Project Report- Group-5

Learn to land Lunar Lander-V2 using DQN and Actor Critic Methods

Professor: Dr.Stas Tiomkin

Group – 5

Akhil Dooliganti(015968238)

Sirkara Mohana Sai Sachin Nekkanti (015900703)

Sudheer Tati (015910674)

Introduction:

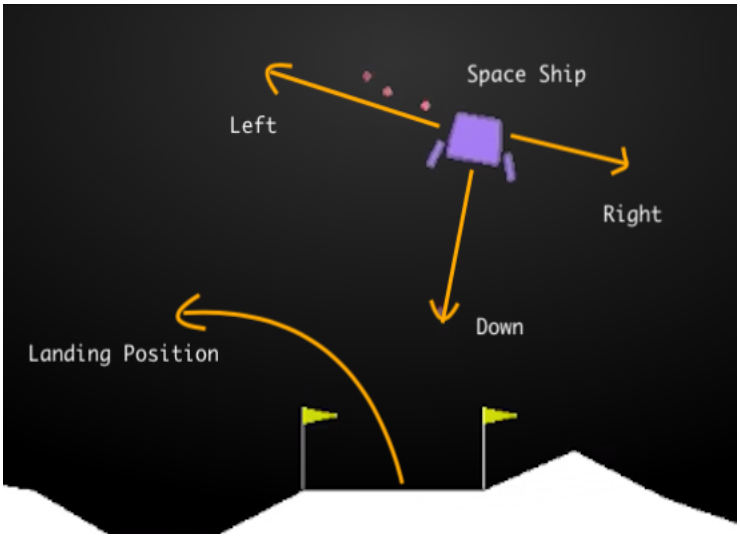
Autonomous drones and rockets have a spectrum of applications from urban transportation and reusable rockets to drug delivery and fugitive methane leak detection. Effective control systems are needed, especially for challenging tasks of landing and takeoff. In this project,we use reinforcement learning, in particular Deep Q-Learning and Actor Critic Algorithm, to train an agent to achieve the goal in the LunarLanderv2 simulation environment from OpenAI Gym .

Project Objective:

The aim is to develop models to train the lunar lander in the landing zone between the flags on OpenAI Gym's LunarLander-v2 environment. In this project, we implement and analyze two different RL techniques, Deep Q-Learning and Actor Critic algorithms.We then perform a comparative analysis of the two techniques to conclude which agent performs better. The problem is considered solved when the average total reward over 100 consecutive episodes is at least 200.

We will see how the agent initially does nothing about how to control and land a rocket, but with time it learns from its mistakes and starts to improve its performance and in the end learns to fully control the rocket and perfectly lands it.

Environment:



State Space:

There are 8 state variables associated with the state space:

1. x coordinate of the lander
2. y coordinate of the lander
3. vx, the horizontal velocity
4. vy, the vertical velocity
5. θ, the orientation in space
6. vθ, the angular velocity
7. Left leg touching the ground (0 or 1)
8. Right leg touching the ground (0 or 1)

Action Space

There are 4 actions:

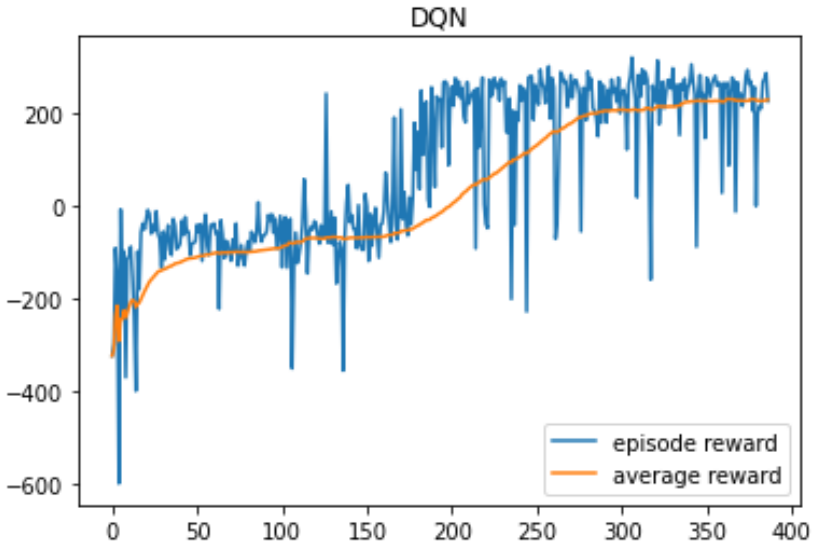
1. 0- Do nothing
2. 1- Fire left engine
3. 2- Fire down engine
4. 3- Fire right engine

Reward Function

1. Vehicle starts from the top of the screen (with random initial velocity) and landing pad is always at coordinates (0,0)
2. Each simulation episode finishes if the lander crashes or comes to rest, receiving additional -100 or +100 points.
3. Each leg ground contact is +10.
4. Firing the main engine is -0.3 points each frame.
5. Firing side engine is -0.03 points each frame.
6. Solving the problem gives 200 points.

DQN Algorithm:

We have used Deep Q-Network to solve this problem. We have used 3 dense layers of neural network(1 Input layer, 1 hidden layer and 1 output layer) for this model with ReLU activation function . We have used Adam optimizer and the loss function is MSE(mean square error). We have run the model for 500 episodes.



