P.M. Lucaci

Deep reinforcement learning Nanodegree, Udacity

Project 3: Tennis

Contents

[Learning Algorithm 2](#_Toc39602674)

[Hyperparameters 2](#_Toc39602675)

[NN Model Architecture 2](#_Toc39602676)

[Plot of Rewards 3](#_Toc39602677)

[Ideas for Future Work 4](#_Toc39602678)

[Improving the report 4](#_Toc39602679)

[Including project details 4](#_Toc39602680)

[GIF of Trained Agent 4](#_Toc39602681)

[Action Space 4](#_Toc39602682)

[State Space 4](#_Toc39602683)

[Rewards 4](#_Toc39602684)

[Solving the Environment 4](#_Toc39602685)

[Documenting Future Improvements of Learning Algorithm 4](#_Toc39602686)

[Add Bayesian Optimization 4](#_Toc39602687)

[Add Prioritised Replay Buffer 4](#_Toc39602688)

[Add Progress Bar 4](#_Toc39602689)

[Add TensorBoard 4](#_Toc39602690)

# Learning Algorithm

DDPG stands for “Deep Deterministic Policy Gradient”, which is an algorithm learning concurrently a Q-function and a policy. This algorithm, with a shared Replay Buffer, has been implemented to train and evaluate the two agents involved in this environment.

## Hyperparameters

|  |  |
| --- | --- |
| n\_episodes = 30000 | # maximum no. of episodes to train |
| max\_t = 2000 | # maximum no. of time steps |
| num\_agents = 2 | # no. of agents |
| random\_seed = 0 | # random seed |
| eps\_start = NaN | # epsilon upper limit (before any decay) |
| eps\_end = NaN | # epsilon lower limit (minimum value) |
| eps\_decay = NaN | # epsilon decay rate |
| BUFFER\_SIZE = int(1e6) | # replay buffer size |
| BATCH\_SIZE = 512 | # minibatch size |
| GAMMA = 0.99 | # discount factor |
| UPDATE\_EVERY = 4 | # how often to update the network |
| TAU = 1e-3 | # for soft update of target parameters |
| LR\_ACTOR = 1e-4 | # learning rate actor |
| LR\_CRITIC = 5e-4 | # learning rate actor |
|  |  |
|  |  |

## NN Model Architecture

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Actor | BatchNorm1d | Linear | ReLu | Linear | ReLu | Linear | Tanh |
| in = out = state\_size | in = state\_size  out = 128 | | in = out = 128 | | in = 128  out = action\_size | |
| state\_size = 8 | | | | action\_size = 2 | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Critic | BatchNorm1d | Linear | ReLu | Linear | ReLu | Linear |
| in = out = state\_size | in = state\_size  out = 128 | | in = 128 + action\_size  out = 128 | | in = 128  out = 1 |
| state\_size = 8 | | | | action\_size = 2 | | |

# Plot of Rewards

|  |
| --- |
|  |
| Episode 3 Average Score: 0.639  **Environment solved in 3 episodes** |

# Ideas for Future Work

## Improving the report

### Including project details

#### GIF of Trained Agent

#### Action Space

#### State Space

#### Rewards

#### Solving the Environment

### Documenting Future Improvements of Learning Algorithm

## Add Bayesian Optimization

## Add Prioritised Replay Buffer

## Add Progress Bar

## Add Tensor Board