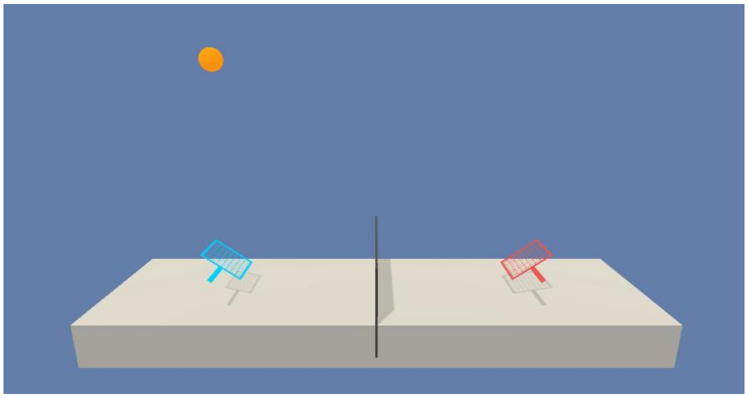
Udacity Deep Reinforcement Learning Nano Degree

Project #3 – Collaboration and Competition

By Kevin Lee - December 11, 2018

Overview

The objective of this project is to train two agents to play tennis using multi-agent DDPG.



In this environment, two agents control rackets to bounce a ball over a net. If an agent hits the ball over the net, it receives a reward of +0.1. If an agent lets a ball hit the ground or hits the ball out of bounds, it receives a reward of -0.01. Thus, the goal of each agent is to keep the ball in play.

The observation space consists of 8 variables corresponding to the position and velocity of the ball and racket. Each agent receives its own, local observation. Two continuous actions are available, corresponding to movement toward (or away from) the net, and jumping.

The task is episodic, and in order to solve the environment, your agents must get an average score of +0.5 (over 100 consecutive episodes, after taking the maximum over both agents). Specifically,

- After each episode, we add up the rewards that each agent received (without discounting), to get a score for each agent. This yields 2 (potentially different) scores. We then take the maximum of these 2 scores.
- This yields a single score for each episode.
- The environment is considered solved, when the average (over 100 episodes) of those scores is at least +0.5.

Implementation

The base code of this project is adapted from the code for project #2 Continuous Control. The code implements a multi-agent DDPG that trains two agents. There are three critical files : the Tennis.ipynb notebook, model.py and ddpg_agent.py.

Tennis.ipynb - this notebook sets up the environment and loops through the learning algorithm. It calls on functions in ddpg_agent.py.

ddpg_agent.py - contains the ddpg algorithm which calls on the neural networks in model.py.

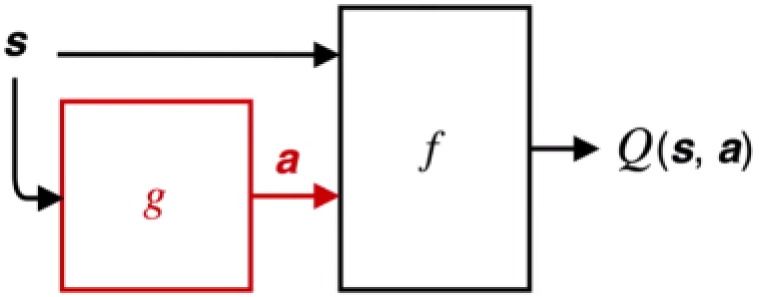
model.py - sets up the Actor and Critic neural networks

Algorithm

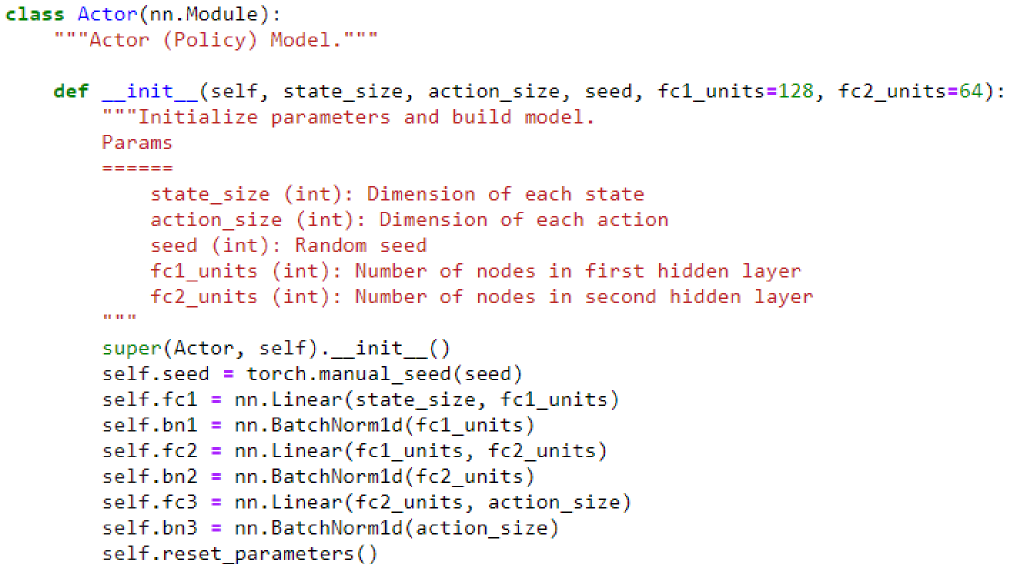
The main algorithm is based on the MADDPG algorithm learned in the DRLND course and by referring to this paper https://papers.nips.cc/paper/7217-multi-agent-actor-critic-for-mixed-cooperative-competitive-environments.pdf

