RBE 595 — Reinforcement Learning
Deep Reinforcement Learning Assignment

Arjan Gupta

Problem 1

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

Answer

The two sources of error in Deep RL with function approximation are as follows:

- **Bootstrapping error** — This is the error that arises due to the use of bootstrapping. Bootstrapping is the process of using the value of a successor state to update the value of a state. This is done in TD methods. The error is the difference between the target value and the current estimate value.
- **Approximation error** — This is the error defined as the difference between the true value function and the approximate value function. This error arises due to the use of function approximation itself.

Problem 2

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

Answer

In general with any deep TD learning method, we aim to minimize the error between the target value and the current estimate Q-value. We try to maximize the value of the current state by choosing the action that maximizes the value of the next state.

We can use the example of gradient Q-learning to answer this question more specifically. In gradient Q-learning, we are trying to minimize the error given by the loss function as follows:

$$e(w) = \frac{1}{2}\left[Q_w(s,a) - \left(r + \gamma \max_{a'} Q_{w'}(s',a')\right)\right]^2$$

We are trying to maximize the value of the current state by choosing the action that maximizes the value of the next state. This is done by updating the weights of the neural network.

Problem 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?

Answer

Deep neural networks (DNNs) have helped reinforcement learning become more powerful and practical. DNNs can be used to approximate the value function as a non-linear function of the state. This is more powerful than linear function approximation, which can only approximate the value function as a linear function of the state. A general advantage of DNNs is also that they learn their own features, which is not the case with linear function approximation.

Linear function approximation are still useful, however. Their advantage is that they can be used to verify theoretical results (in academia, for example). The mathematical analysis of linear function approximation is much easier than that of non-linear function approximation, which is why linear function approximation is still useful.

Problem 4

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

Answer

In DQN, two separate neural networks are used: the target network and the value network. The target network is used to estimate the target value, whereas the value network is used to estimate the Q-value. The purpose of having two separate networks is to serve as one of the two features of DQN that mitigates the problem of divergence (or weaker convergence) that occurs in gradient Q-learning. The other feature is experience replay.

We can provide more context on how DQN achieves this mitigation, as follows. The target network is updated less frequently than the value network for each mini-batch of (s, a, r, s') tuples from the experience replay buffer. This is done to make the target network more stable. The target network is updated as follows:

$$w_{i+1} = w_i + \alpha \left[Q_w(s,a) - r - \gamma \max_{a'} Q_{\overline{w}}(s', a') \right] \frac{\partial Q_w(s,a)}{\partial w}$$

Where the first Q in the square brackets is the value network, and the second Q is the target network. The second Q remains ‘fixed’ for a certain number of iterations. This is akin to the optimal value function being fixed while the value function for all the other states is being updated.

Problem 5

Can we consider TD an online (real-time) method and MC an offline method? Why?

Answer

Yes, we can consider TD an online (real-time) method and MC an offline method. This is because TD learns during the episode, whereas MC learns after the episode has ended. Specifically, TD updates the value of a state based on the value of the next state (during the episode), whereas MC updates the value of a state based on successive returns (after the episode has ended).

Problem 6

Does Q-learning learn the outcome of exploratory actions? (Refer to the Cliff walking example).

Answer

Yes, Q-learning learns the outcome of exploratory actions. This is because Q-learning is an off-policy TD control algorithm. In the context of the Cliff walking example, this means that Q-learning learns the optimal action-value function, q_*, which is closest to the cliff. The behavior policy takes exploratory actions, and the target policy is greedy. This causes makes it so that initially, the agent falls off the cliff a lot, but eventually, the agent learns to avoid the cliff and converges to the optimal action-value function, q_*. This is why the graph of the sum of rewards per episode for Q-learning is initially very low, but eventually converges to a high value. The graph is shown below:

Problem 7

What is the advantage of Double Q-learning over Q-learning?

Answer

The advantage of Double Q-learning over Q-learning is that Double Q-learning is less prone to bias than Q-learning. Specifically, Q-learning is biased toward the maximum value action, which is also known as maximization bias. This is because Q-learning uses the maximum value action to update the value of a state.

In contrast, Double Q-learning is not biased toward the maximum value action. This is because Double Q-learning uses two action-value functions, Q_1 and Q_2, to update the value of a state. With probability 0.5, Double Q-learning uses either Q_1 or Q_2 to update the value of a state. Now, for example, the action is picked as per Q_1, then that action is probably not the maximum value action as per Q_2. This way, for a given state, we have two estimates of the value of the maximum value action. This reduces the bias of Double Q-learning and is hence the advantage over Q-learning.