CS6700 : Reinforcement Learning
Written Assignment #2

Deadline: ?

- 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.
- **Please start early.**

AUTHOR : Manoj Bharadhwaj.S
ROLL NUMBER : CS20Z003

1. (3 points) Consider a bandit problem in which the policy parameters are mean μ and variance σ of normal distribution according to which actions are selected. Policy is defined as $\pi(a;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(a-\mu)^2}{2\sigma^2}}$. Derive the parameter update conditions according to the REINFORCE procedure (assume baseline is zero).

Solution: From the question, the policy,
$\pi(a;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(a-\mu)^2}{2\sigma^2}}$
Baseline $= 0$
On applying partial derivatives, we get:
$\frac{\partial \log(\pi_t(a))}{\partial \mu} = \frac{(a-\mu_t)}{\sigma^2}$
$\frac{\partial \log(\pi_t(a))}{\partial \sigma} = \frac{-1}{\sigma_t} + \frac{(a-\mu_t)^2}{\sigma^3}$
$b_t = 0$
Thus,
$\mu_{t+1} = \mu_t + \alpha r_t(\frac{(a-\mu_t)}{\sigma^2})$
$\sigma_{t+1} = \sigma_t + \alpha r_t(\frac{-1}{\sigma_t} + \frac{(a-\mu_t)^2}{\sigma^3})$

2. (6 points) Let us consider the effect of approximation on policy search and value function based methods. Suppose that a policy gradient method uses a class of policies that do not contain the optimal policy; and a value function based method uses a function approximator that can represent the values of the policies of this class, but not that of the optimal policy.

   (a) (2 points) Why would you consider the policy gradient approach to be better than the value function based approach?

Solution: Since, we don't know the functions for optimal policies in both the cases, we can use policy gradient. In Policy Gradients, there is a chance to learn sub optimal policies with the help of parameterization. If the parameter space is big or chosen non linearly, then Policy Gradients can learn optimal policies too.

(b) (2 points) Under what circumstances would the value function based approach be better than the policy gradient approach?

Solution: If the value function based method is able to learn the optimal policy, by the approximation of state values. Then, we can use them over policy gradients as we will get the optimal policies from the values of the states.

(c) (2 points) Is there some circumstance under which either of the method can find the optimal policy?

Solution: No, not possible in this setting as the function approximator cannot represent the optimal policy.

3. (4 points) Answer the following questions with respect to the DQN algorithm:

- (2 points) When using one-step TD backup, the TD target is $R_{t+1} + \gamma V(S_{t+1}, \theta)$ and the update to the neural network parameter is as follows:

$$\Delta\theta = \alpha(R_{t+1} + \gamma V(S_{t+1}, \theta) - V(S_t, \theta))\nabla_\theta V(S_t, \theta) \tag{1}$$

  Is the update correct ? Is any term missing ? Justify your answer

Solution: No, the update is correct.

- (2 points) Describe the two ways discussed in class to update the parameters of target network. Which one is better and why?

Solution: The two updates are soft updates and full updates. The type of update to choose depends on the setting.

4. (4 points) Experience replay is vital for stable training of DQN.

(a) (2 points) What is the role of the experience replay in DQN?

Solution:

- To keep track of the state, action and reward tuple.
- Breaks correlation between successive samples.
- The data required to perform Q Learning comes from this replay buffer.

(b) (2 points) Consequent works in literature sample transitions from the experience replay, in proportion to the TD-error. Hence, instead of sampling transitions using a uniform-random strategy, higher TD-error transitions are sampled at a higher frequency. Why would such a modification help?

Solution: Sampling at uniform random strategy is costly as the replay memory is mostly so huge and the cumulative rewards are unstable.

5. (3 points) We discussed two different motivations for actor-critic algorithms: the original motivation was as an extension of reinforcement comparison, and the modern motivation is as a variance reduction mechanism for policy gradient algorithms. Why is the original version of actor-critic not a policy gradient method?

Solution: In the original paper, the features and the control policies had to be changed for the actor updates. Hence, the updates had to be made in time scale.

6. (4 points) 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: These safe- state conditions are necessary,

- Leaf irrelevance
- Max Node Irrelevance

7. (3 points) Consider the problem of solving continuous control tasks using Deep Reinforcement Learning.

(a) (2 points) Why can simple discretization of the action space not be used to solve the problem? In which exact step of the DQN training algorithm is there a problem and why?

Solution: If we discretize the continuous action spaces, then the memory required to hold the actions, will shoot up.i.e. for an action space with M dimensions, discretizing K atomic actions per dimension leads to M^K combinations of joint atomic actions. The problem will arise in DQN when we trying to pick the argmax action.

(b) (1 point) How is exploration ensured in the DDPG algorithm?

Solution: The DDPG uses distribution over actions, using which we can sample and perform explorations.

8. (3 points) Option discovery has entailed using heuristics, the most popular of which is to identify bottlenecks. Justify why bottlenecks are useful sub-goals. Describe scenarios in which a such a heuristic could fail.

Solution: Bottlenecks are useful in getting temporal abstractions by experience. These bottlenecks can be used as sub-goals. This will create a platform for greater accessibility to a larger number of states within a well connected region allowing an agent to effectively explore within that region.