Journal Report 4
9/23/19-9/27/19
Pawan Jayakumar
Computer Systems Research Lab
Period 5, White

Daily Log

Monday September 23

I started making improvements on my basic CNN model. I reviewed how convolution and pooling layers work and how to implement them inside keras.

Wednesday September 18

I made more improvements to my CNN model. I learned how to save and load models, view the accuracy and loss graphs, how batch size affects model performance, use cross-validation and batch normalization to prevent over fitting, and how to perform image augmentations such as rotating and translating training images to create a more robust model. The model reached an accuracy of roughly 99.4 percent on training and 98.5 percent on the test set after 10 epochs.

Friday September 20

After giving a demo of the final mnist model, I started working on implementing a deep Q network. One main feature is replay memory. Rather than updating using q values after each step, we save (state,action,reward,nextState) tuples in a deque and randomly sample one to use for updating the model. This stabilizes the training process. I implemented the replay memory and the neural network that will represent the policy. The environment I am training on is openAI gym's cart-pole where the agent learns to balance a pole on a moving cart. Since the environment lets you access information about the orientation of the pole, a CNN is not needed to make decisions–just a dense network.

Timeline

Date	Goal	Met
Today minus 1 weeks	Find a good Hierarchical Reinforcement learning paper to implement	Yes I found one that looks promising, but new ones come out so fast that I may find a better one later
Today	Implement a simple ConvNet in tensorflow on the mnist dataset along with generating the tensor graph to view the accuracy over time	Yes I was able to implement the CNN and see the accuracy over time graph
Today plus 1 week	Implement my own CNN that achieves atleast 99 percent on MNIST	Yes it reached 99.4 percent
Today plus 1 week	Finish implementing the DQN for cartpole	
Today plus 2 weeks	Adapt the DQN to play atari breakout	

Reflection

I ended up using keras for creating training, and running models. I think I will switch over to a lower level of tensorflow when I have to create custom layers or training methods. I like the simplicity of the keras API, so I just went with it to begin with. The DQN is one of the more simpler reinforcement learning models. It is used as the basis for things like the Double DQN, TD3, etc. There is also another branch of reinforcement learning based on policy gradients. I am going to start implementing things like actor critic after working on the DQN because policy based methods are generally more applicable because they can learn stochastic policies, perform better in continuous action spaces, and don't need to learn the reward function.