Jack Burke

Jpb2uj

Applied Reinforcement Learning – Journal

# Module 1 – RL Fundamentals

In this module, we reviewed the fundamentals of reinforcement learning, including how it differs from other fields in machine learning. My biggest takeaway was the interaction between the agent and the environment – this is the fundamental difference between reinforcement learning and supervised/unsupervised learning. With that in mind, defining the reward that the agent receives for its various interactions with the environment seems to be the key piece that makes a model valuable. This also intuitively makes it difficult to build an agent for a driverless car, for example, as the reward function has to define something like staying on the road, \*not\* getting into an accident, and keeping all the passengers comfortable – which is more complicated than building an agent to win a game of checkers, where every action contributes to the desired outcome of defeating the opponent.

I think it may be challenging in many applications to keep the agent from taking the greedy choice every time, thus reducing the long-term value. I am very excited to learn the strategies for achieving this, as it seems like such a human thing to do – making a plan to achieve a goal in the long term by sacrificing short-term rewards.

# Module 2 – K-Armed Bandit

We learned about the k-armed bandit problem, in which an agent tries to maximize the total reward over time by choosing which arm to pull given k slot machines. The reward from each arm is not known, and each time you pull an arm you receive a sample reward from that machine’s reward distribution.

I found the example showing how a greedy approach to this problem to be enlightening, as always taking a good reward, but not necessarily the optimal reward, is an intuitive place for an agent to get stuck. The epsilon-greedy policy provides a good balance between exploitation and exploration and seems quite simple to implement conceptually. This answered my questions from module 1, where I was curious how to prevent the agent from taking the greedy choice every time. I would say the epsilon-greedy policy even describes how I play board games – even for a game I know well and can probably choose the optimal move, I sometimes take a different action to “explore” and see if I can improve on my current best strategy.

# Module 3 – Markov Decision Processes

There was a lot of reading for this module and several new concepts were introduced. We learned about dynamic programming first, which requires knowing the transition probabilities of each state and the rewards. It compares either value functions or policies recursively using the Bellman equation, which seems to be critical for reinforcement learning in general. Dynamic programming is limited because it needs a full model of the environment.

Monte Carlo methods differ in that they work only at the end of episodes, meaning they don’t need the environmental model but may struggle with continuous environments. This method also averages the returns from a state to the end of the episode without finding it exactly.

Temporal difference learning seems to take the best of both methods, as it updates estimates before the estimate ends, improving on MC, and uses bootstrapping like DP to update values based on current estimates for each time step. I think this method will be very important moving forward, as it seems to be quite flexible for many different reinforcement learning problems.

I’m hoping to review the difference between on-policy and off-policy learning in more detail – I don’t quite have a handle on it yet, and I don’t think I can describe which situations use each one.

# Module 4 – Q-Learning

This module introduced Q-learning, which is a model-free reinforcement learning algorithm where the agent learns a Q-function to estimate the expected future reward for an action in a given state. Q-learning incorporates the temporal difference error that we covered with temporal difference learning. Q-learning is an off-policy method, so the optimal policy is learned independently of the behavior policy.

I found the cliff walk example instructive, and I enjoyed the lab assignment. It was interesting to see how the agent would fail often in the beginning, as the optimal policy was being discovered, then fail very rarely once it learned the correct policy.

# Module 5

We learned about Deep Q-Networks, which extend the q-network framework by applying deep learning to estimate the Q-function. The reading assignment for the module was a paper introducing a deep q-network used to play Atari 2600 games, which learned from video input and outperformed human experts on several games. The paper also introduced experience replay, where the agent can learn from past experiences by selecting one from its “replay memory” at random.

I thought this paper was one of the coolest experiments I’ve read about. Creating an agent that can outperform humans on not just one game, but generalize to beat humans on multiple games, is impressive. I found it fascinating that the researchers also trained the network with the visual input from the game itself, rather than transcribing the space into something simpler.

# Module 6 – Extending Deep Q-Networks

We learned about some improvements to the basic Q-Network architecture that have been published recently, including prioritized replay and double deep q-networks. Prioritized replay deals with getting the agent to replay experiences with high learning value to speed up learning and improve performance. Double q-learning decouples the action selection and evaluation into two separate networks, thus reducing the overestimation bias that a single deep q-network tends to have.

I really enjoyed the reading assignment for this module – reading the published papers seemed more real than the textbook, and the experimental examples in the paper helped with my understanding of the underlying concepts. I found the prioritized experience replay especially fascinating, partly because the concept is so intuitive – it makes sense to have the agent replay experiences where it performs poorly, or expects a different outcome than what occurs. The agent should spend more time on these experiences, because they are by definition not well understood by the agent.