Network Model

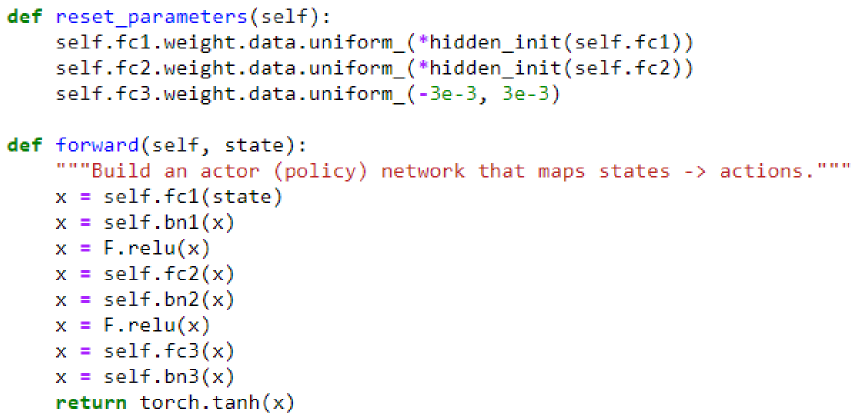
The model contains primarily two neural networks : the Actor and Critic network. The actor network is used to produce Actions that is fed into the Critic Network for evaluation and optimization. The two DNNs are used as function approximators. As shown below, the g function approximates the Actions (a) and the f function approximates the Q(s,a) value. The g network is the Actor network and the f network is the Critic network.