DQN training results

Hyperparameters used for this DQN model are:

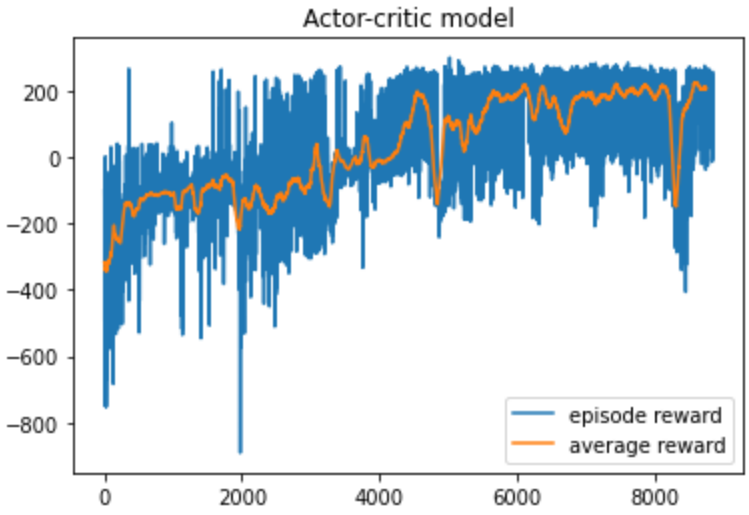
1. Training examples: 1000000(max)
2. Epochs: 500
3. Discount Factor: 0.99
4. Batch size (experience replay): 32
5. Learning rate: 0.00075
6. Initial epsilon value: 1.0
7. Epsilon decay value: 0.001
8. Min epsilon value: 0.1
9. Update rate: 120
10. Weight decay: 0.001

Actor Critic Algorithm

We have also used Actor Critic Algorithm to solve the problem. Actor and Critic, we have used 2 dense layers of neural network(1 Input layer and 1 output layer) for this model with ReLU activation function . We have used the Huber-optimizer and the loss function is MSE(mean square error). We have run the model for 500 episodes.

Loss Function:-

Since a hybrid actor-critic model is used, the chosen loss function is a combination of actor and critic losses for training.The actor loss is based on policy gradients with the critic as a state dependent baseline, the actor (in the case of categorical actions) we sample from a categorical distribution then calculate the loss like we usually do for probabilities, that is using the negative log likelihood, scaled by the advantage.The critic basically tries to apply MSE between TD target and current state value, as we already have the difference on the advantage.



Actor Critic Training results

Hyper Parameters used in this model are:

- Training examples: 1000000(max)
- Epochs: 10000(converge with in 8500)
- Discount Factor: 0.99
- Batch size (experience replay): 32
- Learning rate: 0.01
- Max steps per episode: 10000

Result Analysis/Comparison:

- In General the lunar Lander problem is considered to be solved if average reward is maintained over 200 consistently for 100 iterations. With the DQN Algorithm we are able to stabilize the model with only 500 episodes and with respective Actor Critic we managed to stabilize the model with 8500 episodes.
- The number of episodes taken by Actor Critic was very high when compared to the DQN,but surprisingly the training time was very less for the Actor critic.
- During the testing both models were able to achieve more reward than 200 in each episode.By using DQN the Lunar Lundar was landing quickly with the reason being firing the engine very less,but while we were using the Actor Critic the engine was firing more frequently therefore consuming more time to land.
- The DQN is taking less number of episodes,the reason behind it is that when a model gives a positive reward from there on more frequently positive reward is given by the model which in result helping the model to stabilize very quickly,but with respect to Actor critic the model was consuming more time to stabilize because the rewards are constantly fluctuating from episode to episode.

Difficulties faced:

- Gamma value and learning rate were playing a crucial role in model stabilization,we have challenges finding the correct Gamma Value and Learning rate.
- For training the DQN and Actor Critic it is consuming more time for computation,so we were unable to play around with the Hyper Parameters.
- As a novice in the field of reinforcement learning we found it difficult to implement the loss function for actor-critic and spent significant time figuring it out.
- Actor critic was taking more time when trying to add more hidden layers.

Conclusion

From the above observations we can conclude that both the Actor-Critic and the DQN agents perform well on the original lunar lander problem. However,the DQN model performs better than Actor-Critic Algorithm in terms of time taken(more time taken= more fuel consumption) to land. The Actor-critic also takes more number of episodes to converge than the DQN model.

Task Distribution

1. DQN Modeling - Sudheer, Sachin
2. DQN Training and Hyper parameter search- Sudheer, Akhil
3. Actor Critic Modeling- Sachin, Akhil
4. Actor Critic Training and Hyper parameter search-Sudheer, Sachin
5. Model comparison-Sudheer, Sachin, Akhil

Every one finished the assigned task and no one switched their roles.

References:

1. https://gym.openai.com/envs/LunarLander-v2/
2. https://arxiv.org/pdf/2011.11850.pdf
3. https://github.com/vsaveris/lunar-lander-DQN
4. https://deeplearning.neuromatch.io/projects/ReinforcementLearning/lunar_lander.html
5. https://towardsdatascience.com/policy-gradients-in-a-nutshell-8b72f9743c5d