Sumeet Dinesh Shanbhag

Reinforcement Learning Assignment 7 DRL

1. **What are the two sources of error in Deep RL with function approximation?**

   The two main sources of error in Deep Reinforcement Learning (RL) with function approximation are:

   a) **Approximation Error:** This error occurs when the function approximator (like a neural network) is unable to perfectly represent the true value function. In deep RL, the value function or the policy is approximated using deep neural networks, and these approximations are not always perfect due to the limitations in the network's architecture, the finiteness of the training data, or the complexity of the function being approximated.

   b) **Sampling Error:** This type of error occurs due to the variability in the samples collected from the environment. Deep RL algorithms learn from experiences sampled from the environment, and if these samples are not representative of the entire state-action space, or if the sampling process is biased, it can lead to inaccuracies in learning. Sampling error can affect the generalization of the learned policy or value function to new or unseen states and scenarios.

2. **In TD learning with a neural network what are we trying to minimize? What are we trying to maximize?**

   In Temporal Difference (TD) learning with a neural network, the primary goals are:

   a) **Minimize:** The objective is to minimize the temporal difference error. This error is the difference between the predicted value of the current state (as estimated by the neural network) and the sum of the reward received plus the predicted value of the next state, according to the current policy. The minimization of this error leads to a more accurate estimation of the state-value function or the action-value function.

   b) **Maximize:** While the direct objective of TD learning is not to maximize something, the indirect goal is to maximize the accuracy and efficiency of the value function approximation. By minimizing the TD error, the learning algorithm effectively maximizes the alignment of the neural network's predictions with the actual rewards and transitions observed in the environment, which in turn leads to better policy decisions.

3. **What is the main benefit of deep neural networks in function approximation compared to linear method? What is the linear function approximation useful for?**

   The main benefit of using deep neural networks for function approximation in reinforcement learning is their ability to handle complex, high-dimensional input spaces. Deep neural networks are particularly adept at learning non-linear relationships and patterns in the data, which is often required in complex RL environments. They can effectively approximate complex value functions and policies, which might be infeasible with simpler, linear models.

   Linear function approximation, on the other hand, is useful for its simplicity, interpretability, and computational efficiency. It works well in environments where the value function or policy can

be adequately represented with linear relationships. Linear models are often easier to analyse and less prone to overfitting compared to deep neural networks, making them suitable for problems with lower-dimensional input spaces or where interpretability is a key requirement.

4. **In DQN, what is the purpose of the target network and value network?**

   a) **Purpose of the Target Network in DQN:** The target network in DQN is used to provide stable target values for the Q-learning updates. In DQN, the goal is to learn an optimal policy by learning the Q-values, which are estimates of the expected rewards for taking a given action in a given state and following the optimal policy thereafter. However, using a single network for both selecting actions and generating target values can lead to instability and divergence in the learning process. The target network, which is a copy of the main network but with its weights frozen for a certain number of steps, provides fixed Q-value targets for the updates. This separation helps in stabilizing the training process by reducing the correlation between the target and the estimated Q-values.

   b) **Purpose of the Value Network in DQN:** The value network, also known as the main network or the online network, is actively updated at each step or after a certain number of steps. Its purpose is to estimate the Q-values, which are used both in selecting actions (according to an epsilon-greedy policy for exploration and exploitation) and in calculating the TD (Temporal Difference) error for network updates. The value network learns to approximate the optimal Q-function from the experiences (state, action, reward, next state) it observes.

5. **In the Deep Q-learning method, which are true:**
   a. epsilon-greedy is required to ensure exploration.
   b. exploration this taken care of by the randomization provided by experience replay?

   a) **Epsilon-greedy is required to ensure exploration:** True. The epsilon-greedy strategy is a critical component in Deep Q-learning (and many other RL algorithms) to balance exploration and exploitation. This strategy involves choosing a random action with probability epsilon (exploration) and the best-known action with probability 1-epsilon (exploitation). This ensures that the agent does not solely rely on its current knowledge but also explores the environment to discover potentially better strategies.

   b) **Exploration is taken care of by the randomization provided by experience replay:** This statement is somewhat misleading. While experience replay, which involves storing and randomly sampling past experiences (state, action, reward, next state) to update the network, does contribute to breaking the correlation between consecutive learning updates, it does not fully replace the need for exploration strategies like epsilon-greedy. Experience replay helps in stabilizing the learning process and improving sample efficiency, but on its own, it is not sufficient to ensure adequate exploration of the state-action space. Therefore, both epsilon-greedy strategy and experience replay play distinct and important roles in the Deep Q-learning method.