In this project, the Actor network is implemented as an Actor Class

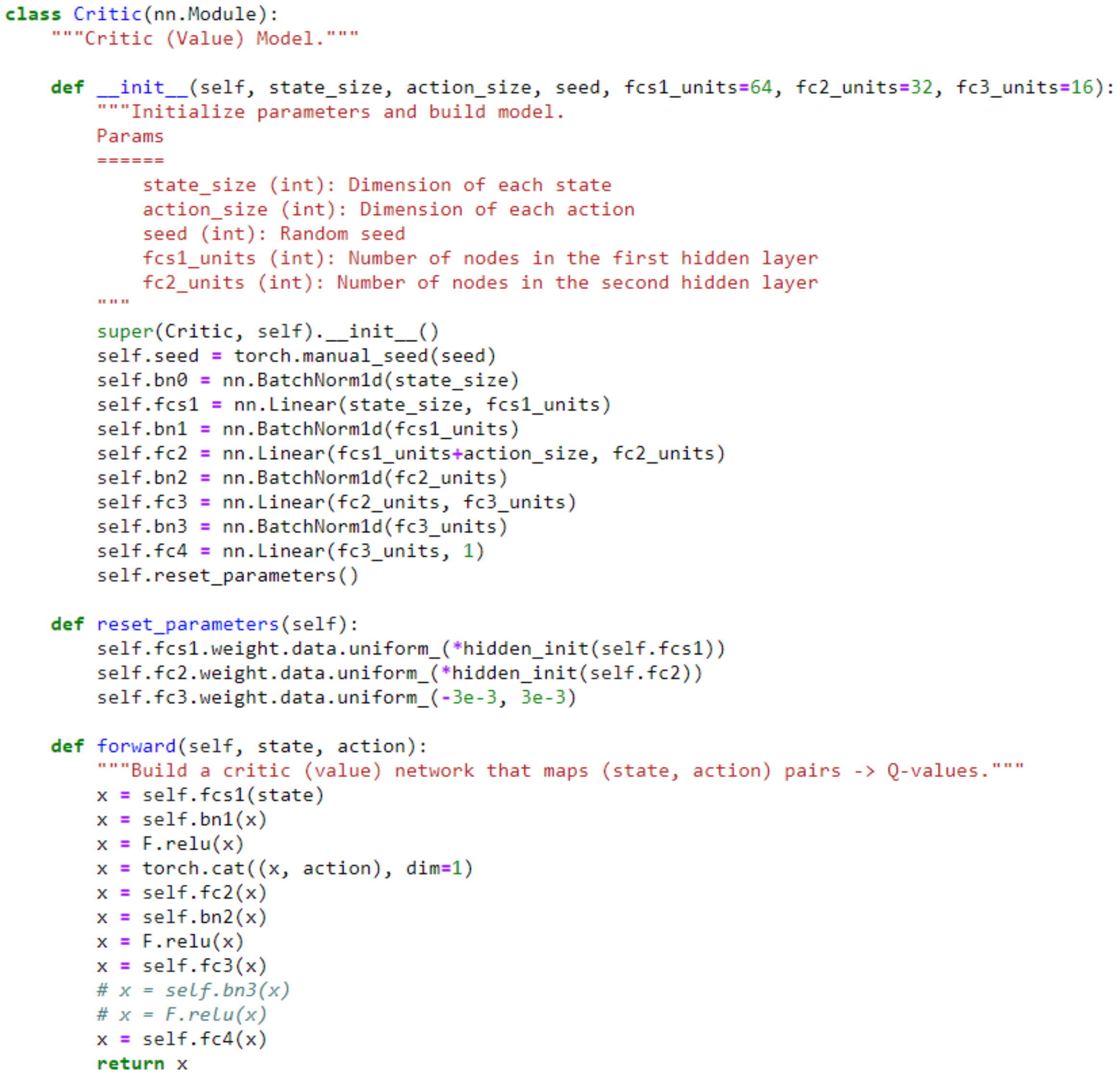


```
class Actor(nn.Module):
    """Actor (Policy) Model."""

    def __init__(self, state_size, action_size, seed, fc1_units=128, fc2_units=64):
        """Initialize parameters and build model.
        Params
        ======
            state_size (int): Dimension of each state
            action_size (int): Dimension of each action
            seed (int): Random seed
            fc1_units (int): Number of nodes in first hidden layer
            fc2_units (int): Number of nodes in second hidden layer
        """
        super(Actor, self).__init__()
        self.seed = torch.manual_seed(seed)
        self.fc1 = nn.Linear(state_size, fc1_units)
        self.bn1 = nn.BatchNorm1d(fc1_units)
        self.fc2 = nn.Linear(fc1_units, fc2_units)
        self.bn2 = nn.BatchNorm1d(fc2_units)
        self.fc3 = nn.Linear(fc2_units, action_size)
        self.bn3 = nn.BatchNorm1d(action_size)
        self.reset_parameters()
```



```
    def reset_parameters(self):
        self.fc1.weight.data.uniform_(*hidden_init(self.fc1))
        self.fc2.weight.data.uniform_(*hidden_init(self.fc2))
        self.fc3.weight.data.uniform_(-3e-3, 3e-3)

    def forward(self, state):
        """Build an actor (policy) network that maps states -> actions."""
        x = self.fc1(state)
        x = self.bn1(x)
        x = F.relu(x)
        x = self.fc2(x)
        x = self.bn2(x)
        x = F.relu(x)
        x = self.fc3(x)
        x = self.bn3(x)
        return torch.tanh(x)
```

The Critic class is also implemented as a fully connected DNN as below.



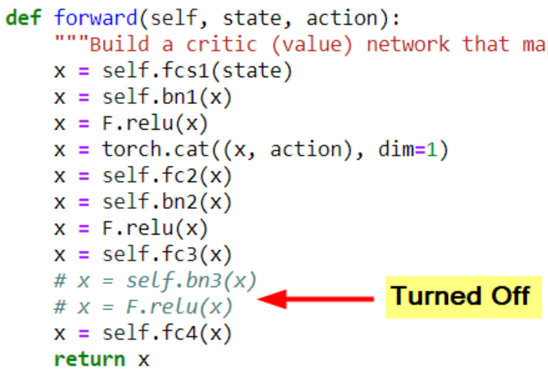
```
class Critic(nn.Module):
    """Critic (Value) Model."""

    def __init__(self, state_size, action_size, seed, fcs1_units=64, fc2_units=32, fc3_units=16):
        """Initialize parameters and build model.
        Params
        ======
            state_size (int): Dimension of each state
            action_size (int): Dimension of each action
            seed (int): Random seed
            fcs1_units (int): Number of nodes in the first hidden layer
            fc2_units (int): Number of nodes in the second hidden layer
        """
        super(Critic, self).__init__()
        self.seed = torch.manual_seed(seed)
        self.bn0 = nn.BatchNorm1d(state_size)
        self.fcs1 = nn.Linear(state_size, fcs1_units)
        self.bn1 = nn.BatchNorm1d(fcs1_units)
        self.fc2 = nn.Linear(fcs1_units+action_size, fc2_units)
        self.bn2 = nn.BatchNorm1d(fc2_units)
        self.fc3 = nn.Linear(fc2_units, fc3_units)
        self.bn3 = nn.BatchNorm1d(fc3_units)
        self.fc4 = nn.Linear(fc3_units, 1)
        self.reset_parameters()

    def reset_parameters(self):
        self.fcs1.weight.data.uniform_(*hidden_init(self.fcs1))
        self.fc2.weight.data.uniform_(*hidden_init(self.fc2))
        self.fc3.weight.data.uniform_(-3e-3, 3e-3)

    def forward(self, state, action):
        """Build a critic (value) network that maps (state, action) pairs -> Q-values."""
        x = self.fcs1(state)
        x = self.bn1(x)
        x = F.relu(x)
        x = torch.cat((x, action), dim=1)
        x = self.fc2(x)
        x = self.bn2(x)
        x = F.relu(x)
        x = self.fc3(x)
        # x = self.bn3(x)
        # x = F.relu(x)
        x = self.fc4(x)
        return x
```

