6/18/2021

**1/ MDP – Markov Decision Processes:**

* Elements of MDP:

**+** Environment

+ Agent

**+** States of environment

**+** Actions that Agent can take

**+** Rewards that Agents can receive

=> Goals: Agents wants to maximize the cumulative rewards

**Diagram

Description automatically generated**

- Expected Return of the reward of a given time step: It is the agent’s goal to maximize the expected discounted return of rewards

- Policy and Value function:

+ Policy = What is the probability that an agent will select a specific action from a specific state? We say that the agent follow a policy meaning that at time t, under policy pi, the probability of taking action a in stat s is pi(s|a).

+ Value function = How good is a specific action or a specific state for the agent? Function of state or state action pair that measure how good it is for an agent to be in a given state or how good it is for agent to take action in a given state. => Return the “expected return”

**Text

Description automatically generated**

**Text

Description automatically generated**

**2/ What do the RL Algorithms Learn?** **- Optimal Policies & Optimal Values**

* In terms of return, a policy pi is considered to be better than or the same as policy pi prime if the expected return of pi is greater than or equal to the expected return of pi prime for all states
* Function value v gives the expected return for starting in state s and following pi thereafter. A policy that is better than or at least the same as all other policies is called the optimal policy

Graphical user interface, text, application

Description automatically generated

Text, letter

Description automatically generated

Graphical user interface, text, application

Description automatically generated

**3/ Q Learning Explained:**

Text, letter

Description automatically generated

* Q learning updated the Bellman optimalizy equation until reaching q\*
* To balance the exploration and exploitation we make use of the epsilon greedy strategy

**4/ Exploration vs Exploitation – Epsilon Greedy Strategy:**

* E = 1, then the agent will 100% exploration
* E = 0, then the agent will 100% exploitation

Diagram

Description automatically generated

Text

Description automatically generated

**5/ DQN:**

* Use NN to predict the Q function

A screenshot of a computer

Description automatically generated with low confidence

Loss = between the predicted Q functions with the right hand side of the Bellman equation.

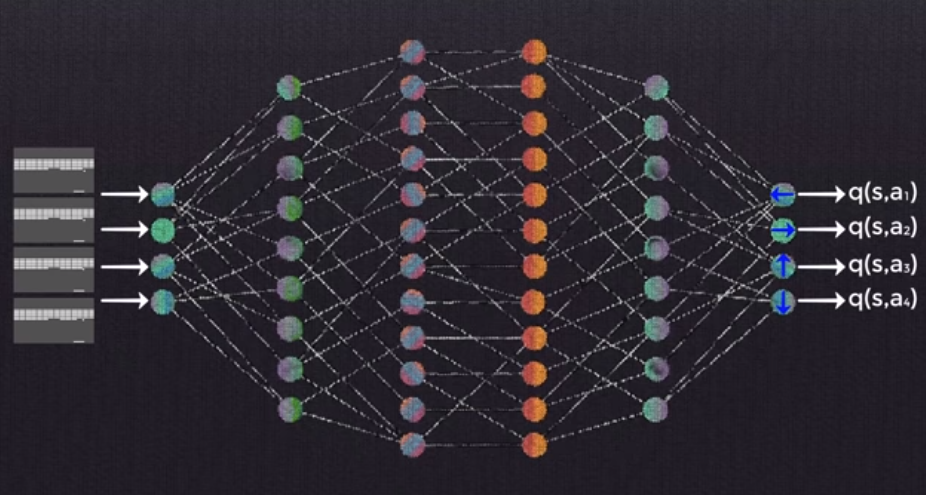
Objective is minimize the loss.

* Instead of using value iteration, we now use the DNN.
* For frozen lake, we use the grid of the game as input or state of the neural network.
* For more complex world like games, we will use images as input to DNN

+ Can gray scale and crop

**+** Instead of having 1 image as input, we will use a stack of images as input. => This is representation of the state of the env.

+ Layers: Just CNN



**6/ Replay Memory Explained – Experience for DQN Training:**

* Experience Replay: We store the agent’s experience at each time step in a dataset a replay memory

Diagram

Description automatically generated

* Replay Memory: All agent’s experience at each time step are stored n the Replay memory.
* Instead of inputing sequential replay we do random replay to break the correlation between them since it will cause unefficient learning

Text

Description automatically generated

* In the Frozen Lake game, we can see that choosing the q\_max value can be done by using look up Q table. Now with CNN or DNN, you must pick Q\_max within the available provided Q values.
* After getting Q\_max, we can calculate Q\* as the predicted value: Text

  Description automatically generated with medium confidence. Then we can calculate the loss: Diagram

  Description automatically generated with low confidence

+ Basically, for the same action, we calculate Q (prediction from the NN) deducted by the optimal Q value (Q\*) for the same action in the following timestep

Chart

Description automatically generated.

A picture containing text

Description automatically generated

* Do 2 fw pass to calculate loss before doing any gradient updates
* Graphical user interface, schematic

  Description automatically generated
* This is for a single time step => Do all step till end episode. Then do all episode till optimize the loss

Text

Description automatically generated

* Note that we will continue to update the weogjt of the policy netowk while keep the target network. However at the certain timestep we will use the weight from the policy network replacing for the target network

Text

Description automatically generated