Deep RL Arm Manipulation

Andrew Truong

Abstract—The project goal was to train a two degree of freedom robotic arm to touch a cylindrical tube. The project used a DQN agent algorithm and Q learning technique to train the robotic arm. Therefore, no dimensional information about the robot nor the tube was needed to train the robot. The robot used reinforcement learning based on a reward structure for any given scenario. In each episode of learning, the robot was given a reward based on the distance from the gripper to the tube object. Scenarios that ended the episode of learning are: ground contact, maximum frames timeout, and robot and object goal collision. Via this reward structure, the robot was trained to accomplish the task with a 0.97 accuracy in challenge 1 and a 0.81 accuracy in challenge 2.

Index Terms—Robot, IEEEtran, Udacity, LaTeX, Reinforced Machine Learning.

1 Introduction

The DQN agent learns based on the reward structure provided. Therefore a logical system is needed for the robot to accomplish desire results. The advantage of deep learning is that the DQN agent can learn how to accomplish a task without prior knowledge about the task. In other words, for this project, inverse and forward kinematics of the robotic arm and the location of the object is not required to train the robot. The DQN agent requires information such as the distance from the robotic arm to the object and a reward structure based on scenarios that would end an episode.

2 Background

Reinforcement learning is an algorithm based on psychology. The task is not predefined with exact constraints, but the robot is rewards based on various tasks that robot performs. The tasks are logged with the corresponding reward in order to base its future action upon. This machine learning algorithm uses a Q learning technique to provide interim rewards based on how close the robot is to the object. This technique improves the efficiency of the algorithm by teaching the robot agent with a lower number of runs to learn the task objective. The advantage of reinforcement learning is that the DQN agent doesn't require prior dimensional analysis of the robot. The disadvantage is that the algorithm can take a large number of runs to complete the desired task with high accuracy. The large numbers of runs can become computationally expensive if the task is not well defined. Machine learning is well suited for complex robotics that have high degrees of freedom and programming of the robot is more time consuming than less accurate than the DQN agent.

3 Results

The results of the project were a 0.97 accuracy for challenge 1 where any part of the robot could collide with the tube object. For challenge 2 where the gripper base needed to collide with the object, the robot earned a 0.81 accuracy. The reward structure defined scenarios that would end each episode to train the robot. If the robot did not reach the task object in 100 frames, the robot was given a reward loss of -50.0. If the robot touched the ground, a reward loss of -100.0 was assigned. A maximum reward win of +100.0 was given if the correct robot part collided with the robot. The scenarios described above ended the episode for the DQN agent. An interim reward was given based on Q learning to give a reward based on how far the robot's gripper base was from the object. The reward structure was smoothed with an alpha parameter to weigh in past episodes. The hyperparameters that were tuned were INPUT WIDTH, INPUT HEIGHT, OPTIMIZER, LEARNING RATE, REPLAY MEMORY, BATCH SIZE, USE LSTM, LSTM SIZE, and EPS DECAY.

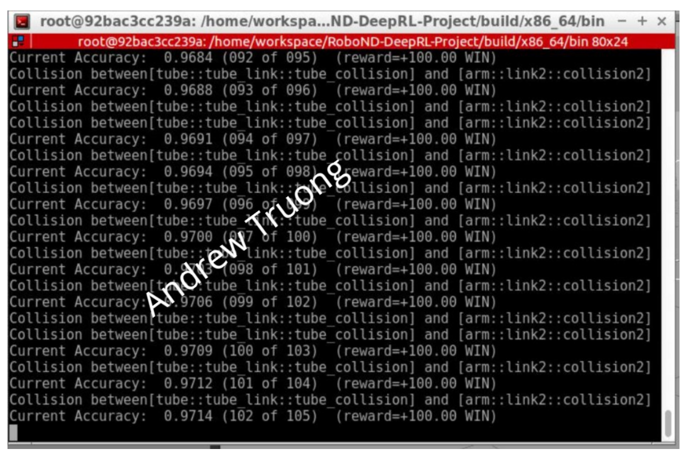

Fig. 1. Challenge 1 Accuracy

4 Discussion

The hyperparameters remained constant from challenge 1 to challenge 2 besides EPS DECAY. The strategy to improve accuracy for challenge 2 was to tune the hyperparameters in challenge 1 for optimum results and then use the same hyperparameter settings for challenge 2. The Input width and height were tuned down for memory requirements and

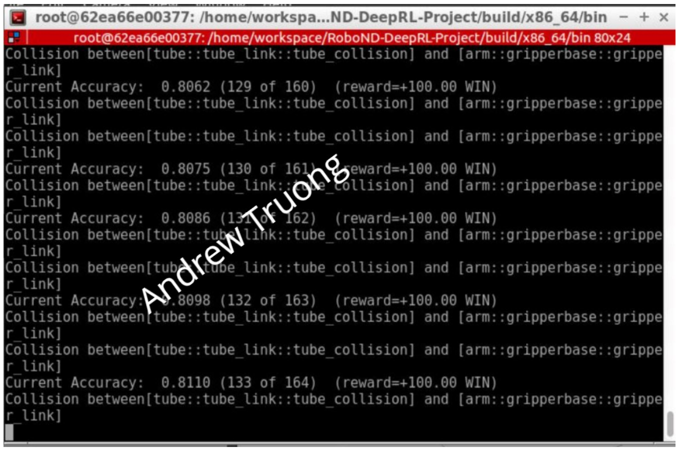

Fig. 2. Challenge 2 Accuracy

did not have effects on the robot performance. The optimizer used was 'Adam'. Results between 'RMSprop' and 'Adam' changed slightly. Hyperparameters that affected the robot performance significantly were learning rate, replay memory, batch size, and lstm size. Increasing the batch size and lstm size improves the accuracy of the robot by going through more iterations through the neural network. Eps decay was used to balance the robot's bias to exploring new tasks when the desired task was already achieved.

5 Conclusion / Future work

The project developed skills using reinforcement machine learning on a robotic simulation in Gazebo. The project practiced skills of subscribe and creating camera and sensor nodes in Gazebo. The implementation of a DQN agent in a robotic application was also taught in this project. The project also taught proper reward structure logic for DQN agents to be successful at learning tasks. Future developments for this project could be another interim reward based on the task goal time to the object, disregarding scenarios where the robot quickly touches the ground. Another future development is the increase in the degrees of freedom to a six axis robot.

- http://colah.github.io/posts/2015-08-Understanding-LSTMs/
- https://artint.info/html/ArtInt_265.html
- https://theses.ubn.ru.nl/bitstream/handle/123456789/5216/Nieuwdorp%2C_T._1.pdf?sequence=1