Robotics Software Engineer Nanodegree: Deep RL Arm

Lincoln Stein

Abstract

Deep neural networks are an exciting area of research in robotics. One particularly interesting application is the creation of Deep Q Networks to implement reinforcement learning, enabling an agent to learn an effective policy by interacting with a system and receiving feedback based on its performance. Over time, the agent develops a policy that leads to the maximum reward.

Index Terms

Robot, IEEEtran, Udacity, LaTeX, Deep Reinforcement Learning.

1 Introduction

The purpose of this project is to implement an agent, develop rewards functions, and tune hyperparameters to train an agent using deep reinforcement learning to control a robot arm in simulation. The goal is to achieve 90 percent accuracy with any part of the robot arm hitting the target object and 80 percent accuracy with the gripper base hitting the target object.

2 Background

The robot arm is and camera are implemented in the gazebo simulator. In this project, the robot arm is constrained to 2 degrees of freedom, able to actuate 2 joints to move its end effector in a single plane. There is a small cylindrical object that is used as the training target. Joint position control was used, where the agent was able to increase or decrease the angle of the joints in discrete amounts to achieve its goal. The agent is implemented using a deep Q network with LSTM nodes.

2.1 Reward Functions

Several reward functions were implemented to provide the necessary feedback for the agent to learn the desired behavior. If the trial lasted more than 100 frames, the robot would receive a small penalty of -1 to promote efficient motion and end the trial. If the robot gripper collides with the ground, a reward of -2 is given to discourage this behavior to a greater extent, this would also end the trial. At each time step that the agent had not reached an exit condition, it is given a reward based proportional to the smoothed moving average of the change in distance from the gripper to the target. If the change in distance is not above 1 cm, the robot is given a -0.5 penalty to discourage actions that hit the motion limits or result in oscillatory motion. If the robot hits the target with any part of the arm except the gripper base, it receives a reward of 1. If it touches the target with the gripper base, it receives a reward of 5. These values were chosen to promote behavior the more difficult and desirable behaviour appropriately.

2.2 Hyperparameters

The hyper parameters were primarily tuned for purposes of memory management. Reducing the resolution of the image and batch size ensure the network stays within GPU memory limits. The replay memory sets the number of transitions our agent can recall and use for learning.

```
#define INPUT_WIDTH   128
#define INPUT_HEIGHT  128
#define OPTIMIZER "RMSprop"
#define LEARNING_RATE 0.01f
#define REPLAY_MEMORY 10000
#define BATCH_SIZE 8
#define USE_LSTM true
#define LSTM_SIZE 256
```

3 RESULTS

The accuracy of the agent is shown in two graphs. Figure 1 shows the accuracy of the agent achieving contact with any part of the arm and with only the gripper base for each 100 trials (1-100, 101-200, ...). Figure 2 shows the running average accuracy over 200 trials for the agent. On both graphs at about 1300 trials, the agent achieves ¿90 percent accuracy with arm hits and ¿80 percent accuracy with gripper base hits. Further on in the trials, the accuracy actually starts to drop significantly. This could be due to overfitting or seeing previously sparsely explored states so the policy isn't as optimized. The included video shows the agent in action at the end of training. You can see the accuracy over the whole training cycle for any arm hit (the first accuracy value) is at 87 percent and the gripper base accuracy (the second value) is at 69 percent.

