BipedalWalker with TD3 Actor-Critic

Team

Asha Aher

Chinmay Vadgama

Goal

The main goal of this project is that to solve the problem characterized by the environment BipedalWalker-v3. (https://gym.openai.com/envs/BipedalWalker-v2/)

The problem arises as concluded when the robot chooses correct actions for don't fall and walk to the end of the path.

In particular, the reward returned by the environment is a value of the continuous space which depends on the distance the robot travels.

Formally the resolution of the problem occurs when the robot gets an average reward greater than 300 (path completed) for 100 consecutive episodes:

We therefore want to find a reinforcement algorithm learning able to achieve the goal mentioned in the shortest possible time.

Algorithm used

- **Twin Delayed Deep Deterministic Policy Gradients (TD3)**
  - TD3 is the successor to the Deep Deterministic Policy Gradient (DDPG)
  - Generally used in continuous control problem such as Robotics problems.
  - Inception on TD3 : DDPG worked well but it continuously over estimated the Q values of the critic (value) network. This errors accumulates and agent could get stuck in local minima or can experience catastrophic forgetting.

Why TD3?

DDPG is capable of providing excellent results but it has its drawbacks.

Like many RL algorithms training DDPG can be unstable and heavily reliant on finding the correct hyper parameters for the current task. Since DDPG is deterministic algorithm, it continuously over estimates Q values and this lead to higher covariance and ultimately getting local minima and agent failing in the environment by experiencing catastrophic forgetting.

TD3 Key features

1. Pair of critic networks hence twin part in the title
2. Delayed updates of the actor
3. Action noise regularisation

TD3 Key Features

Twin Critic Network: TD3 uses two separate critic networks, it uses small value among this two critic values while forming targets.

Actor - TD3 uses a delayed update of the actor network, only updating it every 2 time steps instead of after each time step, resulting in more stable and efficient training.

Noise Regularization - This part reduces variance by adding a random noise to the target.

TD3 Implementation

1. Initialise networks
2. Initialise replay buffer
3. Select and carry out action with exploration noise
4. Store transitions
5. Update critic
6. Update actor
7. Update target networks
8. Repeat until sentient

Training Process & Demo

100 episodes

In start of training agent picks

Staying in a same place instead of

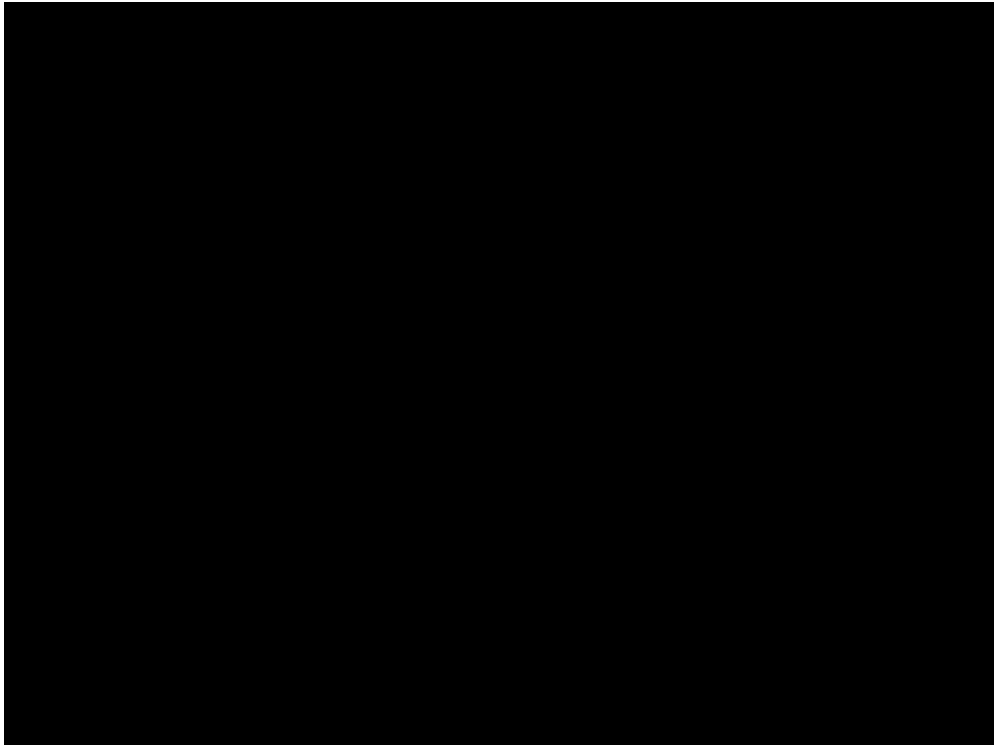
Moving away due to high negative

rewards.

Result

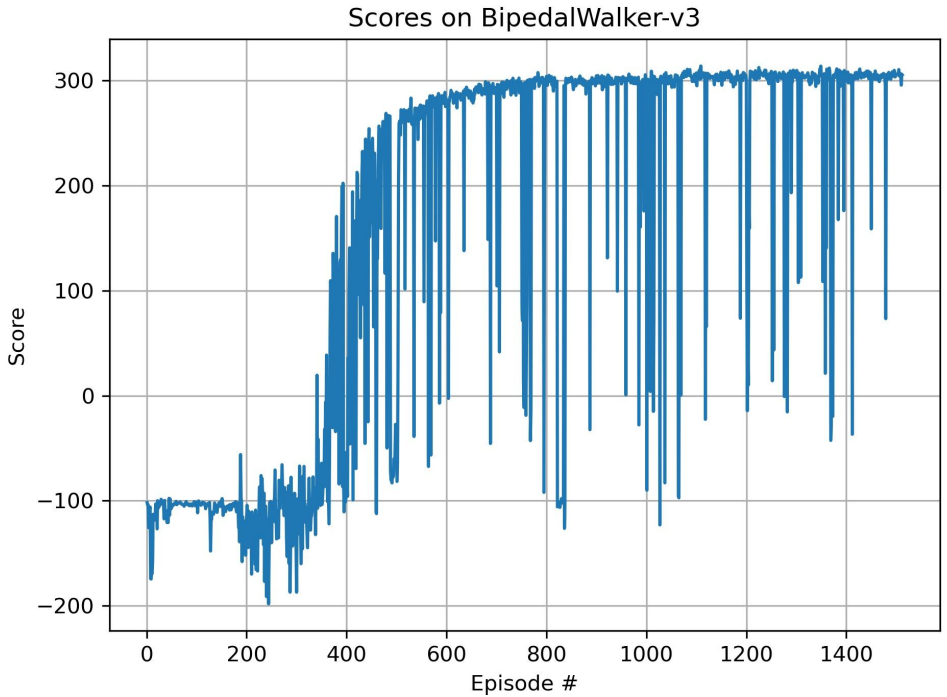
500+ episodes Rewards converge

To more than 300 value.



Evaluation

Reward change per Episode



Trained and tested on:

- Python 3.8
- gym 0.17.2
- box2d 2.3.10
- torch 1.5.0
- numpy 1.18.4
- Matplotlib 3.2.1
- Pillow 7.1.2

References

- “ Addressing Function Approximation Error in Actor-Critic Methods ” https://arxiv.org/pdf/1802.09477.pdf
- https://towardsdatascience.com/td3-learning-to-run-with-ai-40dfc512f93
- https://gym.openai.com/envs/BipedalWalker-v2/