6. **Value function-based RL methods are oriented towards finding deterministic policies whereas policy search methods are geared towards finding stochastic policies**
   a. True
   b. False

   True. Value function-based RL methods, such as Q-learning and Deep Q-Networks, are typically oriented towards finding deterministic policies. In these methods, the decision-making process

often involves choosing the action that maximizes the estimated value function or Q-value, leading to a deterministic selection of actions given a state.

On the other hand, policy search methods, which directly parameterize and optimize the policy, are more naturally suited to finding stochastic policies. These methods, like policy gradient methods, often optimize a parameterized policy that can express probabilities over actions, allowing the policy to specify a distribution over actions rather than a single best action. This approach is inherently more flexible and can be advantageous in environments where stochastic policies are beneficial, such as those with multi-modal action values or in scenarios requiring exploration.

7. **What makes the optimization space smooth in policy gradient methods?**

   In policy gradient methods, the smoothness of the optimization space is primarily due to the way these methods update the policy parameters. Specifically:

   a) **Gradual Policy Updates:** Policy gradient methods update the policy parameters in a way that makes small, incremental changes to the policy. This is achieved by taking steps proportional to the gradient of the expected reward with respect to the policy parameters. Because these updates are gradual and proportional, they tend to avoid drastic changes in the policy, thereby maintaining a smooth optimization landscape.

   b) **Differentiable Policy Representation:** Policy gradient methods typically use a differentiable function (such as a neural network) to represent the policy. This differentiability ensures that small changes in the policy parameters lead to small changes in the policy itself. This continuity in the policy space contributes to the smoothness of the optimization landscape.

   c) **Exploration-Exploitation Balance:** Policy gradient methods inherently maintain a balance between exploration and exploitation, often through the use of stochastic policies. This balance ensures that the policy does not converge too quickly to suboptimal actions, allowing for a smoother exploration of the policy space.

8. **How does actor-critic architecture improve the “vanila” policy gradient?**

   a) **Variance Reduction:** One of the key improvements of actor-critic methods over vanilla policy gradient methods is the reduction in variance. The critic in the actor-critic architecture estimates the value function, which is used to compute a more accurate estimate of the advantage function. This reduces the variance of the policy gradient estimate compared to vanilla policy gradient methods, which typically rely on full episode returns or Monte Carlo methods for estimating the advantage.

   b) **Efficient Learning:** Actor-critic methods allow for more efficient learning by updating the policy (actor) and the value function (critic) simultaneously. This dual approach can lead to faster convergence as the critic provides a learned estimate of the value function, which can guide the policy updates more effectively than relying on empirical returns.

   c) **Stability and Convergence:** The use of a critic for estimating the value function helps stabilize the training process. The actor-critic method tends to have better convergence properties compared to vanilla policy gradient methods, which can suffer from high variability in policy updates due to their reliance on full trajectory returns.

d) **Applicability to Continuous Action Spaces:** Actor-critic methods are well-suited for problems with continuous action spaces. The actor can output a continuous action directly, while the critic evaluates the quality of these actions, a setup that is more challenging to implement with vanilla policy gradient methods.

e) **Ability to Utilize Bootstrapping:** Unlike vanilla policy gradient methods that typically rely on full trajectory rollouts, actor-critic methods can leverage bootstrapping, using estimates of future returns from the critic. This can lead to more efficient learning, especially in environments with long episodes or continuous tasks.

9. **What is the Advantage function, $A_\pi(s_t,a_t)$, in actor-critic architecture? Write down the equation for monte-carlo-based Advantage function and TD-based advantage function and comment on the pro and cons of each on.**

Monte-Carlo-based Advantage function

$$A(s,a) = \sum_{k=t+1}^{T} r(s_k^i, a_k^i) - V_\pi(s_t^i)$$

Pros: Provides an unbiased estimate of the advantage. Does not require a model of the environment.