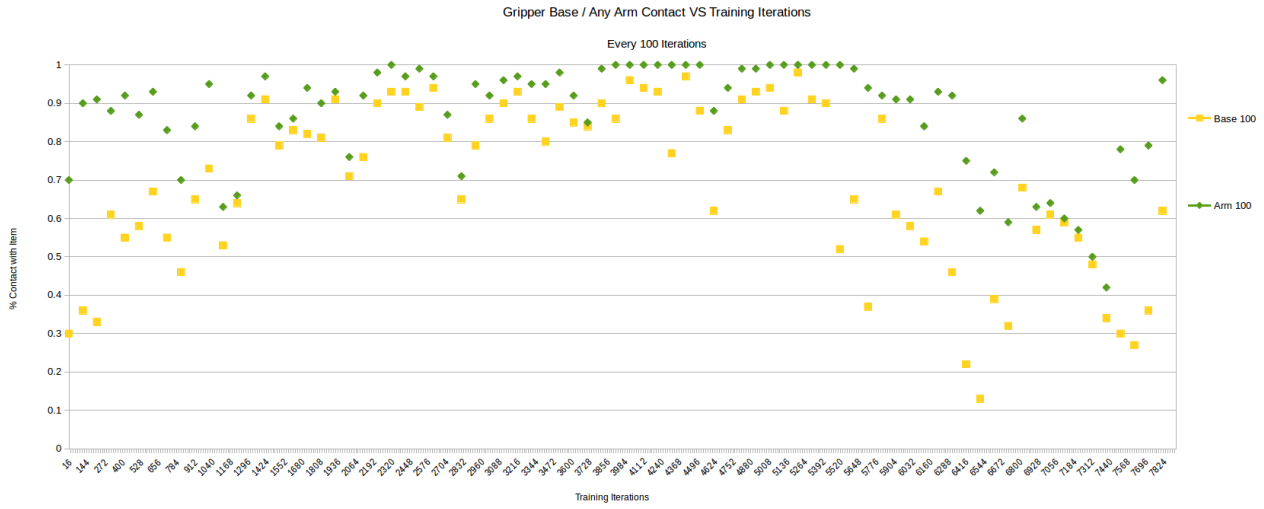

Fig. 1. Accuracy for every 100 trials

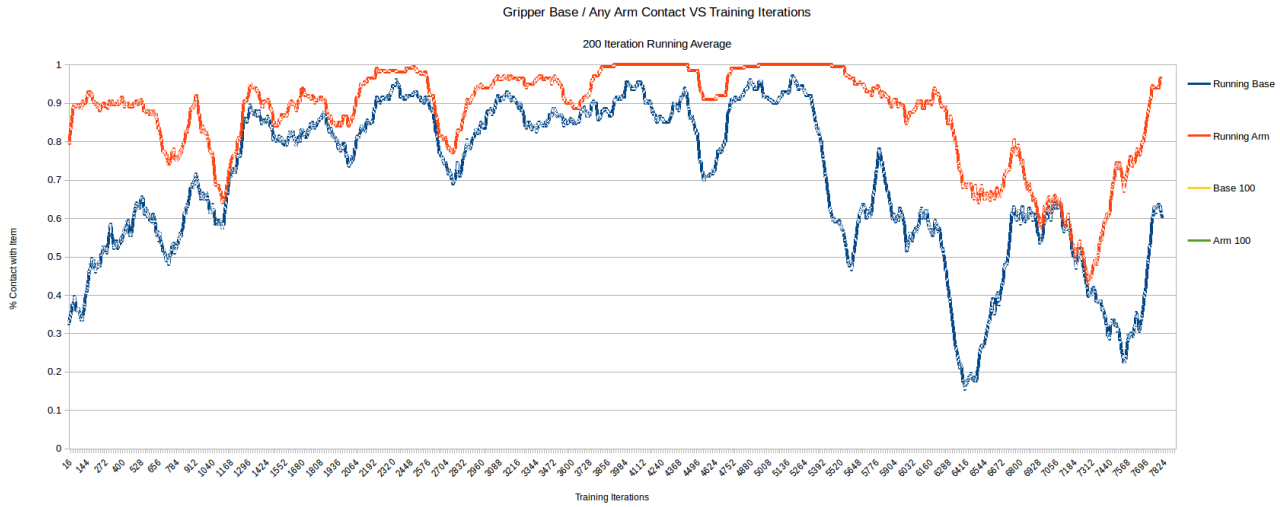

Fig. 2. 200 trial running average accuracy

4 CONCLUSION / FUTURE WORK

Further refinement of the hyperparemeters and continued training could improve the robots performance. Providing more varied initial positions and unlocking the robot base to allow 3D movement would also allow the agent to develop a much more robust and broader policy. The memory parameters were not heavily experimented with so this is an area to be further investigated as well.