I started off with three layers each for the actors and critics. I played around with the learning rate and the number of units for each layers. The results weren't satisfactory. Increasing the number of units didn't help. It actually made the results worse. Then I started to play with the depth of the network. Better results were achieved by making the critic network deeper and narrower but the actor network shallower and wider.

In addition, I experimented with batch normalization. The results improved somewhat. It's worth noting that where batch normalization is implemented makes a significant difference. I refered to the youtube video by Andrew Ng (https://www.youtube.com/watch?v=tNIpEZLv_eg) talking about this topic. His recommendation is to use method #1 and indeed that produced much better results.

Method 1	**Method 2**
x = self.fc1(state)	x = self.fc1(state)
x = self.bn1(x)	x = F.relu(x)
x = F.relu(x)	x = self.bn1(x)

After that I also experimented with some other tweaks to the network model. Strangely, I was able to get much better results when I turned off the batch normalization as well as the activation function between fc3 and fc4. Apparently two fully connected layers directly connected worked better.

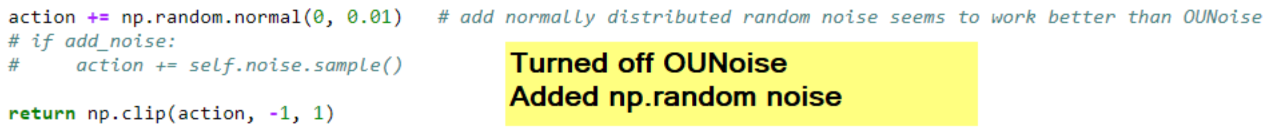


```
def forward(self, state, action):
    """Build a critic (value) network that maps (state, action) pairs
    x = self.fcs1(state)
    x = self.bn1(x)
    x = F.relu(x)
    x = torch.cat((x, action), dim=1)
    x = self.fc2(x)
    x = self.bn2(x)
    x = F.relu(x)
    x = self.fc3(x)
    # x = self.bn3(x)
    # x = F.relu(x)
    x = self.fc4(x)
    return x
```

Turned Off

Noise

One other thing that I changed was the noise. I experimented with different type of noise and found that a simple random number noise with a small std dev of 0.01 worked better than the default OUNoise.



```
action += np.random.normal(0, 0.01)   # add normally distributed random noise seems to work better than OUNoise
# if add_noise:
#     action += self.noise.sample()

return np.clip(action, -1, 1)
```

Turned off OUNoise
Added np.random noise

Refer to the Appendix at the end of this report for details. The appendix captures the sample set of results for the many experimentations done for this project.

Hyper-Parameters

There are other parameters that could be tweaked.

Buffer size I tried going from small (10) to big (2048). It didn't work that well when it's extremely small. That makes sense. After increasing it to the size of 128 or more, then this setting doesn't make a huge difference any more.

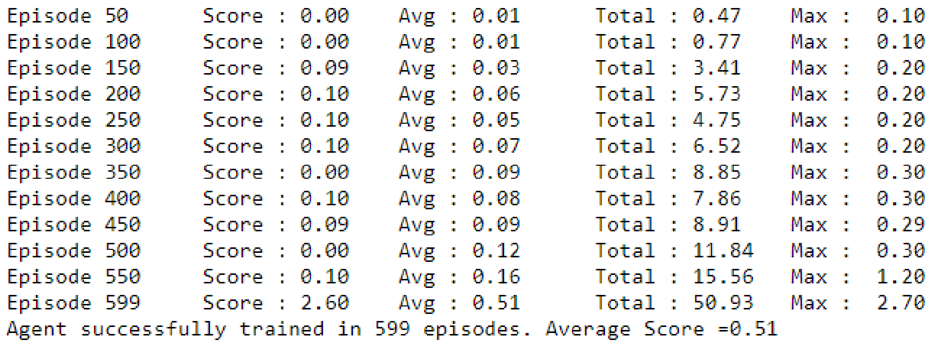
learn_steps This means learning is skipped. It's done for every "lean_steps". I played around with between 5 to 30. Settled with 10

learning_episodes Once learning is activated, it'll do a few times, specified by this parameter. Again, I played with a few combinations between learn_steps and learning episodes. I settled with 20 for this parameter.

