IMPLIMENTATION OF POLICY GRADIENT AND DEEP Q NETWORKS ON OPENAI ENVIRONMENTS

Joseph Distefano

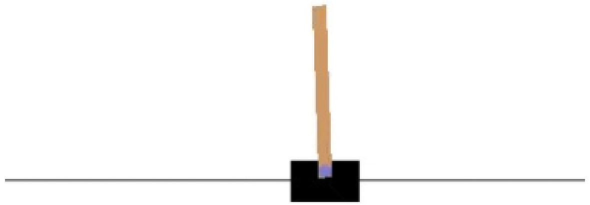
University at Buffalo
School of Engineering and Applied Sciences

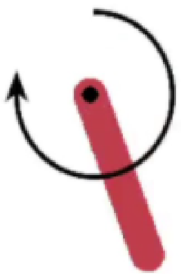
Project Description

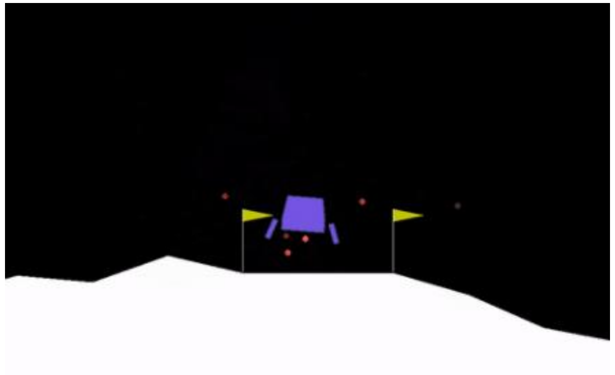
Goal 1:

Implement 3 deep q networks and 4 policy gradient methods on anon openai environment

- Deep Q Learning (DQN)
- Double Deep Q Learning (DDQN)
- Dueling Deep Q Learning (Dueling DQN)
- REINFORCE
- Advantage Actor Critic (A2C)
- Proximal Policy Optimization (PPO)
- Deep Deterministic Policy Gradient (DDPG)







Goal 2:

- Set up future work to continue pursuing reinforcement learning with Atari environment and original idea

Deep Q Learning vs Policy Gradient methods

	Deep Q Learning	Policy Gradient
Objective Function	The deep networks are used to predict Bellman's equation which gives the value function	The policy gradients try to maximize the expected return from the policy
On vs Off- policy	Off policy (value function)	On policy (policy)
Stability and Sample Efficiency	Not always optimal, more sample efficient (batch)	Sample inefficient but better behavior (stable)

DDPG algorithm

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:

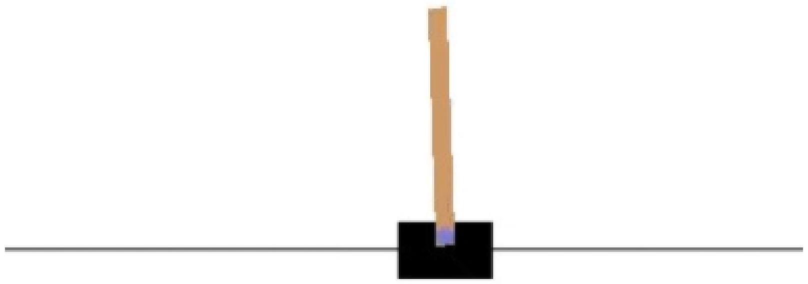
$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$

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

  end for
end for

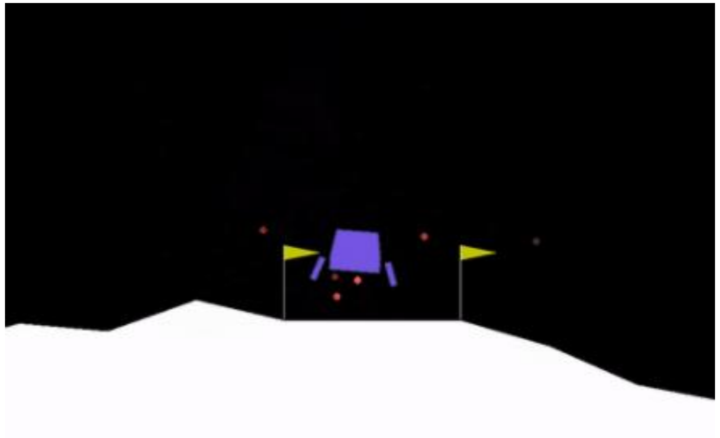
Environments

CartPole



Actions: Left or Right
Rewards = +1 for every time step
States = cart position, cart velocity, pole angle, tip velocity
Goal: stay up for as long as possible

