RL: Lab Actor Critic - Write Up

October 2021

Algorithm

Actor-Critic algorithms have two components, the actor who takes actions in the world based on its observations. The critic who observes the action the actor took and determines the value of that action - in the case of Q-value Actor-Critic algorithms this value is used to determine the Q-value of the state given that action. In our case we used the Q-value Actor-Critic algorithm. There are other approaches which modify the loss function (as represented by ∇J), for example TD (temporal difference) actor critic and advantage Actor-Critic. In essence, both the actor and the critic aim to approximate functions. A common example for Actor-Critic algorithms is a mother and their child. The child (our actor) takes a variety of actions for example running on ice, cycling with no hands and eating their vegetables. The mother observes the child's action in each given state and informs the child whether it was a good choice or bad choice. Following this, the child will adjust accordingly and act better over time according to what the mother recommended. This is done via the use of two neural networks, where one is used as the actor and it approximates the optimal action given an observation. The other network is used to approximate the value for the action in the given state. This is done with the aim to reach the global maximum Q-value - which approximates future rewards.

Actor-Critic Network Architecture

Typically the architecture of the Actor-Critic algorithm has the actor and the critic following very similar network setups. The actor and the critic both take in an observation of the environment as input to the neural network. They both then forward propagate the input through the network - which is often a deep neural network. The result differs for actor and critic in terms of the output of their networks. Firstly, the critic gives an estimated value of the state based on the observed state of the world (our input). The actor gives a policy of actions to take - i.e the probability of taking each action in the current state. Therefore, the actor's last layer in the neural network must be a softmax activation, or another activation that allows for a probability distribution amongst possible actions to be generated. In updating the neural network, the loss function is

modified to take into account what the agent perceived the value of the state was (as given by the critic). This in essence allows the actor to use the critic as a more intelligent baseline - if you compare the algorithm to REINFORCE, this would be the fundamental difference.