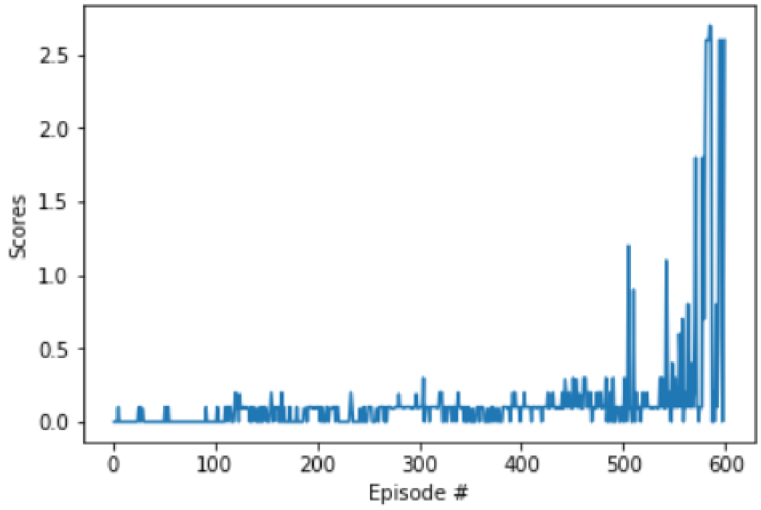
learning rate Surprisingly this didn't make a huge difference. Therefore I used the default setting for both Actor and Critic

<u>Results</u>

The training was successful, attaining an average score (over 100 episodes) of >0.50 in 599 episodes.



```
Episode 50      Score : 0.00    Avg : 0.01      Total : 0.47    Max :  0.10
Episode 100     Score : 0.00    Avg : 0.01      Total : 0.77    Max :  0.10
Episode 150     Score : 0.09    Avg : 0.03      Total : 3.41    Max :  0.20
Episode 200     Score : 0.10    Avg : 0.06      Total : 5.73    Max :  0.20
Episode 250     Score : 0.10    Avg : 0.05      Total : 4.75    Max :  0.20
Episode 300     Score : 0.10    Avg : 0.07      Total : 6.52    Max :  0.20
Episode 350     Score : 0.00    Avg : 0.09      Total : 8.85    Max :  0.30
Episode 400     Score : 0.10    Avg : 0.08      Total : 7.86    Max :  0.30
Episode 450     Score : 0.09    Avg : 0.09      Total : 8.91    Max :  0.29
Episode 500     Score : 0.00    Avg : 0.12      Total : 11.84   Max :  0.30
Episode 550     Score : 0.10    Avg : 0.16      Total : 15.56   Max :  1.20
Episode 599     Score : 2.60    Avg : 0.51      Total : 50.93   Max :  2.70
Agent successfully trained in 599 episodes. Average Score =0.51
```



Ideas for Future Work

I didn't spend enough time tweaking all the possible hyper-parameters. For example – the learning rate, although I did play around with some variations, I believe if I try harder, I might get different results.

In addition, the learn_steps can be further tweaked, perhaps using a slow decay. Also learning_episodes can be explored further, now by increasing the number of rounds to train in the later stage.

I'm most interested in adapting this algorithm to train an agent to trade the options market. Perhaps markets can be viewed as agents competiting against each other. Therefore, I will explore how to change this algorithm from a collaboration to competition.

In addition, I believe in the markets, many of the factors are unobservable. As a result, I cannot assume full markov states but instead have to use the partially observable markov decision process (POMDP). I need to research on this topic.

One more thing is that market data is a time series. I would need to learn more about time series analysis as well as incorporate RNN into the structure.

