# Deep Reinforcement Learning

Gabriel Lee

March 4, 2018

# Chapter 1

# Lecture 3: Q Learning

Agent Environment Basically Iterative regression Fitting Q Values to targets computed Camera with Images Nowadays we can make Q function parametrized by neural network but it will involve some craftiness Experience loop -¿ compute target at each step Need terminal stage to converge If state not terminal, target is reward, discounted value of best action Gradient update on function approximator State aliasing/generlization -¿ targets inherently unstable How to solve this problem Experience replay Instead of online updates as they come in Take frame put into replay buffer Learning time -¿ sample transitions from buffer Mini batches of past experience, a lot more stable than online gradient updates Don?t use current weights for neural network, keep and use a copy that?s held the same for some period of time Almost like supervised learning Periodically update Q function by taking latest weights and copying them into target network Send data to server, server periodically send weights back 32 might be fine Need to play with how many frames pass until you save Momentum and target networks Don?t reset them (momentum) Average episode rewards main thing to look at Swuared error

# Chapter 2

# Lecture 4

# Chapter 3

# Lecture 5: Advanced Policy Gradient Methods: Natural Gradient, TRPO, and More