using the incurred Reward
      iv. Update Q-Value using the new sample and learning rate alpha, with the same update equations as normal Q-Learning

Unfortunately, Jethro has made a critical error in this modification of Q-Learning. Explain the nature of Jethro's mistake, what step in the algorithm the error appears at, and what an effective solution might be.

Answer: The error occurs at step 2ai. Jethro never randomizes the action taken in J-Learning, which means that the Agent will only stick to the States it has already explored. As a result, the Agent will not fully explore the State Space, and may not find the optimal solution. To fix this, Jethro could use ϵ-Greedy to randomize with the small probability ϵ, as is used in regular Q-Learning.

4 Deep Reinforcement Learning

4.1 Q10

Alice and Bob both skimmed through the Note and have different understandings of what Policy Gradients are.

1. Alice: Policy Gradients involves computing the gradient of the Value function at each State, and then using Gradient Ascent to iteratively improve these estimates. The learned policy is Deterministic, implemented using an argmax operation over the Q-States.
2. Bob: Policy Gradients involves computing the gradient of the Policy parameters at each State, and then using Gradient Descent to iteratively improve these estimates. The learned policy is Stochastic, implemented using a Classification Neural Net.

Which parts of each student's explanations are correct? Formulate a concise explanation for the main idea behind Policy Gradients.

Answer: Policy Gradients involves computing the gradient of the Policy parameters at each State, and then using Gradient Ascent to iteratively improve these estimates. The learned policy is Stochastic, implemented using a Classification Neural Net.

4.2 Q11

Briefly explain each part of A3C's full name: Asynchronous Advantage Actor-Critic.

Answer:

1. Asynchronous: A3C uses multiple asynchronous worker Agents that work on their own, local copies of the model and environment. These Agents periodically update the global model once they find an improvement.
2. Advantage: A3C learns the Advantage instead of the Value function, exhibiting a preference to explore 'promising' areas of the State Space in which the actual Rewards were higher than the expected Rewards, instead of simply exploring the areas with high expected Rewards.
3. Actor-Critic: A3C uses an Actor-Critic system, in which the Actor attempts to choose the best Action given the current State, and in which the Critic evaluates the goodness of the Actor's choice.

4.3 Q12

What is one benefit of using OpenAI's Gym environment? Why is it useful for us to use the same environments as other RL researchers and engineers?

Answer: Using the OpenAI Gym environment provides a standardized set of problems to compare various RL algorithms. By ensuring that researchers across the world are using the same set of environments, the Gym makes it easy to identify which kinds of problems a new algorithm is better or worse at compared to the state-of-the-art.