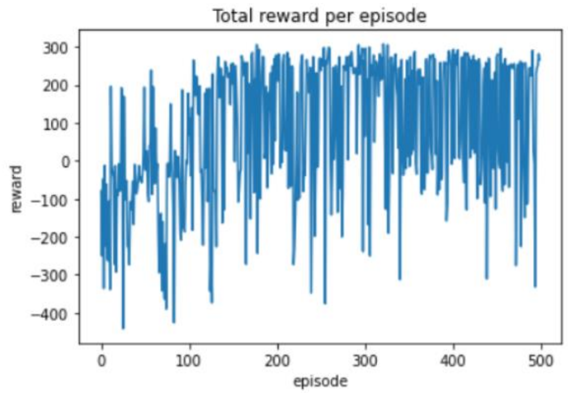
LunarLander



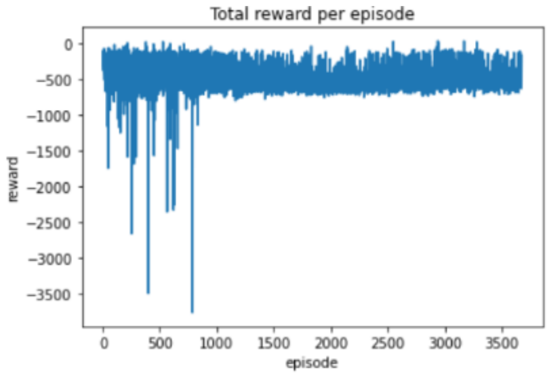
Actions: do nothing, fire left engine, fire down engine, fire right engine
Rewards = land in designated area, legs on ground, rest at episode end
States = horizontal and vertical position, horizontal and vertical velocity, angle and angular velocity, and left and right leg contact
Goal: land softly in defined area

