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

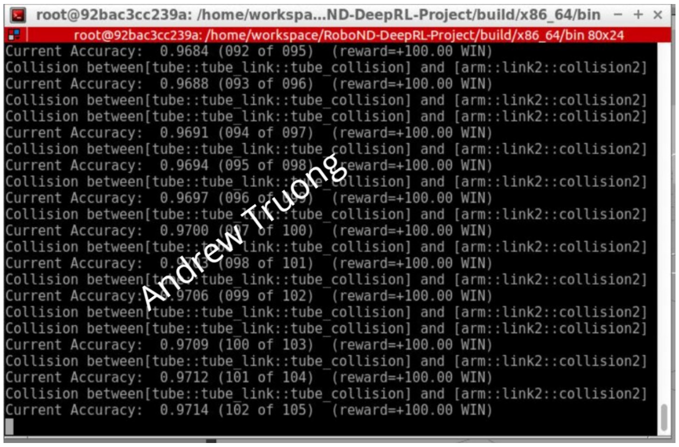
Reinforcement learning is an algorithm based on psychology. The task is not predefined with exact constraints, but the robot is rewards based on various tasks that robot performs. The tasks are logged with the corresponding reward in order to base its future action upon. This machine learning algorithm 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 rewards were scaled to correctly weigh the most significant scenarios. A maximum reward of +100.0 was given, if the robot collided with the tube and -100.0 for colliding with the ground. Interim reward is on the scale of 0.0 to 4.0, so the robot is rewarded for getting closer to the object. The maximum reward wins and loss were optimized to 100.0 and -100.0, because a maximum reward to is too low will encourage the robot to stall next to the object. For example, if the maximum reward is 1.0 then the interim reward will take greater importance than colliding with the object. Therefore, the robot will not learn as effectively as if it were to gain a maximum reward of +100.0 when colliding with the object. The hyperparameters that were tuned were INPUT WIDTH, INPUT HEIGHT, OPTIMIZER, LEARNING RATE, REPLAY MEMORY, BATCH SIZE, USE LSTM, LSTM SIZE, and EPS DECAY. The robot movement parameter is also able to be switched from velocity control to joint control. Joint control was shown to yield greater accuracy than velocity control. Joint control is more accurate, because it is able to move the robot into the full range of position states more efficiently. If velocity control is chosen, than the full range of positions is available. However, the robot will not explore every position in a direct manner, and cause the robot to be sporadic.

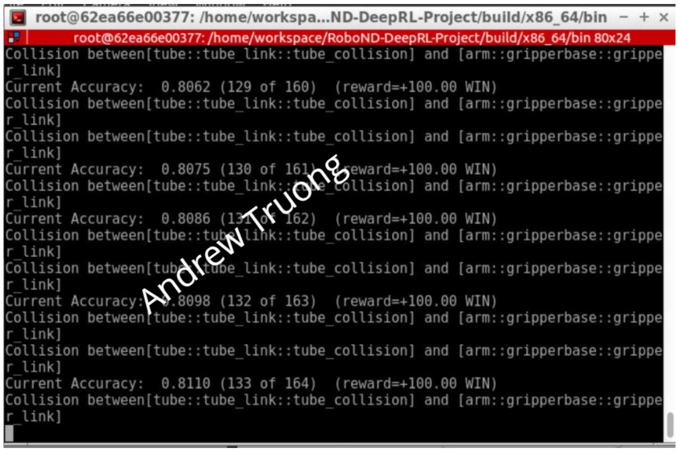
4 Discussion

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


root@92bac3cc239a: /home/workspa...ND-DeepRL-Project/build/x86_64/bin
root@92bac3cc239a: /home/workspace/RoboND-DeepRL-Project/build/x86_64/bin 80x24

```
Current Accuracy:  0.9684 (092 of 095)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::link2::collision2]
Current Accuracy:  0.9688 (093 of 096)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::link2::collision2]
Collision between[tube::tube_link::tube_collision] and [arm::link2::collision2]
Current Accuracy:  0.9691 (094 of 097)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::link2::collision2]
Current Accuracy:  0.9694 (095 of 098)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::link2::collision2]
Current Accuracy:  0.9697 (096 of 099)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::link2::collision2]
Current Accuracy:  0.9700 (097 of 100)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::link2::collision2]
Current Accuracy:  0.9703 (098 of 101)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::link2::collision2]
Current Accuracy:  0.9706 (099 of 102)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::link2::collision2]
Collision between[tube::tube_link::tube_collision] and [arm::link2::collision2]
Current Accuracy:  0.9709 (100 of 103)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::link2::collision2]
Current Accuracy:  0.9712 (101 of 104)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::link2::collision2]
Current Accuracy:  0.9714 (102 of 105)  (reward=+100.00 WIN)
```

Fig. 1. Challenge 1 Accuracy


root@62ea66e00377: /home/workspa...ND-DeepRL-Project/build/x86_64/bin
root@62ea66e00377: /home/workspace/RoboND-DeepRL-Project/build/x86_64/bin 80x24

```
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::grippe
r_link]
Current Accuracy:  0.8062 (129 of 160)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::grippe
r_link]
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::grippe
r_link]
Current Accuracy:  0.8075 (130 of 161)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::grippe
r_link]
Current Accuracy:  0.8086 (131 of 162)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::grippe
r_link]
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::grippe
r_link]
Current Accuracy:  0.8098 (132 of 163)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::grippe
r_link]
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::grippe
r_link]
Current Accuracy:  0.8110 (133 of 164)  (reward=+100.00 WIN)
Collision between[tube::tube_link::tube_collision] and [arm::gripperbase::grippe
r_link]
```

Fig. 2. Challenge 2 Accuracy

and height were tuned down for memory requirements and did not have effects on the robot performance. If the input width and height parameters are not tuned or larger than the memory capabilities, then the program will cease. Therefore an input width equal to 64 give accurate enough data to the DQN agent without overloading the memory. 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. A batch size and lstm size of 256 were chosen. A learning rate = 0.01 was chosen for optimal results. Eps decay was used to balance the robot’s bias to exploring new tasks when the desired task was already achieved. Decreasing the Eps decay to 20 showed significant improvements to the robot learning. The epsilon parameter is used to tune the robot’s bias towards taking new actions. This tradeoff of exploration versus exploitation is the reason why challenge 1 yields greater accuracy than challenge 2. If the robots reaches the goal than the robot will exploit that run and continue to produce analogous joint outputs and results. Since challenge 1 calls for any part of the robot to touch the object, then the robot’s easiest way to accomplish this goal is to move the base joint down towards the object without moving the elbow joint. The robot in challenge 1 may still explore various elbow joint position, but the robot will still collide with the object since the base joint moves completely down. Therefore, the accuracy in challenge 1 is greater than in challenge 2. When moving the gripperbase to the object in challenge 2, the robot has to move both joints. Since there is a greater solution set to accomplishing the challenge 2 the DQN agent must calculate the correlations between all corresponding joint positions. A greater solution set means there is more for the robot to explore. Therefore, the robot will need to explore more joint positions to accurately train the robot to reach the object.

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