# CSE 4/546: Reinforcement Learning
# Assignment 2 - Deep Q-Networks

Vignesh Prakash
UBID : 50478782
prakash8@buffalo.edu

Mohit Sai Aravind Nunna
UBID : 50468322
mnunna@buffalo.edu

## 1 Benefits

### 1.1 Using Experience Replay in DQN

Using experience replay helps in sampling efficiency by allowing agent to learn from past experiences. It also helps learning from rare events as these rare events get stored in replay buffer. It also helps in improving stability. Modifying the size of replay buffer can also influence the results of learning process. A large replay buffer allows the agent to store more experiences and learn from more diverse set of experiences, which can improve quality of learned policy. However increasing the replay buffer too much can require more memory and computation time.

### 1.2 Introducing Target Network

1. Improvement in stability: By introducing a target network, the correlation between the target Q-values and the predicted Q-values is reduced, thus stabilizing the training process.
2. Convergence : By using Deep Q Network, the agent can learn to find the optimal Q-values more accurately, thus improving the convergence
3. Reduces Overestimation : As the Q-values of the target network are learned using the same network, the target network helps to reduce overestimation of the Q-values. The rate at which the target network is updated can also affect the performance of the DQN. Updating too often can lead to instability and updating infrequently can slow down the convergence. Thus, the frequency of updating the target network has to be done for every few thousand steps.

### 1.3 Representing the Q Function as $q^{(}s, w)$

1. Flexibility : For large state spaces and complex dynamics, Neural networks can be used as it is very flexible and can accurately learn to represent complex functions
2. Generalization : By using a Neural network, DQN algorithm can map the states to Q-values and thus can generalize its model to new and unseen states

## 2 Environments

### 2.1 Grid Environment

In our project, we have used a basic grid world with the following parameters and objective

#### 2.1.1 Deterministic Environment Parameters

1. **Number of states** : 12
2. **Number of Actions**: 4
3. **Number of Rewards**: 6 (0,1,2,3,4,-2.5)

#### 2.1.2 Stochastic Environment Parameters

1. **Number of states** : 12
2. **Number of Actions**: 4
3. **Number of Rewards**: 6 (0,1,2,3,4,-2.5)

In the stochastic environment, the agent moves to the proper position as per the action with only 0.4 probability, and it moves to the other 3 positions at 0.2 probability.

To illustrate this, If right action is provided to the agent, the agent moves right with a 0.4 probability and it might move to left, top, down with a probability of 0.2

### 2.2 CartPole-v1

CartPole-v1 is one of the Classic Control Environments from the Gymnasium library.

1. The action space of this environments is a discrete set of 2 numbers, indicating the left and the right movement of the cart.
2. The observation space consists of values like Cart Position, Cart Velocity, Pole angle,Pole Angular Velocity.
3. The goal of this model is to stabilize the pole supported in the cart and make it stand upright.
4. The rewards will be +1 as long as the pole stays upright.

### 2.3 LunarLander-v2

LunarLander-v2 is one of the Box2D Environments from the Gymnasium library.

1. The action space of this environments is a discrete set of 4 numbers, indicating the do nothing, fire left orientation engine, fire main engine and fire right orientation engine
2. The observation space is a 8 dimensional vector, containing the coordinates of the lander in x & y, its linear velocities in x & y, its angle, its angular velocity, and two booleans that represent whether each leg is in contact with the ground or not.
3. The goal of this model is to properly land the lander in the landing pad in a upright position without crashing
4. The reward function is based on multiple factors, like the distance of the lander to landing pad, movement speed of the lander, tilt of the lander, contact to the ground and amount of fuel used. The reward of -100 or +100 points is given for crashing or landing safely respectively.

## 3 Results

### 3.1 Hyper Parameters used for Grid Environment

- Memory Size - 500
- Epsilon - 1
- Minimum Epsilon - 0.01
- Epsilon Decay - 0.999
- Learning Rate ($\alpha$) - 0.01
- Discount Factor ($\gamma$) - 0.9
- Batch Size - 16
- No Of Episodes - 10
- Steps per Episode - 15
- Target Network Update Frequency - 5

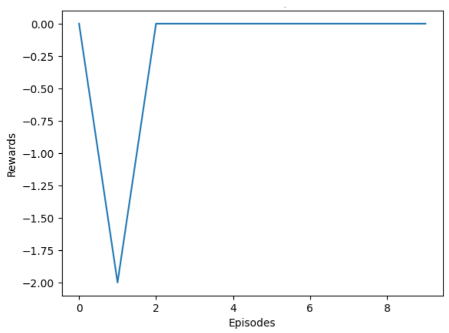

Figure 1: Total Rewards vs Episodes in Grid Environment

### 3.2 Hyper Parameters used for Cartpole-v1 Environment

- Memory Size - 200
- Epsilon - 1
- Minimum Epsilon - 0.01
- Epsilon Decay - 0.999
- Learning Rate ($\alpha$) - 0.01
- Discount Factor ($\gamma$) - 0.9
- Batch Size - 64
- No of Episodes - 30
- Steps per Episodes - 100
- Target Network Update Frequency - 5

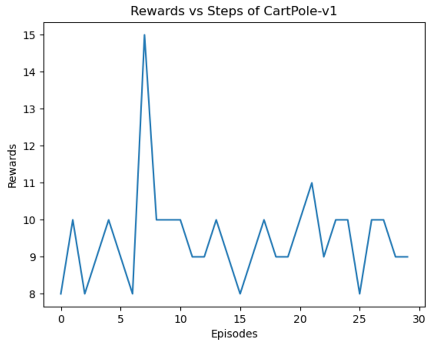

Figure 2: Total Rewards vs Episodes in CartPole-v1

## 4 Interpretion of the Results

### 4.1 Grid Environments

Currently, we have implemented the DQN environment for our Deterministic grid world environment. However, on training the model for 10 episodes with a batch size of 16, learning rate of 0.01 and with 100 steps per episode, we notice that the rewards vs graphs decreases at the 1st epoch but for the other episodes it reaches 0 directly. This is happening because the model is not able to fit properly. We will try to increase the replay buffer so that the model can sample from more unique experiences. We inferred that this happened as our number of steps per epochs × number of epochs is lesser than the batch size, hence there aren't many unique experiences generated.

### 4.2 CartPole-v1

We ran our Cartpole environment for 30 epochs and we got a fluctuating graph of rewards vs steps. It falls down to a minimum of 8 and rises to a maximum of 13. It means the model is able to generalize a pattern but it needs more information and epochs to understand how to balance a cart pole.

## 5 References

Gym Documentation

Human-level control through deep reinforcement learning

## 6 Github Link

UB_CSE546_RL_Assignment_2