# 10-703 Deep RL and Controls Homework 2

Shuguan Yang (shuanguany), Xuan Li (xuanli1)

April 4, 2017

## Summary

In this assignment, we implement all six models based on the papers provided. For the deep Q learning implementations, we tried both the two convolution layers network setup in [1] as well as the three convolution layers setup in [2], it turns out that the network with three conv layers from [2] was too slow to converge in 30 hours when running on psc clusters, so we stick to the two convolution layers network from [1] in model evaluations.
We evaluate our models based on Atari game: Space-Invaders-v0. The duel deep Q model achieves the best performance among all with an average rewards of 353 per game.
Regarding the learning process, we majorly follows the parameters provided in the writeup with minor modifications to speed-up the learning. We reduce the total number of steps to 3M, adopt a burn_in number of 100K, and update the network every 4 steps.
Our code is available on Github repository: `https://github.com/BreadYang/DQN`.
Game videos of final models as well as intermediate models of double deep Q [1] and dueling deep Q [2] are uploaded to our Youtube channel.

*Note*: Due to cluster slow-responding, our jobs had waited in queue for more than 24 hours time to time, so it would be greatly appreciated if we can be granted one or more late days.

## Question 1

Proof: When the feature function $\phi$ is of the form $\phi(s,a)_{s',a'} = \mathbb{1}[s = s', a = a']$, the value of $\phi(s,a)_{s',a'}$ will be deterministic and one-hot at any step since the state and action taken is always known. There is $Q(s,a) = \omega\phi(s,a) = \omega \cdot \mathbb{1}[s = s', a = a'] = \omega$ when $[s = s', a = a']$. Also, $\nabla_\omega Q_\omega(s,a) = \mathbb{1}[s = s', a = a'] = 1$ when $[s = s', a = a']$. Since the update is always executed on the current state-action pair $[s', a']$, the last equals in the above two equations always hold. By replacing the above two terms in Eq(1) and Eq(2), they will be in the same form.

[1] https://www.youtube.com/watch?v=4nWYqqlPbFklist=PLbCp1gNbcKRZEOGmrj31cKBta10jexS6t
[2] https://www.youtube.com/playlist?list=PLbCp1gNbcKRajm0F0cfxjIqJefMP4ozY0

# Q2: linear Q-network

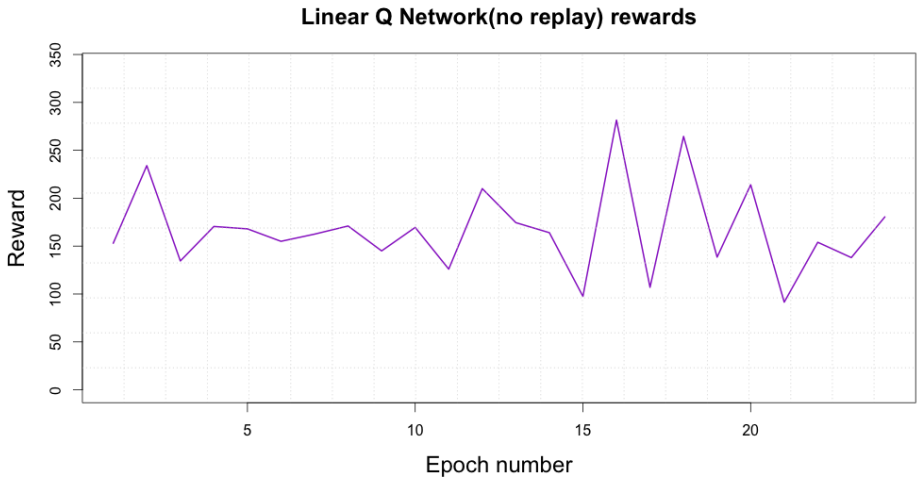


Figure 1: Reward per epoch (10,0000 steps)

