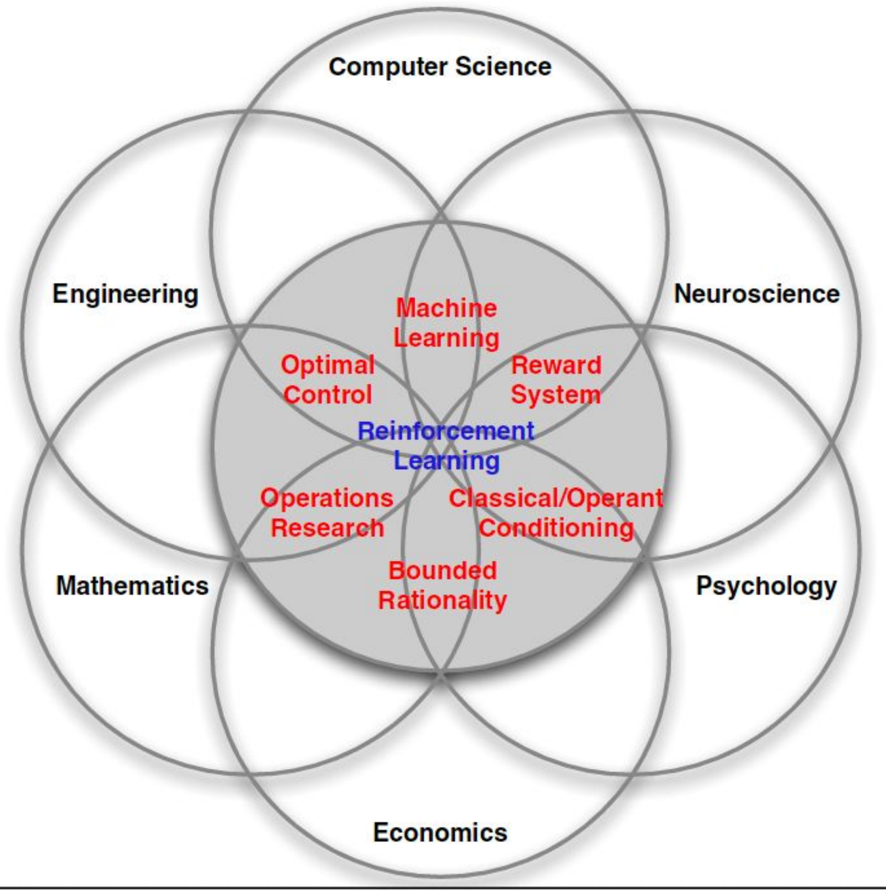
Deep Reinforcement Learning Algorithms on Deterministic Environment

Shashank Bhat and Anirudh Sridhar

CSE 510 Reinforcement Learning (Instructor: Alina Vereshchaka)