Appendix – Experimental Results

#	Program Parameters	Network Parameters	Results
1	learn_steps : 10 batch_size : 128 learn_episode : 20 seed : 8	Actor - 128x64 Critic - 64x32c Batch Norm : Yes	Episode 100 Score : 0.00 Avg : 0.01 Total : 0.98 Min : 0.00 Max : 0.20 Avg Steps: 13.00 Episode 200 Score : 0.00 Avg : 0.04 Total : 4.20 Min : 0.00 Max : 0.40 Avg Steps: 13.00 Episode 300 Score : 0.00 Avg : 0.01 Total : 0.67 Min : 0.00 Max : 0.20 Avg Steps: 13.00 Episode 400 Score : 0.00 Avg : 0.00 Total : 0.00 Min : 0.00 Max : 0.00 Avg Steps: 13.00 Episode 500 Score : 0.00 Avg : 0.00 Total : 0.00 Min : 0.00 Max : 0.00 Avg Steps: 13.00 Episode 548 Score : 0.00 Avg : 0.00 Total : 0.00 Min : 0.00 Max : 0.00 Avg Steps: 13.00
2	learn_steps : 10 batch_size : 128 learn_episode : 20 seed : 8	Actor - 128x64 Critic - 128x64c Batch Norm : Yes	Episode 100 Score : 0.00 Avg : 0.02 Total : 2.18 Min : 0.00 Max : 0.20 Avg Steps: 13.00 Episode 200 Score : 0.10 Avg : 0.02 Total : 1.98 Min : 0.00 Max : 0.10 Avg Steps: 25.00 Episode 300 Score : 0.00 Avg : 0.02 Total : 1.85 Min : 0.00 Max : 0.20 Avg Steps: 13.00 Episode 400 Score : 0.00 Avg : 0.00 Total : 0.10 Min : 0.00 Max : 0.10 Avg Steps: 19.00 Episode 500 Score : 0.00 Avg : 0.01 Total : 0.57 Min : 0.00 Max : 0.10 Avg Steps: 13.00
3	learn_steps : 10 batch_size : 128 learn_episode : 20 seed : 8	Actor - 128x64 Critic - 512x128c Batch Norm : Yes	Episode 100 Score : 0.00 Avg : 0.01 Total : 0.58 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 200 Score : 0.00 Avg : 0.01 Total : 1.37 Min : 0.00 Max : 0.50 Avg Steps: 13.00 Episode 300 Score : 0.00 Avg : 0.00 Total : 0.10 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 400 Score : 0.10 Avg : 0.01 Total : 0.96 Min : 0.00 Max : 0.10 Avg Steps: 29.00 Episode 500 Score : 0.00 Avg : 0.03 Total : 3.33 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 600 Score : 0.00 Avg : 0.03 Total : 3.16 Min : 0.00 Max : 0.20 Avg Steps: 13.00 Episode 700 Score : 0.00 Avg : 0.05 Total : 4.60 Min : 0.00 Max : 0.20 Avg Steps: 13.00 Episode 800 Score : 0.00 Avg : 0.01 Total : 1.48 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 900 Score : 0.00 Avg : 0.02 Total : 1.78 Min : 0.00 Max : 0.10 Avg Steps: 14.00 Episode 1000 Score : 0.00 Avg : 0.02 Total : 1.69 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 1100 Score : 0.00 Avg : 0.00 Total : 0.00 Min : 0.00 Max : 0.00 Avg Steps: 13.00 Episode 1200 Score : 0.00 Avg : 0.00 Total : 0.00 Min : 0.00 Max : 0.00 Avg Steps: 13.00
4	learn_steps : 10 batch_size : 128 learn_episode : 20 seed : 8	Actor - 128x64 Critic - 512x128 BN Actor : Yes BN Critic : Yes	Episode 100 Score : 0.00 Avg : 0.00 Total : 0.29 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 200 Score : 0.00 Avg : 0.00 Total : 0.00 Min : 0.00 Max : 0.00 Avg Steps: 13.00 Episode 300 Score : 0.00 Avg : 0.00 Total : 0.00 Min : 0.00 Max : 0.00 Avg Steps: 13.00 Episode 400 Score : 0.00 Avg : 0.00 Total : 0.00 Min : 0.00 Max : 0.00 Avg Steps: 13.00 Episode 500 Score : 0.00 Avg : 0.00 Total : 0.10 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 600 Score : 0.00 Avg : 0.00 Total : 0.10 Min : 0.00 Max : 0.10 Avg Steps: 13.00
5	learn_steps : 10 batch_size : 128 learn_episode : 20 seed : 8	Actor - 128x64 Critic - 64x32x16 BN Actor : Yes BN Critic : No	Episode 100 Score : 0.00 Avg : 0.01 Total : 0.96 Min : 0.00 Max : 0.19 Avg Steps: 13.00 Episode 200 Score : 0.00 Avg : 0.00 Total : 0.49 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 300 Score : 0.00 Avg : 0.02 Total : 1.65 Min : 0.00 Max : 0.20 Avg Steps: 13.00 Episode 400 Score : 0.00 Avg : 0.04 Total : 3.58 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 500 Score : 0.10 Avg : 0.08 Total : 7.81 Min : 0.00 Max : 0.10 Avg Steps: 43.00 Episode 534 Score : 2.60 Avg : 0.12 Total : 11.76 Min : 0.00 Max : 2.60 Avg Steps: nan Episode 600 Score : 0.10 Avg : 0.17 Total : 16.52 Min : 0.10 Max : 2.60 Avg Steps: 50.00 Episode 645 Score : 0.10 Avg : 0.33 Total : 32.55 Min : 0.10 Max : 2.70 Avg Steps: 50.00
6	learn_steps : 10 batch_size : 128	Actor - 128x64 Critic - 64x32x16 BN Actor : Yes	Episode 100 Score : 0.00 Avg : 0.00 Total : 0.40 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 200 Score : 0.00 Avg : 0.00 Total : 0.37 Min : 0.00 Max : 0.10 Avg Steps: 13.00