Results for Deep Q Learning Method

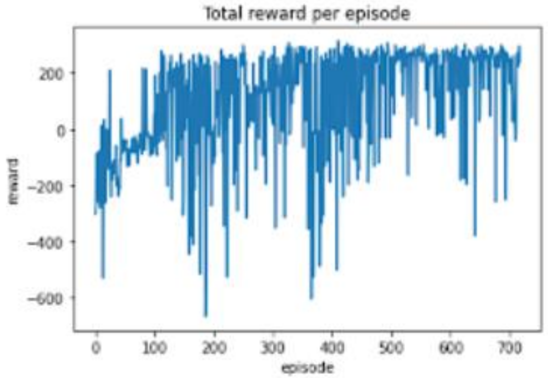
Vanilla DQN Lunar Lander



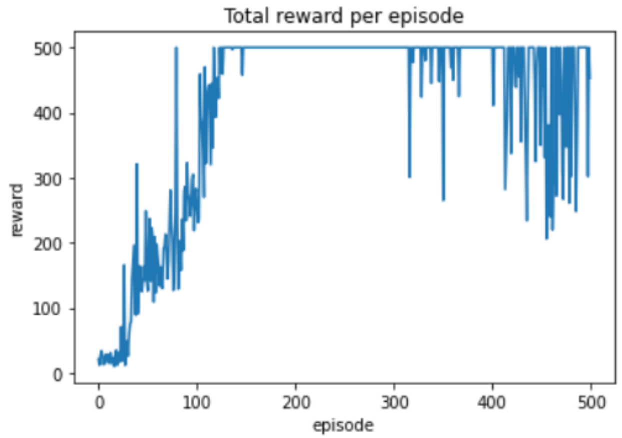
Double DQN LunarLander



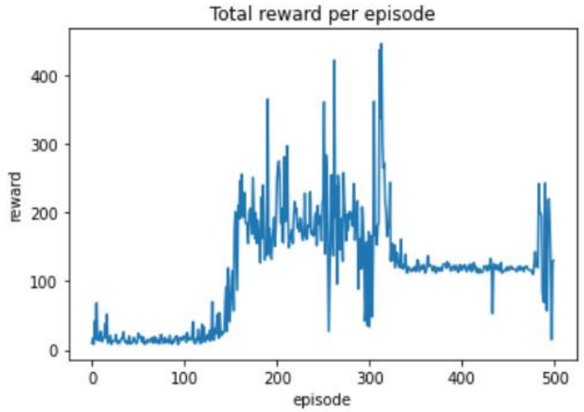
Dueling DQN LunarLander



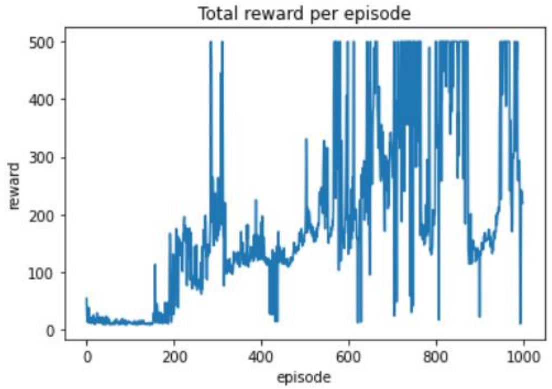
Vanilla DQN CartPole



Double DQN CartPole

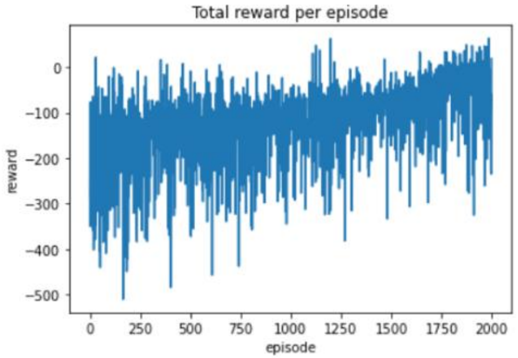


Dueling DQN CartPole

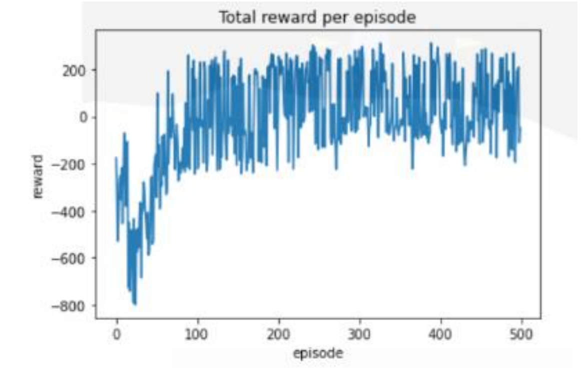


Results for Policy Gradient Methods

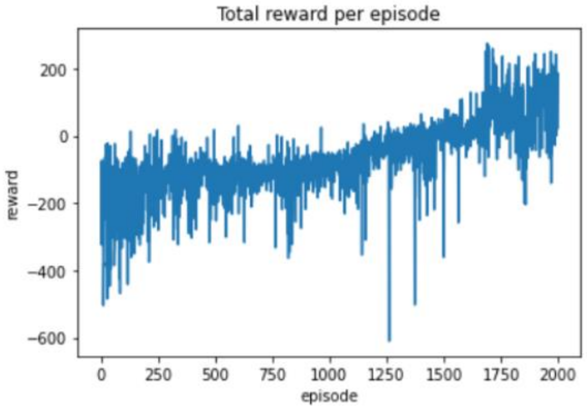
REINFORCE LunarLander



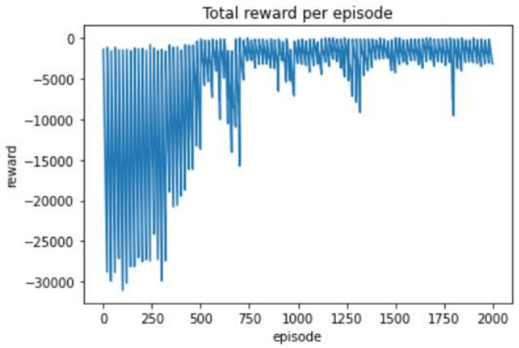
A2C LunarLander



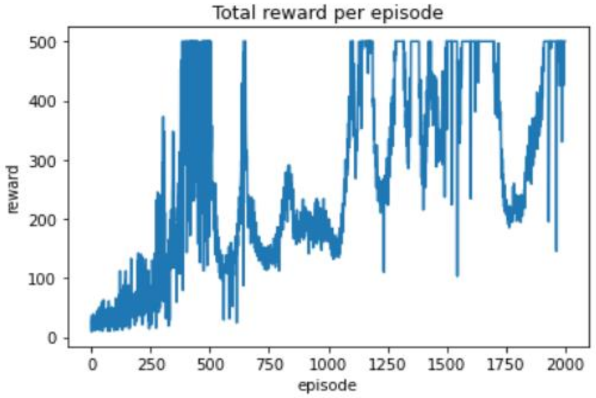
PPO LunarLander



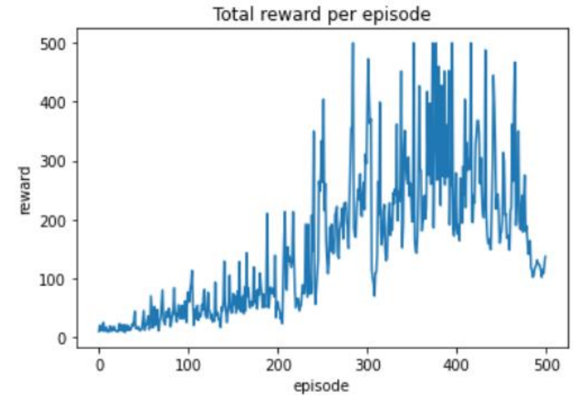
DDPG pendulum



