Reinforcement Learning

Season of Code, 2021

By

Devakar Pradhan (203310007)

Q-learning

Goal of the Q-Learning algorithm is to learn the Q-Value for a new environment. The Q-Value is the maximum expected reward an agent can reach by taking a given action A from the state S. After an agent has learned the Q-value of each state-action pair, the agent at state S maximizes its expected reward by choosing the action A with the highest expected reward.

The Q-values are stored in a simple list data structure called Q-table. It is used to keep track of the states, actions, and their expected rewards. At the start of the Q-Learning algorithm, the Q-table is initialized to all zeros indicating that the agent doesn't know anything about the world. As the agent tries out different actions at different states through trial and error, the agent learns each state-action pair's expected reward and updates the Q-table with the new Q-value.

Using trial and error to learn about the world is called exploration**.** Explicitly choosing the best known action at a state is called exploitation.

Choosing an action

1. **Epsilon greedy**

   At the beginning of the algorithm, every step the agent takes will be random which is useful to help the agent learn about the environment it's in. As the agent takes more and more steps, the value of epsilon decreases and the agent starts to try existing known good actions more and more. Near the end of the training process, the agent will be exploring much less and exploiting much more.

2. Softmax

Drawback of epsilon-greedy action selection is that when it explores it chooses equally among all actions. This means that it is as likely to choose the worst-appearing action as it is to choose the next-to-best action. In tasks where the worst actions are very bad, this may be unsatisfactory. The obvious solution is to vary the action probabilities as a graded function of estimated value. The greedy action is still given the highest selection probability, but all the others are ranked and weighted according to their value estimates. These are called softmax action selection rules.

Bellman equation

The agent updates the current perceived value with the estimated optimal future reward which assumes that the agent takes the best current known action. In an implementation, the agent will search through all the actions for a particular state and choose the state-action pair with the highest corresponding Q-value.

Deep Q-Network

Rather than mapping a state-action pair to a q-value, a neural network maps input states to (action, Q-value) pairs. It replaces the regular Q-table with a neural network.

The learning process deep Q-learning uses 2 neural networks. These networks have the same architecture but different weights. Every N steps, the weights from the main network are copied to the target network. Using both of these networks leads to more stability in the learning process and helps the algorithm to learn more effectively.

The main and target neural networks map input states to an (action, q-value) pair. Each output node represents an action. Each of these output nodes contain the action's q-value as a floating point number. The output nodes do not represent a probability distribution so they will not add up to 1.