Project Documentation :

------- ---------------

Core Application file:

---- ----------- -----

1) The first and main thing is to choose the proper environment

   1.1) The file varies from platform to pltform so if you wish to run it locally make sure you have the proper file.

2) Load all the essential libraires (They are mentioned in detail in the jupyter notebook)

3) Then to just run the environment with random actions to see how it performs

4) Import the DQN_Agent file as an alias agent

5) Load it with the necessary configuration

6) Defining the training loop

7) Train you Networks and save the model

8) Plot the scores to visualize the performance

DQN_Agent file:

--------- -----

1) The agent file is a file having main 2 classes :

1.1) The core agent class that coordinates actions,learns,updates parameters,etc.

1.2) The second main class is the experience replay class that dynamically stores the data and helps the agent learn from its

experiences.

2) This file utilizes the model file that defines the Neural Network architecture.

Model file:

----- -----

1) Define the model class

2) Add Linear layers to it

3) Add a forward function to access after the forward pass

1) Load the PyTorch library along with torch.nn & functional.

2) Define you ANN(Artificial Neural Network architecture).

Hyperparameters & Architecture :

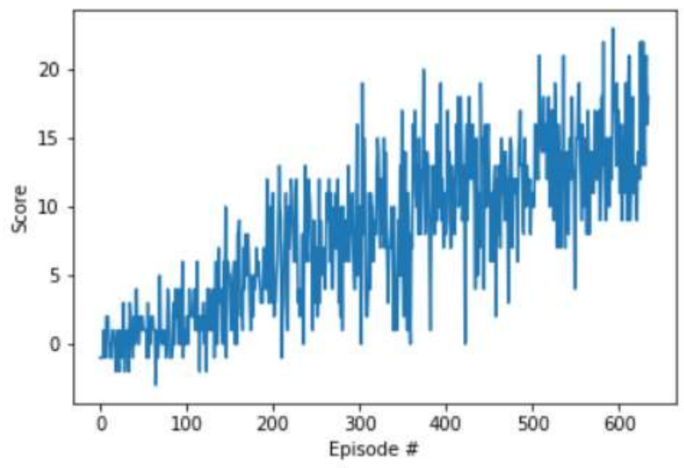
--------------- - --------------

The model is an ANN with 4 linear stages and a batchnorm at the beginning.

```
QNetwork(
  (fc1): Linear(in_features=37, out_features=512, bias=True)
  (fc2): Linear(in_features=512, out_features=256, bias=True)
  (fc3): Linear(in_features=256, out_features=128, bias=True)
  (out): Linear(in_features=128, out_features=4, bias=True)
  (bn): BatchNorm1d(37, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
)
```

```
BUFFER_SIZE = int(1e5)
BATCH_SIZE = 64
GAMMA = 0.995
TAU = 1e-3
LR = 5e-4
UPDATE_EVERY = 4
num_episodes = 2500
max_t = 1000
eps_start = 1.0
eps_end =0.01
eps_decay = 0.995
```

Reward Plot


Score
Episode #