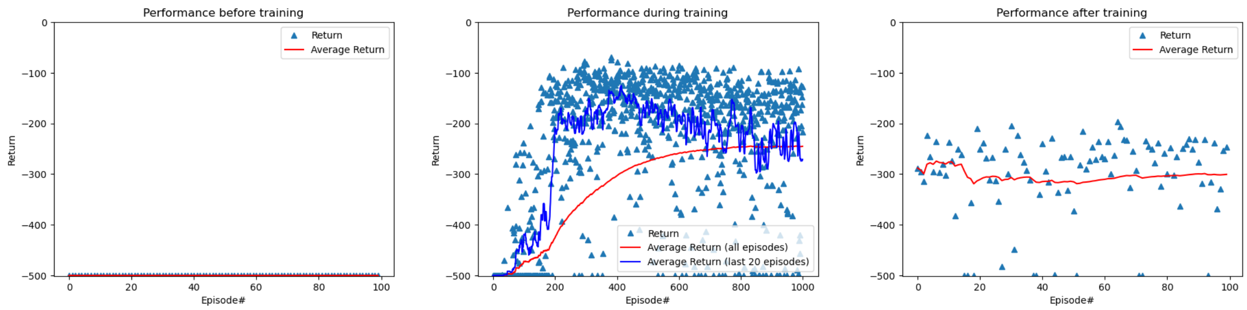
Assignment 3

Kevin McCoy

1. Q Learning with Function Approximation

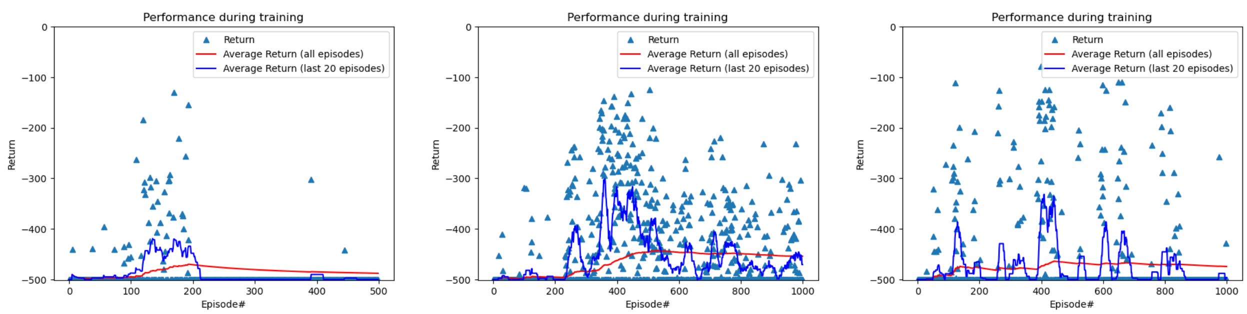
1.1 **30pts**



(a) Average Reward = -500 (b) Training Performance (c) Average Reward = -300.52

Figure 1: Grader Code Output

1.2 **30pts**



(a) Online Learning (b) No Target Networks (c) No Replay Buffer

Figure 2: Grader Code Output

Overall, it is easy to tell that the target network and replay buffer are both critically important to the Q Learning Algorithm. Fully online learning does even worse than missing either one component.

In Figure 3, all models are evaluated on new data. The control / general model does far better than the others.

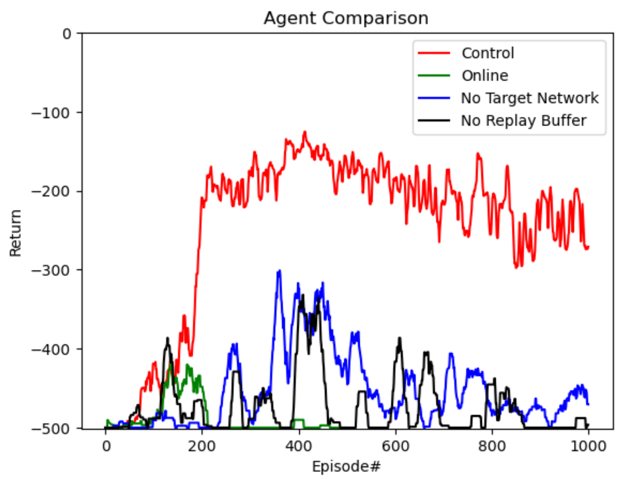


Figure 3: Agent Comparison

1.3 15pts

Figure 4: Agent Comparison With Differing Learning Rate

I chose to vary learning rate. I had originally guessed that the learning rate was not optimal, as my performance seemed to ’bounce around’ too much. Sometimes it would get high returns, but then bounce back to -500. I also chose an ADAM optimizer, which might not need the same learning rate as another optimizer, like SGD.