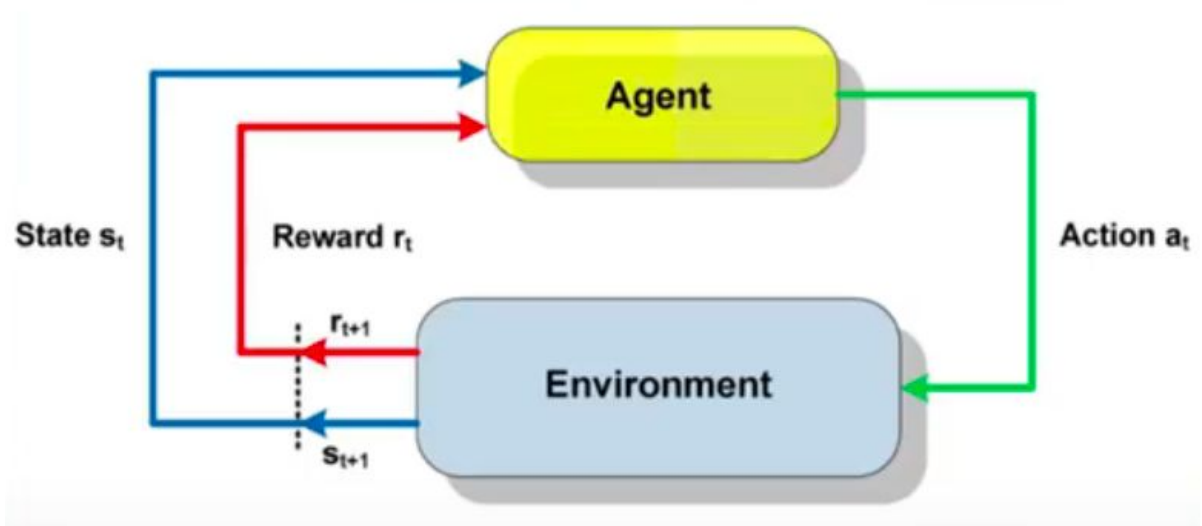
Introduction

Deep RL has gained a lot of success lately in the domains of finance, robotics, multi-agent video games, and text summarization. In this project we are comparing advanced RL algorithms such as DQN, DDQN, Actor-Critic, and PPO on OpenAI Lunar Lander environment.



Components of RL

- Environment, Reward signal and Agent
- The agent further contains agent state, policy, value function (probably), model (optionally).



State Value Function:

$$V(s) = \mathbb{E}_{\pi}[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s]$$

State-Action Value Function:

$$Q(s,a) = \mathbb{E}_{\pi}[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s, A_t = a]$$

Advantage Function:

$$A(s,a) = Q(s,a) - V(s)$$

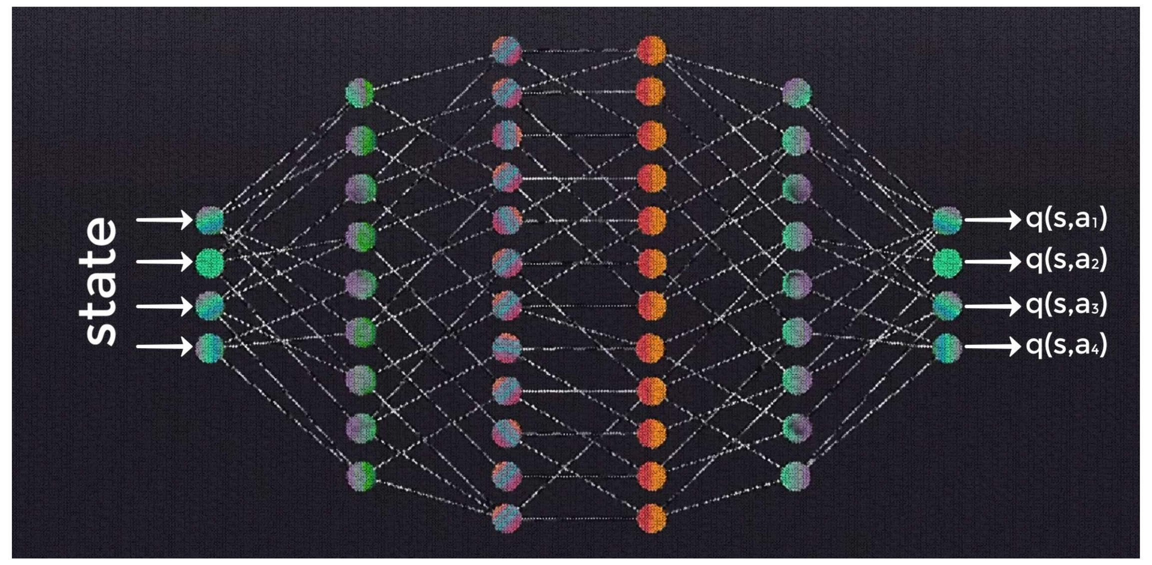
Q-Learning

- Model-free reinforcement learning algorithm
- Goal - learn a policy, which tells an agent what action to take under what circumstances.
- it can handle problems with stochastic transitions and rewards, without requiring adaptations.

$$Q(s,a) \xleftarrow{Q} (s,a) + \alpha(r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s,a))$$

Deep Q Network (DQN)

- Traditional Q Learning requires a large state space depending on the amount of states and actions.
- This is where DQN improves upon where we train a neural network to predict the actions rather than storing them.