	learn_episode : 20 seed : 8	BN Critic : Yes	Episode 300 Score : 0.10 Avg : 0.03 Total : 2.66 Min : 0.00 Max : 0.20 Avg Steps: 30.00 Episode 400 Score : 0.09 Avg : 0.05 Total : 4.74 Min : 0.00 Max : 0.20 Avg Steps: 29.00 Episode 500 Score : 0.00 Avg : 0.03 Total : 3.08 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 600 Score : 0.00 Avg : 0.04 Total : 3.60 Min : 0.00 Max : 0.19 Avg Steps: 15.00 Episode 700 Score : 0.00 Avg : 0.05 Total : 4.88 Min : 0.00 Max : 0.30 Avg Steps: 14.00 Episode 800 Score : 0.00 Avg : 0.04 Total : 3.66 Min : 0.00 Max : 0.20 Avg Steps: 27.00 Episode 900 Score : 0.10 Avg : 0.04 Total : 3.74 Min : 0.00 Max : 0.20 Avg Steps: 24.00 Episode 1000 Score : 0.00 Avg : 0.02 Total : 2.27 Min : 0.00 Max : 0.10 Avg Steps: 14.00 Episode 1100 Score : 0.00 Avg : 0.04 Total : 3.82 Min : 0.00 Max : 0.30 Avg Steps: 27.00 Episode 1200 Score : 0.00 Avg : 0.02 Total : 1.84 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 1237 Score : 0.00 Avg : 0.01 Total : 1.46 Min : 0.00 Max : 0.10 Avg Steps: 13.00
7	learn_steps : 10 batch_size : 128 learn_episode : 20 seed : 8	Actor - 128x64 Critic - 64x32x16 BN Actor : Yes BN Critic : Yes BN implemented before ReLU for Critic	Episode 100 Score : 0.00 Avg : 0.01 Total : 0.70 Min : 0.00 Max : 0.10 Avg Steps: 14.00 Episode 200 Score : 0.00 Avg : 0.04 Total : 3.61 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 300 Score : 0.00 Avg : 0.01 Total : 0.50 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 400 Score : 0.10 Avg : 0.05 Total : 5.04 Min : 0.00 Max : 0.20 Avg Steps: 30.00 Episode 500 Score : 0.10 Avg : 0.09 Total : 8.62 Min : 0.00 Max : 0.20 Avg Steps: 29.00 Episode 600 Score : 0.10 Avg : 0.09 Total : 8.81 Min : 0.00 Max : 0.20 Avg Steps: 51.00 Episode 700 Score : 0.80 Avg : 0.20 Total : 19.56 Min : 0.00 Max : 2.60 Avg Steps: 319.00 Episode 796 Score : 0.30 Avg : 0.48 Total : 47.79 Min : 0.00 Max : 2.60 Avg Steps: 121.00
8	learn_steps : 10 batch_size : 128 learn_episode : 20 seed : 8	Actor - 128x64 Critic - 64x32x16 BN Actor : Yes BN Critic : Yes BN implemented before ReLU for Critic and Actor	Episode 100 Score : 0.00 Avg : 0.04 Total : 3.73 Min : 0.00 Max : 0.20 Avg Steps: 13.00 Episode 200 Score : 0.00 Avg : 0.01 Total : 0.86 Min : 0.00 Max : 0.10 Avg Steps: 13.00 Episode 300 Score : 0.00 Avg : 0.09 Total : 8.68 Min : 0.00 Max : 0.40 Avg Steps: 12.00 Episode 400 Score : 0.20 Avg : 0.12 Total : 12.10 Min : 0.00 Max : 0.50 Avg Steps: 77.00 Episode 500 Score : 0.70 Avg : 0.29 Total : 29.37 Min : 0.00 Max : 2.60 Avg Steps: 260.00 Episode 600 Score : 0.10 Avg : 0.60 Total : 59.58 Min : 0.10 Max : 2.60 Avg Steps: 48.00 Episode 700 Score : 0.00 Avg : 0.60 Total : 60.12 Min : 0.00 Max : 2.60 Avg Steps: 12.00
9	learn_steps : 10 batch_size : 128 learn_episode : 20 seed : 8	Actor - 128x64 Critic - 64x32x16 BN Actor : Yes BN Critic : Yes BN implemented for last layer in Actor	Episode 100 Score : 0.09 Avg : 0.02 Total : 2.49 Max : 0.10 Avg Steps: 31.00 Episode 200 Score : 0.00 Avg : 0.06 Total : 5.71 Max : 0.20 Avg Steps: 15.00 Episode 300 Score : 0.10 Avg : 0.07 Total : 7.31 Max : 0.30 Avg Steps: 47.00 Episode 400 Score : 0.00 Avg : 0.06 Total : 5.79 Max : 0.19 Avg Steps: 13.00 Episode 500 Score : 0.00 Avg : 0.00 Total : 0.40 Max : 0.10 Avg Steps: 13.00 Episode 600 Score : 0.00 Avg : 0.00 Total : 0.29 Max : 0.10 Avg Steps: 13.00 Episode 700 Score : 0.10 Avg : 0.05 Total : 5.21 Max : 0.20 Avg Steps: 30.00 Episode 800 Score : 0.10 Avg : 0.09 Total : 8.88 Max : 0.20 Avg Steps: 68.00
10	learn_steps : 10 batch_size : 128 learn_episode : 20 seed : 8	Actor - 128x64 Critic - 64x32x16 BN Actor : Yes BN Critic : Yes Last layer BN in Actor Removed Removed Noise	Episode 100 Score : 0.00 Avg : 0.02 Total : 2.26 Max : 0.19 Avg Steps: 13.00 Episode 200 Score : 0.00 Avg : 0.04 Total : 3.90 Max : 0.10 Avg Steps: 13.00 Episode 300 Score : 0.20 Avg : 0.04 Total : 4.34 Max : 0.50 Avg Steps: 90.00 Episode 400 Score : 0.10 Avg : 0.13 Total : 13.23 Max : 1.50 Avg Steps: 31.00 Episode 500 Score : 0.00 Avg : 0.08 Total : 8.01 Max : 0.50 Avg Steps: 13.00 Episode 600 Score : 0.19 Avg : 0.13 Total : 12.94 Max : 0.60 Avg Steps: 62.00
11	learn_steps : 10 batch_size : 128 learn_episode : 20 seed : 8	Actor - 128x64 Critic - 64x32x16 BN Actor : Yes BN Critic : Yes	Episode 100 Score : 0.00 Avg : 0.01 Total : 1.37 Max : 0.10 Avg Steps: 14.00 Episode 200 Score : 0.00 Avg : 0.00 Total : 0.00 Max : 0.00 Avg Steps: 13.00 Episode 300 Score : 0.10 Avg : 0.04 Total : 4.37 Max : 0.20 Avg Steps: 69.00 Episode 400 Score : 0.00 Avg : 0.06 Total : 5.57 Max : 0.20 Avg Steps: 13.00