# Q3: linear Q with experience replay and target fixing

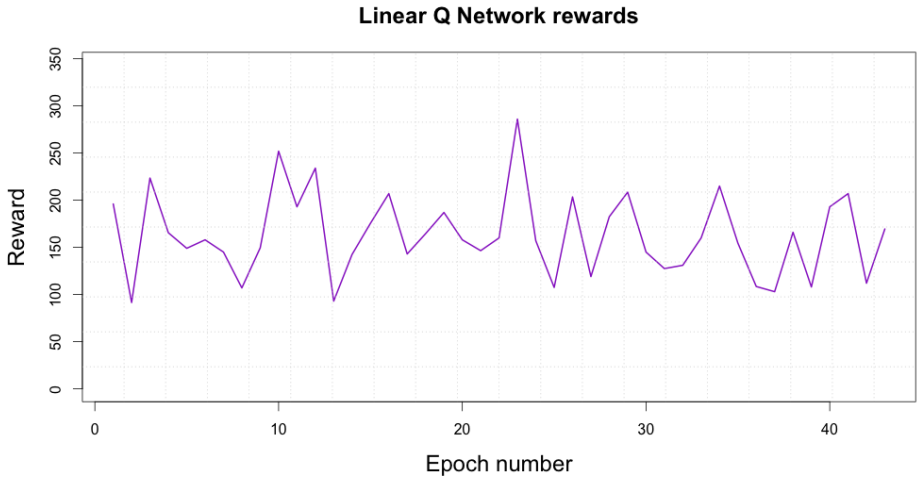


Figure 2: Reward per epoch (10,0000 steps)

# Q4: linear double Q-network

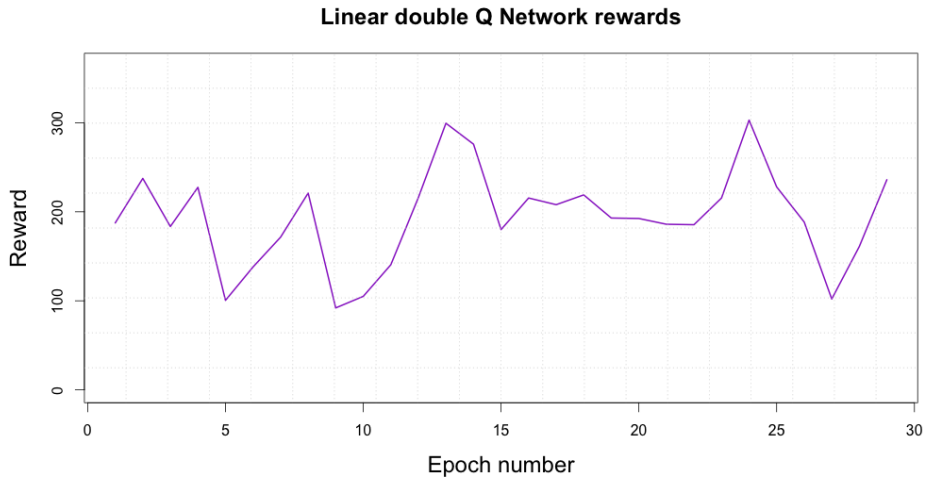


Figure 3: Reward per epoch (10,0000 steps)

# Q5: deep Q-network

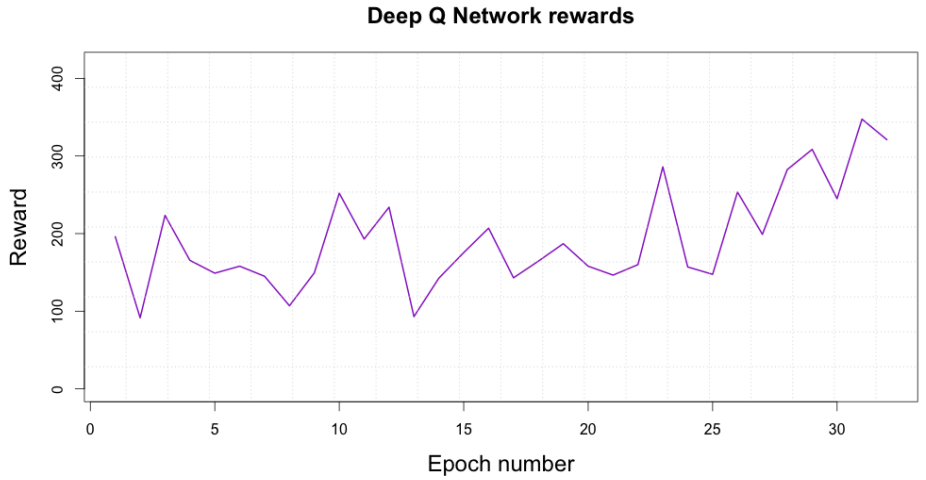


Figure 4: Reward per epoch (10,0000 steps)