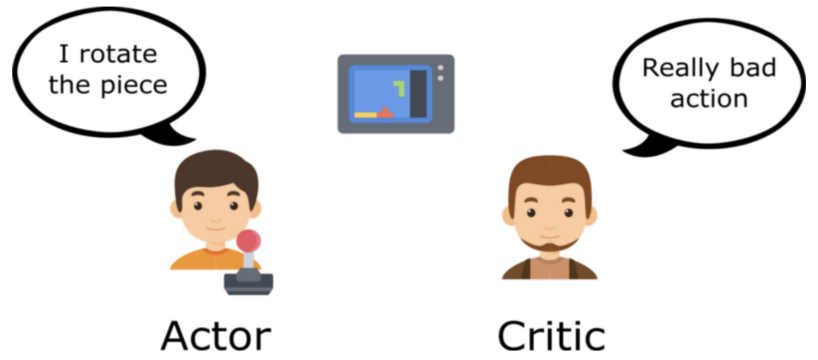


Double DQN (DDQN)

Algorithm 1 Double Q-learning

Initialize Q^A,Q^B,s
repeat
  Choose a, based on $Q^A(s,\cdot)$ and $Q^B(s,\cdot)$, observe r, s'
  Choose (e.g. random) either UPDATE(A) or UPDATE(B)
  if UPDATE(A) **then**
    Define $a^* = \arg\max_a Q^A(s',a)$
    $Q^A(s,a) \leftarrow Q^A(s,a) + \alpha(s,a)\left(r + \gamma Q^B(s',a^*) - Q^A(s,a)\right)$
  else if UPDATE(B) **then**
    Define $b^* = \arg\max_a Q^B(s',a)$
    $Q^B(s,a) \leftarrow Q^B(s,a) + \alpha(s,a)(r + \gamma Q^A(s',b^*) - Q^B(s,a))$
  end if
  $s \leftarrow s'$
until end

Advantage Actor-Critic Methods



Algorithm 1 Q Actor Critic

Initialize parameters s, θ, w and learning rates α_θ, α_w; sample $a \sim \pi_\theta(a|s)$.
for $t = 1 \ldots T$: **do**
  Sample reward $r_t \sim R(s,a)$ and next state $s' \sim P(s'|s,a)$
  Then sample the next action $a' \sim \pi_\theta(a'|s')$
  Update the policy parameters: $\theta \leftarrow \theta + \alpha_\theta Q_w(s,a) \nabla_\theta \log \pi_\theta(a|s)$; Compute the correction (TD error) for action-value at time t:
    $\delta_t = r_t + \gamma Q_w(s',a') - Q_w(s,a)$
  and use it to update the parameters of Q function:
    $w \leftarrow w + \alpha_w \delta_t \nabla_w Q_w(s,a)$
  Move to a $\leftarrow a'$ and s $\leftarrow s'$
end for

Proximal Policy Optimization (PPO)

Algorithm 5 PPO with Clipped Objective

Input: initial policy parameters θ_0, clipping threshold ϵ
for $k = 0, 1, 2, \ldots$ **do**
  Collect set of partial trajectories $\mathcal{D}_k$ on policy $\pi_k = \pi(\theta_k)$
  Estimate advantages $\hat{A}_t^{\pi_k}$ using any advantage estimation algorithm
  Compute policy update

$$\theta_{k+1} = \arg\max_\theta \mathcal{L}_{\theta_k}^{CLIP}(\theta)$$

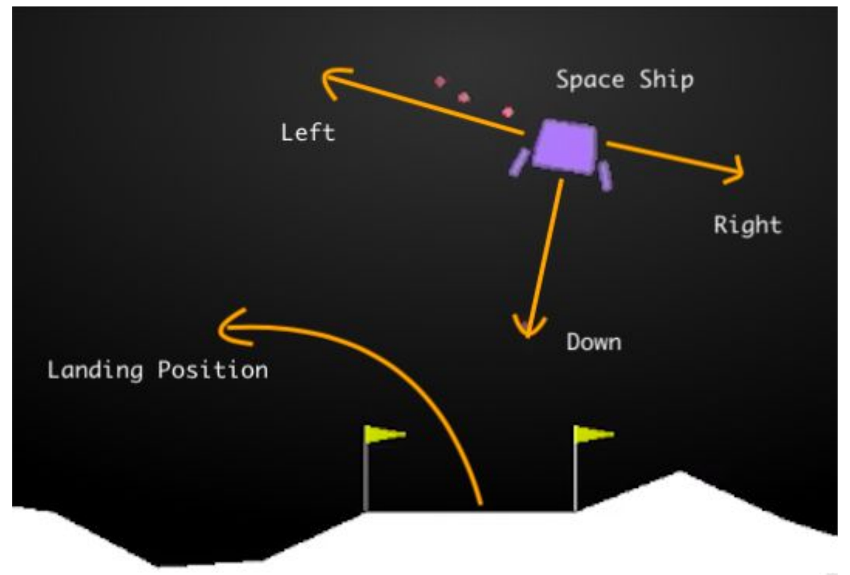
  by taking K steps of minibatch SGD (via Adam), where

$$\mathcal{L}_{\theta_k}^{CLIP}(\theta) = \mathop{\mathrm{E}}_{\tau \sim \pi_k}\left[\sum_{t=0}^{T}\left[\min(r_t(\theta)\hat{A}_t^{\pi_k}, \mathrm{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t^{\pi_k})\right]\right]$$

