Inf368 – project 5 – Problems

For this project we implemented function approximation methods and actor critic methods to address the problem of monitoring a conveyor belt with a robot tasked with retrieving cookies. The objective is to maximize reward while avoiding collisions and not using too much time. The task is naturally described as continuing since there is no predetermined end state. The environment continually generates cookies, and the robot's objective is ongoing. It could also be made to be somewhat episodic. Each episode starts when the robot begins patrolling the floor, and it ends after a certain number of time steps, or a number of cookies are retrieved or discarded. Given a finite time horizon and well-defined terminal states, it could be suitable for episodic reinforcement learning. We ended up going with this approach. Where after a certain number of steps, and no cookies being found to have the agent reset.

Feature engineering techniques were employed to preprocess the state representation. Features such as the position and velocity of the robot, the position and remaining time of the nearest cookie, and the distance to the nearest wall, the direction of the cookie as well as their relative distance were all added to represent the state. To avoid exploding gradients and weights these values were normalized to be between -1 and 1. In addition to prevent the agent from standing still or just staying in a corner, we also added some punishment and reward as a feature in the processed state.

4

In the first solution, we employ a linear approximation to approximate $q\pi(a,s)$ and solve the control problem using Q-learning. In the beginning the agent simply learned that standing still was better or avoiding the cookie was better to prevent the high velocity collision punishment. However, after implementing the somewhat episodic approach and the feature engineering described above the agent became rather good. Learning to go after the cookie and trying to do so fast whilst somewhat controlling the velocity. It still makes errors and collides, but in general does accumulate quite a few cookies, whilst trying to minimize the punishments.

In the second solution, we utilized a neural network to approximate $q\pi(a,s)$ and solve the control problem using Q-learning. Similar to the linear approximation solution, we employed feature engineering to preprocess the state and enhance the neural network's learning capabilities. This solution should be more fitted to understand the nonlinear and more complex patterns in the data and should therefore perform better than the simple function approximation solution. This solution contrasted the previous one, by not always running after the cookie, but rather standing still a lot, or moved at a very slow pace, often times not getting the cookie in the 5 second window.

When looking at the plot below we see the cumulative reward of the different solutions learned policy over 500 episodes. The orange represents the Q-learning with the neural network and blue represents the Q-learning with linear approximation. As we can see the Neural Network cumulative rewards is quite higher than the one of the linear approximations. However, there's also many stagnant times, with the last 200 episodes decreasing 0.5 reward for every 25 episodes. Indicating it didn't manage to get the cookie often at all in that period. The blue line in contrast, is way below the orange one, indicating the linear solution is worse. However, the total rewards over the time greatly vary with it often times getting to the cookie before the timer, resulting in positive rewards. But also, often times colliding with the cookie or walls as it has too high a velocity, resulting in some negative rewards as well.

Simply looking at the graph one could indeed believe the neural network did way better than the function approximation, but really it learned to minimize risk rather than go after the cookie in a

controlled pace. Looking at further features or balancing the exploration with various epsilon and decaying variables, using better learning rates, number of episodes and discount factor as well as the neural network structure itself all could affect the agents training greatly. If we had more time to explore all of these, we assume it would be possible for the neural network solution to learn a better policy that isn't as conservative as it currently is.

6

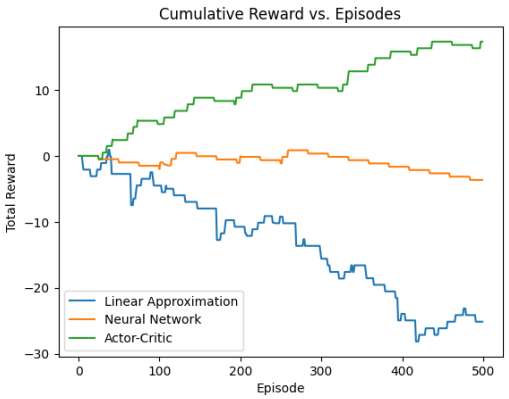
Next, we created an actor-critic model to learn the best policy $\pi(a|s)$. Our model comprises two neural networks: an actor, which determines actions based on state observations, and a critic, which evaluates these actions by estimating their future rewards.

The actor produces action distributions, while the critic assesses state values. Both components are optimized using the Adam optimizer for adaptive learning rates.

A key part of our approach is an ε-greedy strategy with ε-decay, enhancing exploration. This helps maintaining a balance between exploring new actions and exploiting known rewards. As training progresses, ε-decay shifts the balance toward exploitation, allowing focused policy refinement. We originally had an issue with the robot going to the sides or standing still and taking the negative points instead of trying to catch the cookie, this was improved with the use of ε-greedy strategy with ε-decay including changing up the parameters until the robot had a great balance. We used the parameters learning rate = 0.001, start epsilon of 1.0, epsilon min of 0.01 and an epsilon decay of 0.999. We trained over 10000 episodes but tested on only 500 episodes due to time constraints.

The graph shows the accumulation of the total rewards from the three methods: Linear Approximation, Neural Network, and Actor-Critic. Now considering the green line, the Actor-Critic method, it is way above the other two methods, staying at a positive total reward almost from the beginning. This cumulative trend shows the agent mostly gets positive rewards and doesn't as often get negative rewards.

This indicates the agent has learnt to go after the cookie similarly to the linear approximation policy. However, unlike the linear approximation it doesn't accumulate as much negative rewards, in other words the agent has learnt to rarely collide. The agent is also similarly to the Neural Network solution somewhat careful and learnt to rarely collide. However, unlike the Network solution has learnt to try and go after the cookie within the 5 seconds and thus doesn't go as slowly as the Neural network.



In terms of policy development and learning efficiency, the Actor-Critic method seems superior. It not only achieves a higher reward but does so with consistency. This suggests that learning a policy while also estimating the value function—as the Actor-Critic method does—can lead to a more effective and robust approach in environments where a balance between exploration and exploitation is crucial for optimal performance.