CS6700 : Reinforcement Learning
Written Assignment #2

Topics: Adv. Value-based methods, POMDP, HRL **Deadline**: 30 April 2023, 11:59 pm
Name: Vishnu Vinod **Roll Number:** CS19B048

- This is an individual assignment. Collaborations and discussions are strictly prohibited.
- Be precise with your explanations. Unnecessary verbosity will be penalized.
- Check the Moodle discussion forums regularly for updates regarding the assignment.
- Type your solutions in the provided LaTeXtemplate file.
- **Please start early.**

1. (3 marks) Recall the four advanced value-based methods we studied in class: Double DQN, Dueling DQN, Expected SARSA. While solving some RL tasks, you encounter the problems given below. Which advanced value-based method would you use to overcome it and why? Give one or two lines of explanation for 'why'.

   (a) (1 mark) Problem 1: In most states of the environment, choice of action doesn't matter.

   Solution: Duelling DQN
   In such cases, the Duelling Deep Q-Network (DQN) algorithm can be employed, as it provides a more generalized learning approach that focuses on updating the relevant states. This is achieved by using two estimators - the advantage A(s,a) and the value V(s) - in a dueling architecture. By explicitly separating these estimators, the dueling architecture can learn which states are valuable without having to learn the effect of each action for each state. This approach improves the efficiency of the learning process by reducing the need for exhaustive exploration of the entire state space.

   (b) (1 mark) Problem 2: Agent seems to be consistently picking sub-optimal actions during exploitation.

   Solution: Double DQN
   Maximization bias is a common issue in reinforcement learning, which can be mitigated through the use of Double Deep Q-Network (Double DQN). This method employs two value-approximators that are updated with respect to each other, thereby reducing the bias. By minimizing the overestimation of action values, Double DQN improves the accuracy and stability of the Q-learning algorithm, making it an effective tool for a variety of reinforcement learning tasks.

   (c) (1 mark) Problem 3: Environment is stochastic with high negative reward and low positive reward, like in cliff-walking.

Solution: Expected SARSA
In situations where it is desirable for the agent to select a safer path, such as in cliff-walking scenarios, Expected SARSA can be employed. This approach utilizes expected values instead of actual actions to update the policy, incorporating the stochastic nature of the environment and resulting in convergence towards safer paths. Expected SARSA outperforms traditional SARSA in stochastic environments due to its ability to mitigate high variance.

2. (4 marks) Ego-centric representations are based on an agent's current position in the world. In a sense the agent says, I don't care where I am, but I am only worried about the position of the objects in the world relative to me. You could think of the agent as being at the origin always. Comment on the suitability (advantages and disadvantages) of using an ego-centric representation in RL.

Solution:
Egocentric representations are a useful approach in reinforcement learning, as they provide several advantages over other representations as listed below:

- **Computational Advantage**
  They reduce the computational requirements, as they only consider the immediate environment around the agent. This reduces the amount of data the agent needs to process, enabling faster learning and better scalability to larger environments.
- **Safety first**
  They also enable quick learning of negative and positive rewards by allowing the agent to avoid highly negative reward states (like falling off a cliff) and to quickly reach the goal states with positive rewards.

However, there are certain limitations to egocentric representations. These disadvantages are listed below:

- **Short-Sightedness**
  Learning in egocentric representations is primarily focused on immediate rewards, rather than long-term rewards. This can lead to non-convergence when attempting to impose high gamma values.
- **Limited Knowledge**
  They may not perform optimally in certain situations, where it is important to know the exact location of the agent within the environment. In such scenarios, egocentric representations may lack the necessary knowledge to perform optimally.

It is however important to note that in the real world, agents often rely on sensors to determine their surroundings. Therefore, while egocentric representations may have limitations, they are still a valuable tool for reinforcement learning in real-world applications.

3. (12 marks) Santa decides that he no longer has the memory to store every good and bad deed for every child in the world. Instead, he implements a feature-based linear function approximator to determine if a child gets toys or coal. Assume for simplicity that he uses only the following few features:

- Is the child a girl? (0 for no, 1 for yes)
- Age? (real number from $0 - 12$)
- Was the child good last year? (0 for no, 1 for yes)

- Number of good deeds this year
- Number of bad deeds this year

Santa uses his function approximator to output a real number. If that number is greater than his good threshold, the child gets toys. Otherwise, the child gets coal.

