Table of Contents

Learning Algorithm

Deep Deterministic Policy Gradient (DDPG)

Algorithm 1 DDPG algorithm

Randomly initialize critic network $Q(s, a|\theta^Q)$ and actor $\mu(s|\theta^\mu)$ with weights θ^Q and θ^μ.
Initialize target network Q' and μ' with weights $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$
Initialize replay buffer R
for episode = 1, M **do**
  Initialize a random process $\mathcal{N}$ for action exploration
  Receive initial observation state s_1
  for t = 1, T **do**
    Select action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$ according to the current policy and exploration noise
    Execute action a_t and observe reward r_t and observe new state s_{t+1}
    Store transition (s_t, a_t, r_t, s_{t+1}) in R
    Sample a random minibatch of N transitions (s_i, a_i, r_i, s_{i+1}) from R
    Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
    Update critic by minimizing the loss: $L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$
    Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N} \sum_i \nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu} \mu(s|\theta^\mu)|_{s_i}$$

    Update the target networks:

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$
$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$

  end for
end for

Code Walk-through

File: Tennis.ipynb

1. Instantiate ddpg_agent
2. Initialize Variables
3. Outer Loop (2000 Episodes)
    a. Reset the Tennis Environment
    b. Retrieve the Initial State from the Environment
    c. Reset the Learning Agent
    d. Inner Loop (1 Full Episode)
        i. Get Action for Given State from Agent
        ii. Send the Action to the Environment
        iii. Get the New State
        iv. Get the Reward
        v. Check if Episode is Done
        vi. For Each Agent Loop
            - Transition to the next state
        vii Accumulate rewards for the Episode
        vii. Exit Loop when Done
    e. Store each Episode's Score for the last 100 Episodes
    f. Store each Episode's Score for Plotting

Model Architecture for Neural Network

Actor Model:

Three layer Neural Network

1st Layer takes as input the number of STATES, 128 units as output
2nd Layer – 128 units as input, 128 units as output
3rd Layer – 128 units as input, output is the number of ACTIONS

Rectified Linear Units connect the Layers together with a Batch Normalization for the first layer.

Critic Model:

Three layer Neural Network

1st Layer takes as input the number of STATES, 128 units as output
2nd Layer – 128 units + number of ACTIONS as input, 128 units as output
3rd Layer – 128 units as input, output is 1

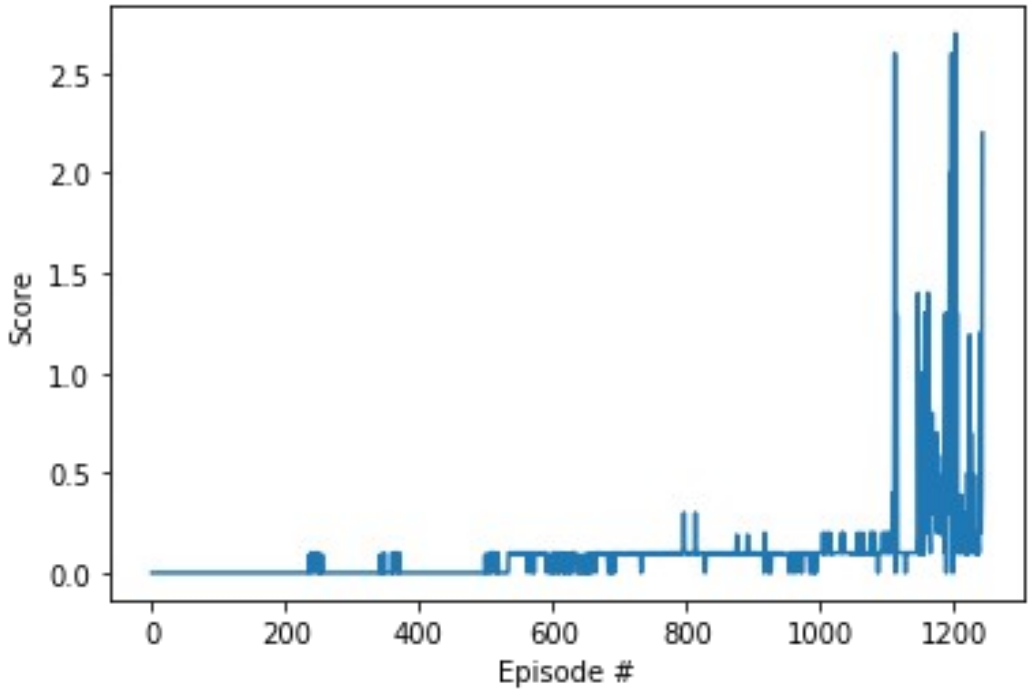
Rectified Linear Units connect the Layers together with a Batch Normalization for the first layer.

Hyper-parameters

Hyper Parameter	Value	Description
BUFFER_SIZE	1×10^{6}	replay buffer size
BATCH_SIZE	1024	Mini batch size
GAMMA	0.99	discount factor
TAU	1×10^{-3}	for soft update of target parameters
LR_ACTOR	2×10^{-4}	learning rate of the actor
LR_CRITIC	2×10^{-4}	learning rate of the critic
WEIGHT_DECAY	0	L2 weight decay
THETA	0.15	
SIGMA	0.1	

Results

(Goal: Average Score of .5 over 100 episodes)



Episode 100.....Average Score: 0.00
Episode 200.....Average Score: 0.00
Episode 300.....Average Score: 0.01
Episode 400.....Average Score: 0.01
Episode 500.....Average Score: 0.00
Episode 600.....Average Score: 0.07
Episode 700.....Average Score: 0.06
Episode 800.....Average Score: 0.10
Episode 900.....Average Score: 0.10
Episode 1000.....Average Score: 0.08
Episode 1100.....Average Score: 0.11
Episode 1200.....Average Score: 0.42
Episode 1242.....Average Score: 0.52
Environment solved in 1242 episodes!.....Average Score: 0.52

Ideas for Future Work

Further tuning and optimization of parameters to solve the problem in less episodes.

References

@misc{lillicrap2015continuous,

title={Continuous control with deep reinforcement learning},

author={Timothy P. Lillicrap and Jonathan J. Hunt and Alexander Pritzel and Nicolas Heess and Tom Erez and Yuval Tassa and David Silver and Daan Wierstra},

year={2015},

eprint={1509.02971},

archivePrefix={arXiv},

primaryClass={cs.LG}

}