		Changed Noise to np.random	Episode 500 Score : 0.00 Avg : 0.08 Total : 8.26 Max : 0.49 Avg Steps: 17.00 Episode 600 Score : 0.10 Avg : 0.24 Total : 24.25 Max : 2.60 Avg Steps: 30.00 Episode 673 Score : 2.60 Avg : 0.51 Total : 50.93 Max : 2.70 Avg Steps: nan Agent successfully trained in 673 episodes. Average Score =0.51
12	learn_steps : 10 batch_size : 128 learn_episode : 20 seed : 8	Actor - 128x64 Critic - 64x32x16 BN Actor : Yes BN Critic : Yes Noise set to OUNoise	Episode 100 Score : 0.00 Avg : 0.00 Total : 0.28 Max : 0.10 Avg Steps: 13.00 Episode 200 Score : 0.00 Avg : 0.00 Total : 0.00 Max : 0.00 Avg Steps: 13.00 Episode 300 Score : 0.00 Avg : 0.00 Total : 0.00 Max : 0.00 Avg Steps: 13.00 Episode 400 Score : 0.00 Avg : 0.02 Total : 1.72 Max : 0.10 Avg Steps: 24.00 Episode 500 Score : 0.00 Avg : 0.00 Total : 0.30 Max : 0.10 Avg Steps: 14.00 Episode 533 Score : 0.00 Avg : 0.00 Total : 0.40 Max : 0.10 Avg Steps: 13.00