(a) (4 marks) Write the full equation to calculate the value for a given child (i.e., $f(s, \vec{\theta}) = \ldots$), where s is a child's name and $\vec{\theta}$ is a weight vector $\vec{\theta} = (\theta(1), \theta(2), \ldots, \theta(5))^{\mathrm{T}}$. Assume child s is described by the features given above, and that the feature values are respectively written as ϕ_s^{girl}, ϕ_s^{age}, ϕ_s^{last}, ϕ_s^{good}, and ϕ_s^{bad}.

Solution:
The full equation to calculate the value for a given child is given by:

$$f(s, \vec{\theta}) = \vec{\phi}^T \cdot \vec{\theta} = [\phi_s^{girl}, \phi_s^{age}, \phi_s^{last}, \phi_s^{good}, \phi_s^{bad}] \cdot [\theta(1), \theta(2), \ldots, \theta(5)]^T$$

Upon expanding this we get,

$$f(s, \vec{\theta}) = \theta(1) \cdot \phi_s^{girl} + \theta(2) \cdot \phi_s^{age} + \theta(3) \cdot \phi_s^{last} + \theta(4) \cdot \phi_s^{good} + \theta(5) \cdot \phi_s^{bad}$$

The decision boundary here will be $f(s, \vec{\theta}) = 0$. If $f(s, \vec{\theta}) > 0$ then the child gets the gift else if $f(s, \vec{\theta}) < 0$ the child doesn't get a gift.

(b) (4 marks) What is the gradient $\left(\nabla_{\vec{\theta}} f(s, \vec{\theta})\right)$? I.e. give the vector of partial derivatives

$$\left(\frac{\partial f(s, \vec{\theta})}{\partial \theta(1)}, \frac{\partial f(s, \vec{\theta})}{\partial \theta(2)}, \ldots, \frac{\partial f(s, \vec{\theta})}{\partial \theta(n)} \right)^{\mathrm{T}}$$

based on your answer to the previous question.

Solution: The gradient can be given as:

$$\nabla_{\vec{\theta}} f(s, \vec{\theta}) = \vec{\phi} = [\phi_s^{girl}, \phi_s^{age}, \phi_s^{last}, \phi_s^{good}, \phi_s^{bad}]^T$$

(c) (4 marks) Using the feature names given above, describe in words something about a function that would make it impossible to represent it adequately using the above linear function approximator. Can you define a new feature in terms of the original ones that would make it linearly representable?

Solution:
All decision boundaries cannot be accurately represented by a linear approximator, especially when they involve multiplication of variables to remove linearity of features.

Consider the case of a new feature - which can be seen as a product of the age and number of bad deeds to penalize older kids for bad deeds more (since they ought to know better). To address this issue, new features can be added to the input space, such as the product of the age and number of bad deeds of each kid. With this expanded feature space, the decision boundary can be transformed into a linear form.

Alternatively, a more general approach would be to use kernel methods in Support Vector Machines (SVMs), which leverage the kernel trick to map the input feature space into higher dimensions, potentially an infinite-dimensional space, where the input data can be linearly separable.

4. (5 marks) We typically assume tabula rasa learning in RL and that beyond the states and actions, you have no knowledge about the dynamics of the system. What if you had a partially specified approximate model of the world - one that tells you about the effects of the actions from certain states, i.e., the possible next states, but not the exact probabilities. Nor is the model specified for all states. How will you modify Q learning or SARSA to make effective use of the model? Specifically describe how you can reduce the number of *real* samples drawn from the *world*.

Solution:
In RL, incorporating prior knowledge about the problem can lead to faster convergence to optimal policies and reduce the need for exploration. One way to do this is to use heuristic q-values of state-action pairs. Theese q-values can then be used to initialize the q-tables of Q-learning or SARSA algorithms. Another approach is to eliminate redundant states that do not lead to the goal state or lie on a loop, which can be done by truncating such state sequences. However, this approach requires complete knowledge of the state space, which may not always be feasible. Thus, while incorporating prior knowledge can be beneficial, it also poses challenges, such as incomplete knowledge and potential bias or overfitting.

