Custom Environment RL application

By Sefunmi ASHIRU

Gridworld Environment

Legend:

X : Agent
- : Empty Space
O : Tree

Environment:

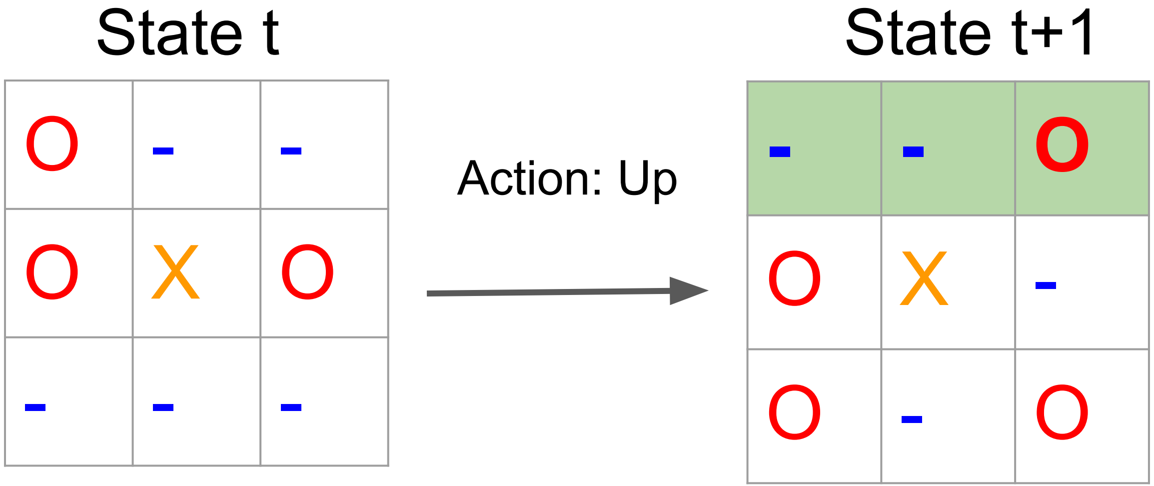
2D

Continuous

Partially Observable

Agent:

Reinforce with Baseline



Policy Gradients

Pros:

- Often easier to approximate than value function
- Works well empirically
- Learns stochastic optimal policies

Cons:

- Training can take a while.
- Converge to local optima although Often there are many optima.
- Unlike human learning: humans can use rapid, abstract model building.

Reinforce with Baseline

1. Initialize the policy parameter θ at random.
2. Generate one trajectory on policy π_θ: $S_1, A_1, R_2, S_2, A_2, \ldots, S_T$.
3. For t=1, 2, … , T:
    1. Estimate the the return G_t;
    2. Update policy parameters: $\theta \leftarrow \theta + \alpha\gamma^t G_t \nabla_\theta \ln \pi_\theta(A_t|S_t)$

Test Implementation

Parameters:

Alpha (Learning Rate)= 0.0005

Gamma (Discount Factor) = 0.99

Environment:

Grid size = 3

Tree Population Density = 0.5

Rewards:

Hitting a tree = -10

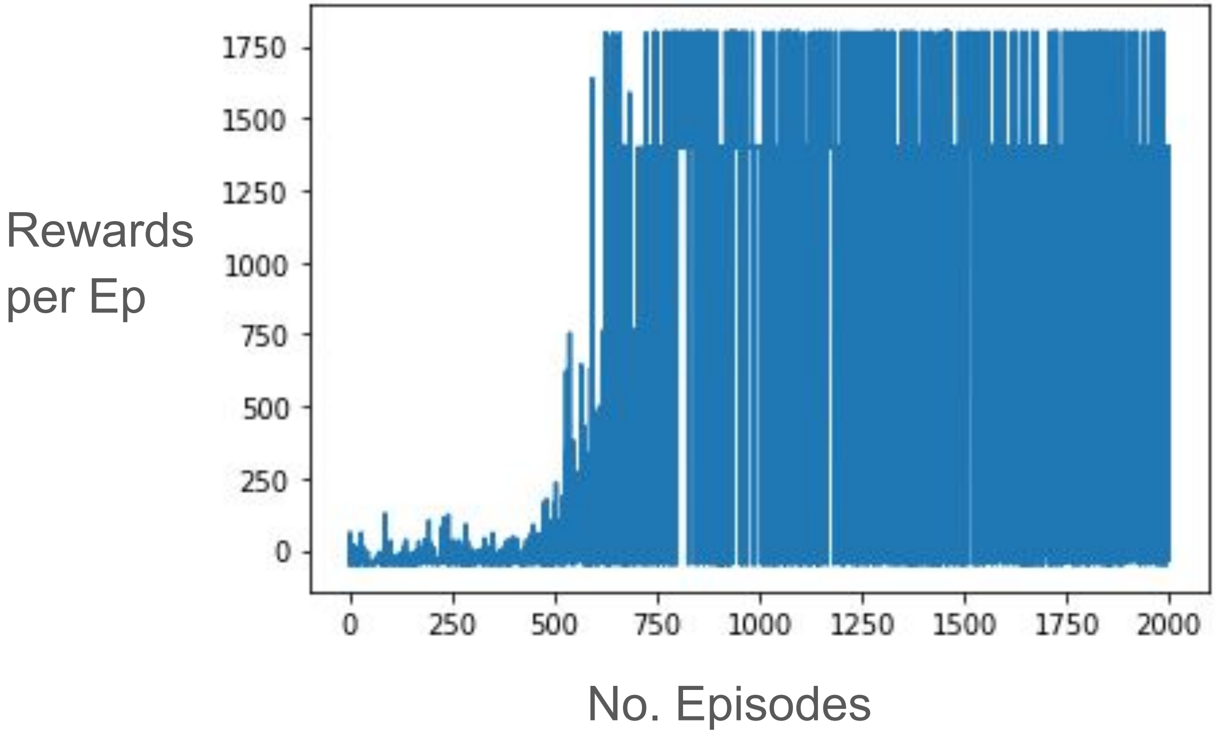
Moving in a pref action = 3

Moving in alt action = 1

Staying alive = 10

-	-	O
O	X	-
O	-	O

Results



Interpretation:

- 400-600 learns how to stay alive in the environment
- Max score on prefered action: Right
- Often took the wrong first step - possibly linked to discount factor

Next Steps

- Set test environment (eg Max Steps, Episodes, Tree Population density, grid-size)
- Experiment with reward shaping
- Experiment with parameters for given method
- Graph steps and score

Citation

Lambert, J., 2020. *Understanding Policy Gradients*. [online] Johnwlambert.github.io. Available at: <https://johnwlambert.github.io/policy-gradients/> [Accessed 20 July 2020].

Weng, L., 2020. *Policy Gradient Algorithms*. [online] Lil'Log. Available at: <https://lilianweng.github.io/lil-log/2018/04/08/policy-gradient-algorithms.html> [Accessed 20 July 2020].