# Q6: double deep Q-network

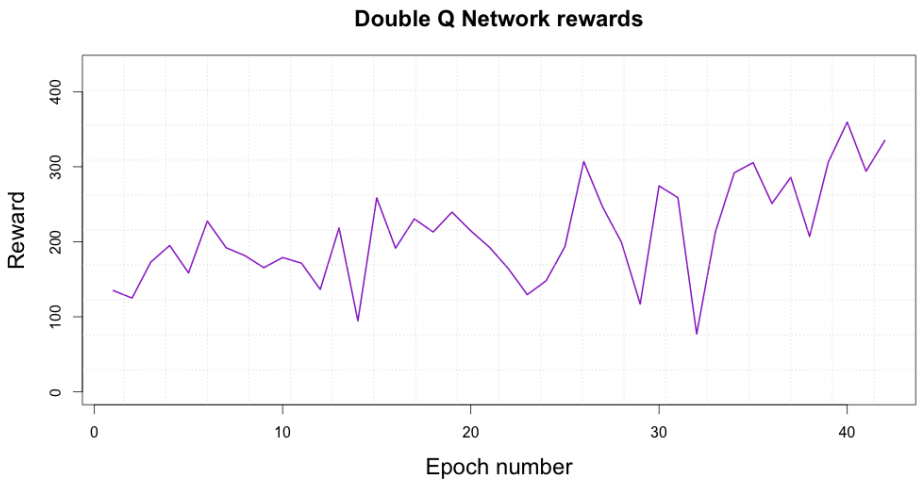


Figure 5: Reward per epoch (10,0000 steps)

# Q7: dueling deep Q-network

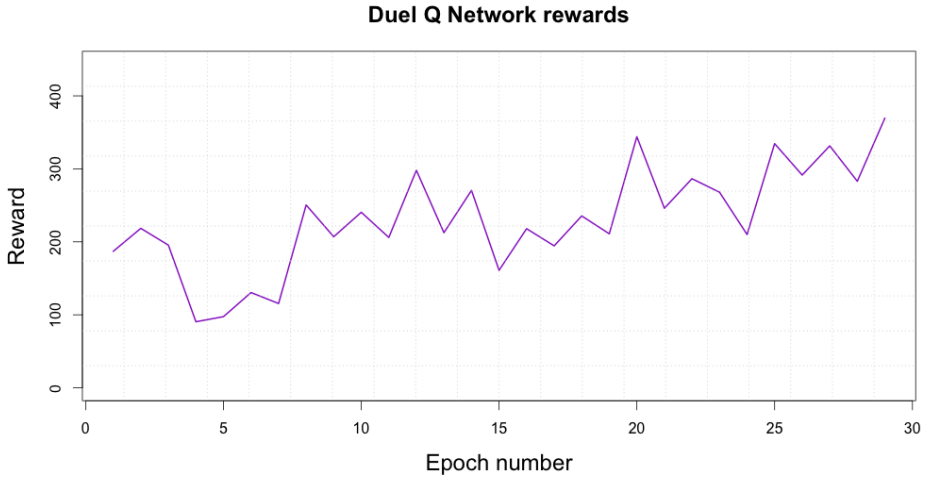


Figure 6: Reward per epoch (10,0000 steps)

# Videos

The videos could be found from the link below:

`https://cmu.box.com/s/2cspy0edndhdc7hz3vhijvvle74exxxf`

Also we have posted a demo of one episode of our best model, dueling deep Q-network, on youtube, which might provide some insights of the learned policy. The link is listed below:

https://www.youtube.com/watch?v=eYC5CyWT4vg

# Performance Table

Below is the average total reward per episode in 100 episodes achieved by our final models. We have found several interesting patterns:

1. In general the 'deep' models outperform the 'linear' ones in terms of mean rewards (scores). However, the standard deviation across these models are similar, which means more training are required to achieve a steady model.

2. For both the 'linear' and 'deep' models, the performance increases with model complexity (no replay $\Rightarrow$ replay $\Rightarrow$ double, deep $\rightarrow$ double deep $\rightarrow$ dueling deep).

3. Together with video analysis, generally 'linear' models would be more likely to generate random policy, while 'deep' models would generate a more deterministic policy to reach high scores. For examples, we have observed that the 'dueling' model tend to shoot enemy in vertical direction first, and in this way the agent would then shoot the pink balls (200 scores). This is because we have used raw rewards in our model, which makes the agent able to recognize high reward actions and make decision in this way.

4. During training stages, all of the models show a upward trends in terms of average reward per 10 episode. However, due to the limitation of computing resources, we are not able to extend our training to more than 5000000 samples, and did not observe a steady or converging pattern of rewards.

| Model | Mean | std |
|---|---|---|
| Linear Q-network (no replay) | 120.83 | 120.37 |
| Linear Q-network | 159.95 | 110.95 |
| Linear double Q-network | 164.01 | 96.05 |
| deep Q-network | 321.25 | 116.82 |
| double deep Q-network | 336.92 | 108.72 |
| dueling deep Q-network | 353.83 | 92.41 |