1.4 10pts

Interestingly, the performances of the off-the-shelf and from-scratch models are about equivalent after enough training. However, the off-the-shelf method gets to that point much quicker. It is worth investigating why this is the case, but I would guess it is because it uses a more flexible MLP architecture than I used.

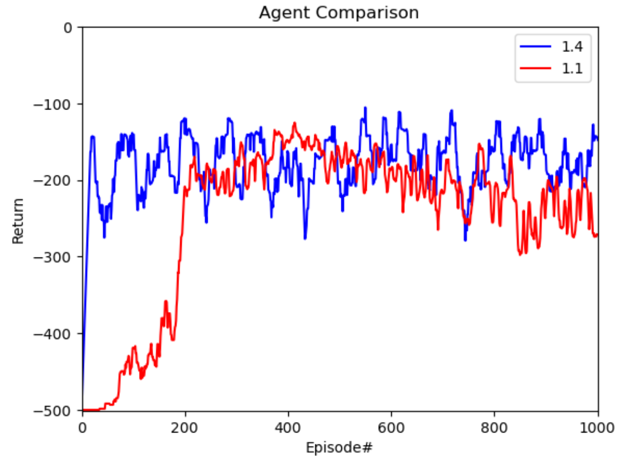


Figure 5: Agent Comparison Between Part 1.1 and 1.4

2. Policy Gradient Theorem

2.1 3pts

Policy gradient methods learn the policy directly, which thus allows for a stochastic policy. In other words, one would not be limited to a epsilon greedy policy based on a value function. Policy gradient also works when the action space is continuous. One example of this is a driverless car. Just taking into consideration wheel control, the agent can choose any wheel angle that the car is capable of. (e.g. $[-720°, 720°]$)

2.2 12pts

We can write:

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \nabla_\theta \upsilon(s) && (1)\\
&= \nabla_\theta \Big[\sum_a \pi(a|s,\theta) q_\pi(s,a)\Big] && (2)\\
&= \sum_a \Big[\nabla_\theta \pi(a|s,\theta) q_\pi(s,a) + \pi(a|s,\theta)\nabla_\theta q_\pi(s,a)\Big] && (3)\\
&= \sum_a \Big[\nabla_\theta \pi(a|s,\theta) q_\pi(s,a) + \pi(a|s,\theta)\nabla_\theta [\sum_{s',r} p(s',r|s,a)(r + \gamma * \upsilon_\pi(s'))]\Big] && (4)\\
&= \sum_a \Big[\nabla_\theta \pi(a|s,\theta) q_\pi(s,a) + \gamma * \pi(a|s,\theta)[\sum_{s'} p(s'|s,a)\nabla_\theta \upsilon_\pi(s')]\Big] && (5)\\
&= \sum_{x\in\mathcal{S}} \sum_{t=0}^{\infty} \gamma^t \mathbb{P}(s \to x, t, \pi) \sum_a \nabla_\theta \pi(a|x,\theta) q_\pi(x,a) && (6)\\
& && (7)
\end{aligned}
$$

Thus,

$$
\begin{aligned}
\nabla_\theta \upsilon(s_0) &= \sum_s \sum_{t=0}^{\infty} \gamma^t \mathbb{P}(s_t = s|s_0,\pi) \sum_a \nabla_\theta \pi(a|x,\theta) q_\pi(x,a) && (8)\\
&= \sum_s d_\pi(s) \sum_a \nabla_\theta \pi(a|s,\theta) q_\pi(s,a) && (9)\\
& && (10)
\end{aligned}
$$

Q.E.D.