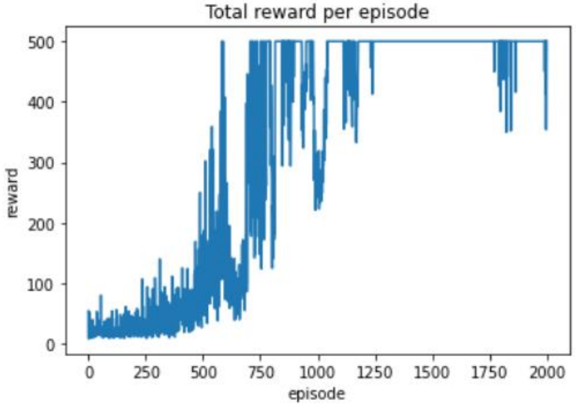
REINFORCE CartPole



A2C CartPole



PPO CartPole

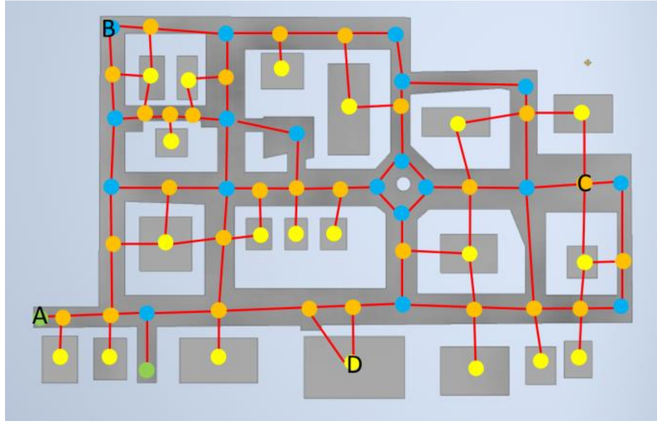


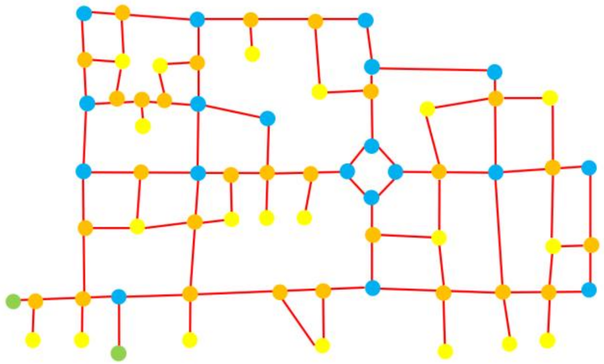
Discussion

- Policy Gradient Out performed Deep Q methods in general
- Deep Q methods often times converged but not to the optimal reward
- PPO performed the best
- From REINFORCE to PPO the stability of the results decreased
- Policy gradient trains much faster
- All algorithms are sufficient for solving these environments
- Hyper parameters could always use improvement (especially in Atari games)

Future work

- Finishing the MARL problem
- Implementing all algorithms with ATARI environments
- Augmenting RL with human behavior





Thank you so much!