Cons: Can have high variance, especially in environments with long episodes or high variability in returns. Inefficient in terms of sample usage as it requires full episode rollouts for each estimate.

TD-based advantage function

$$A_\pi(s_t, a_t) = R_{t+1} + \gamma V_\pi(s_{t+1}) - V_\pi(s_t)$$

Pros: Reduces variance compared to the Monte-Carlo method. More sample-efficient as it does not require full episode rollouts.

Cons: Introduces bias due to the use of value function estimates. Requires a model for the prediction of next state values unless bootstrapping from observed transitions.

10. **Can you find a resemblance between actor-critic architecture and generalized policy iteration in chapter 4?**

Yes, there is a resemblance between the actor-critic architecture and generalized policy iteration. Generalized policy iteration (GPI) is a fundamental concept in reinforcement learning that involves two interacting processes: policy evaluation, where the value of a policy is assessed, and policy improvement, where the policy is modified to perform better based on the evaluation.

Actor-critic methods mirror this structure:

- **Actor**: Represents the policy improvement part of GPI. It makes decisions (selects actions) based on the current policy.
- **Critic**: Corresponds to the policy evaluation part of GPI. It assesses the quality of the actions taken by the actor by estimating the value function.

11. **In the actor-critic architecture described in the lecture, assuming the use of differentiable function approximators, we can conclude that the use of such a scheme will result in:**
a) convergence to a globally optimal policy
b) convergence to a locally optimal policy
c) cannot comment on the convergence of such an algorithm.

In the actor-critic architecture described, assuming the use of differentiable function approximators, we can conclude that the use of such a scheme will result in:

a) **Convergence to a globally optimal policy: This is not guaranteed.** While differentiable function approximators allow for gradient-based optimization, finding a global optimum in complex, high-dimensional spaces where local optima may exist is not assured.

b) **Convergence to a locally optimal policy: This is more likely.** Differentiable function approximators, through gradient descent, are typically capable of finding local optima. In the context of actor-critic methods, this would correspond to a policy that is locally optimal given the current value function approximation.

c) **Cannot comment on the convergence of such an algorithm: It's difficult to make a definitive comment on the convergence without additional context**, such as the specific details of the function approximator, the learning rates, and the environment. However, generally, actor-critic methods with differentiable function approximators are designed to converge to at least a local optimum under certain conditions.

12. **Assume that we are using the linear function approximation for the critic network and SoftMax policy with linear function approximation for the actor network. Write down the parameter update equations for the actor and the critic network (use on-line update equations similar to the algorithm in page 332)**

Actor Network Parameter Update Equation and Critic Network Parameter Update Equation:

$$\begin{aligned}
\delta_t &= R_{t+1} + \gamma \hat{v}(S_{t+1},\mathbf{w}) - \hat{v}(S_t,\mathbf{w}),\\
\mathbf{z}_t^{\mathbf{w}} &= \lambda^{\mathbf{w}} \mathbf{z}_{t-1}^{\mathbf{w}} + \nabla \hat{v}(S_t,\mathbf{w}),\\
\mathbf{z}_t^{\boldsymbol{\theta}} &= \lambda^{\boldsymbol{\theta}} \mathbf{z}_{t-1}^{\boldsymbol{\theta}} + \nabla \ln \pi(A_t|S_t,\boldsymbol{\theta}),\\
\mathbf{w} &\leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta_t \mathbf{z}_t^{\mathbf{w}},\\
\boldsymbol{\theta} &\leftarrow \boldsymbol{\theta} + \alpha^{\boldsymbol{\theta}} \delta \mathbf{z}_t^{\boldsymbol{\theta}},
\end{aligned}$$

13. **Consider a trajectory rollout of (s, a_1, s, a_2, s, a_3). The initial policy depicted in the figure below for state s. The horizontal line also shows the value of the Advantage function for each (state, action) pair. After applying the policy gradient update, what do you expect the probability of each action to change to?**

Option b)