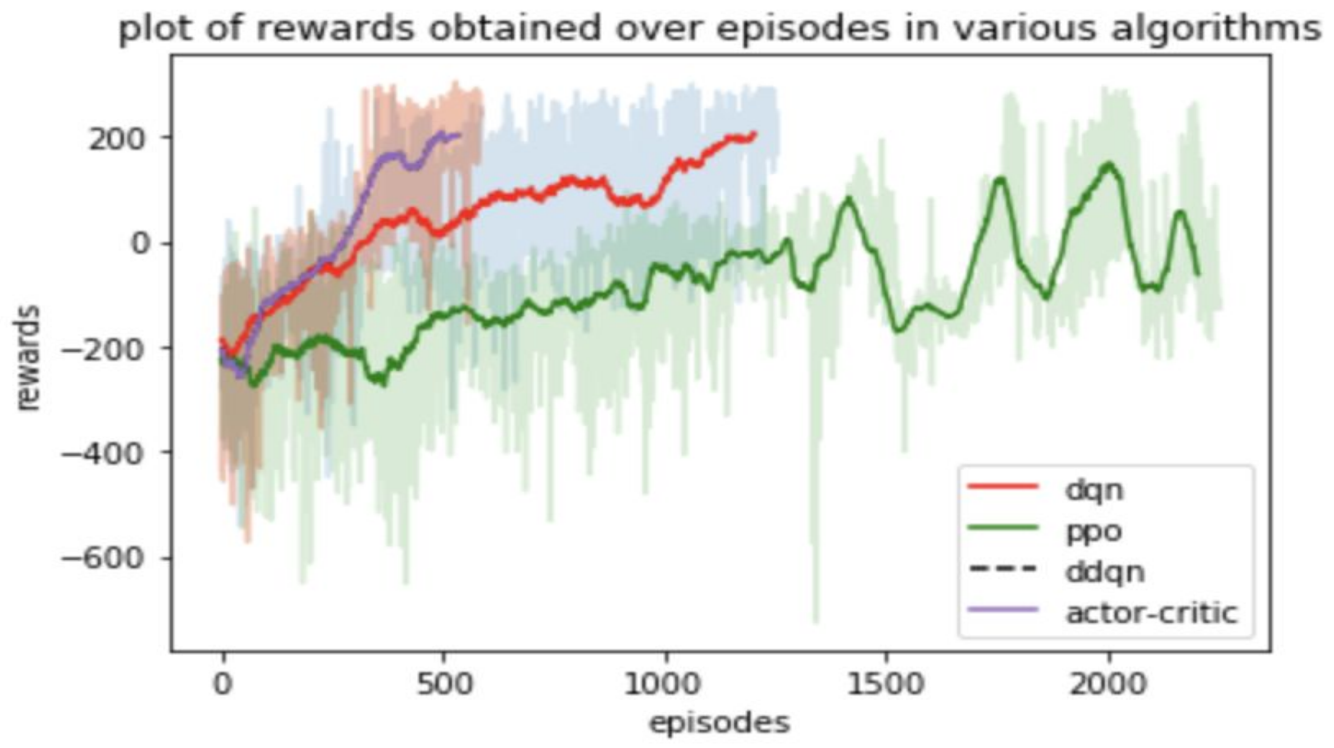
end for

LunarLander Environment

For our experiments we have been working with LunarLander deterministic environment from OpenAI. The environment consists of 4 actions.



Results



Conclusion

The project implemented various Deep RL algorithms and achieved the results as shown in the graph above within the range of 1000 episodes.

References

1. Proximal Policy Optimization by OpenAI
2. CSE510 Reinforcement Learning Lecture Slides

University at Buffalo The State University of New York

Department of Computer Science
University at Buffalo
sbhat4@buffalo.edu, as443@buffalo.edu