Example:
For instance, consider the taxi example in PA-3, where the walls of the environment can be used to guide the agent's learning. We can incorporate this information by removing actions that would ram the taxi onto the walls or by initializing such state-action pairs with highly negative values. Similarly, we can initialize state-action pairs near the goal state with highly positive values. This can help the agent to converge to the optimal policy faster and with fewer samples.

5. (4 marks) We discussed Q-MDPs in the class as a technique for solving the problem of behaving in POMDPs. It was mentioned that the behavior produced by this approximation would not be optimal. In what sense is it not optimal? Are there circumstances under which it can be optimal?

Solution:
When solving Partially-Observable MDPs (POMDPs) using Q-MDPs, only the partially observable states are considered. However, when Q-MDPs provide optimal solutions based on the input provided, they do not account for partial observability ie. they assume that one step of control leads to full observability. This is usually false and can lead to convergence to suboptimal policies over the entire state space.

For instance, two states with the same representation in the Q-MDP framework, such as one near a cliff and the other near a goal, may have the same q-value, leading to cancelling updates to their q-values when applying Q-MDP and resulting in a suboptimal policy. However, if states with the same representation are symmetric, such as both being close to a cliff, this method can become optimal.

6. (3 marks) This question requires you to do some additional reading. Dietterich specifies certain conditions for safe-state abstraction for the MaxQ framework. I had mentioned in class that even if we do not use the MaxQ value function decomposition, the hierarchy provided is still useful. So, which of the safe-state abstraction conditions are still necessary when we do not use value function decomposition.

Solution:
In the paper[1], Dietterich proposed a hierarchical reinforcement learning framework called MAXQ, which involves decomposing a complex task into a hierarchy of subtasks. To maintain this hierarchy, Dietterich listed five conditions for safe state abstraction, which ensure that the subtasks are well-defined and can be executed safely:

- **Subtask Irrelevance**: The value function of a subtask should be independent of the state variables that are irrelevant to the subtask.
- **Leaf Irrelevance**: The value function of a leaf subtask (i.e., a subtask that cannot be decomposed further) should be independent of the state variables that are irrelevant to the subtask.
- **Result Distribution Irrelevance**: The distribution of outcomes from executing a subtask should be independent of the state variables that are irrelevant to the subtask.
- **Termination**: The termination condition specifies when a subtask is considered complete.
- **Shielding**: The shielding condition ensures that the execution of a subtask does not affect the state variables that are irrelevant to the subtask.

The first two tasks ie. **Subtask Irrelevance and Leaf Irrelevance** are still necessary to maintain the state heirarchy when using state abstraction without value function decomposition. However the last three remove the need for maintaining a complete function, and are not necessary for this.

7. (4 marks) One of the goals of using options is to be able to cache away policies that caused interesting behaviors. These could be rare state transitions, or access to a new part of the state space, etc. While people have looked at generating options from frequently occurring states in a goal-directed trajectory, such an approach would not work in this case, without a lot of experience. Suggest a method to learn about interesting behaviors in the world while exploring. [*Hint: Think about pseudo rewards.*]

Solution: Intrinsic Motivation
Discovering rare but interesting behaviors in a reinforcement learning setting can be a challenging task, as these behaviors may not necessarily be associated with high reward values. Such behaviors may manifest as infrequent state transitions, potentially located in bottlenecks of the state space, with only one or very few valid incoming and outgoing transitions. The rarity of these behaviors can make them difficult to detect during the exploration phase of the learning process, requiring specific techniques to encourage the agent to explore and identify them.

Pseudo-rewards are additional reward signals that are defined based on some specific features of the state or action space that are considered interesting or informative. Intrinsic motivation is a class of pseudo-rewards that are based on the agent's own internal state or goals, rather than external rewards or punishments. For example, an agent might receive a pseudo-reward for visiting a state that it has not visited before, or for taking an action that leads to a novel or unexpected outcome.

By incentivizing these kinds of pseudo-rewards, we encourage the agent to explore the environment more effectively and discover interesting behaviors.

References

[1] Thomas Dietterich. State abstraction in maxq hierarchical reinforcement learning. In S. Solla, T. Leen, and K. Müller, editors, Advances in Neural Information Processing Systems, volume 12. MIT Press, 1999.