CIS 365-01

Artificial Intelligence

Fall 2017

Rodney Fulk

David Fletcher

Michael Chau

**Arcade Learning Environment**

For this project we first wanted to use the Arcade Learning Environment (ALE). It provides an interface to train game playing agents on different Atari games. This environment returns the currently displayed screen as a block of data in RAM for ALE to process and give to a user provides agent environment to attempt to play. This system returns a reward and lets the computer know when the simulation ended. When trying to find pre-created scripts to run with this environment none seemed to be newer than 2015 and due to some of the dependencies being newer it was next to impossible to find working setups. Although ALE is written in C++, we were able to find a python interface that interacts with the ALE framework. Through this python interface we created some simple random agents that played the game. We were not able to really make significant progress with these agents as we had hoped and thus found it a better use of time to find agents and develop them through machine learning.

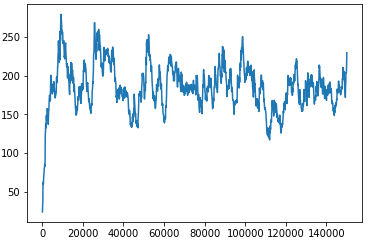


**OpenAI Gym**

Eventually research brought us to the OpenAI Gym project which is a more universal game playing system originally created by Elon Musk. This system does overlay over the top of the ALE system to be able to include Atari games. It comes with a scoring system and a goal to reach for some games to solve the game if you want to upload your results to their web site. We used the Python interface to this system.

We attempted to use different Agents to play Atari’s Space Invaders. A random agent that just randomly picked directions that came with gym ended up with scores between 5 and 550 with an average of 150.6.

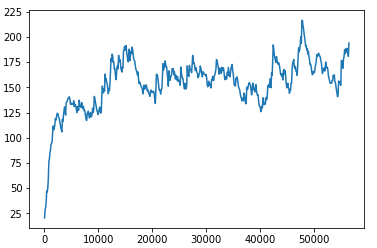
The second agent we attempted to use was apparently part of a ML class. This Deep learning method used the Lasagne Neural Network for N Step Q learning. Not really fully sure how this method works other than it takes the dump of the picture frame and converts it to a setup that Lasagne can understand, uses in this case a Dense Neural Network to store trained actions, it makes a guess based on the results of the neural network then applies an action. Upon the returned response it will adjust its Neural Network accordingly. With this particular agents we ran with 10 parallel agents which allowed it to train on ten games at the same time. We ran using an Nvidia 770 base GPU and completed a total of 150,000 episodes or games without running video. Using the GPU this took 12 and a half hours to train. This particular agent displays a learning graph while training. This graph is not very encouraging:



When the trained model was run on an additional 20 episodes we ended up with a low score of 50, a high score of 590 and an average of 232. This is slightly higher than our random bot and doesn’t quite seem like worth the effort. When trying to rerun this setup with other settings the program errored out so was not able to rerun with different inputs. Seemed to be an issue with something in the system that must have updated while this run was in process because restarting or changing to CPU only did not allow us to rerun this program and reinstallation also did not create a working solution..

**Deep Learning**

Our next model used an Actor Critic deep learning method. Due to issues trying to get the agent to work with a GPU it was run instead off the CPU directly. Still used 10 parallel agents and very similar setup to our previous agent. This agent however had a slightly different layer on it to change the weights within the neural network than the previous method. Due to using the CPU this agent was very slow. I stopped the experiment after 66 hours of training and it only reached 56444 of 150000 episodes.. Since I stopped this agent before it finished we did not receive a summary but the training graph looked like this:



This graph shows this next method seemed to be a little smoother on the average but over all scored much lower than the previous method. This method didn’t seem to realistically give a better score than our random agent since the average score during training appears to be around 150-160. Since it did seem to be trending up and not have the up and down wave of action of the previous agent it is possible that if we trained on more games it may have eventually gotten much better but due to equipment limitations that is only a guess.