Danny Ly

Captsone Research

Reinforcement Learning, with emphasis on Genetic Algorithms

Learning algos

RMSPROP

Neural Networks

Control each pixel of the frames that are given to it, and outputs a probability to move the controller (agent up or down)

ReLu rule: if input is less than zero N< 0 a =0, otherwise if n > 0 a = n, like a real neuron in your body. F(x) = max(x,0), means it will be 0 vs. X if its less than 0 its 0 if its greater than its x

Non linear activation function,

BackProp to hidden layers is just 1 since you won’t get any squeezing effect on your data

Softmax function:

Basically it ‘squeezes’ the output vector each neurons output between [0,1] then for all output neurons or the output vector a of the entire network it divides the output of each neuron such that the entire a vector will equal 1, giving a **categorical probability distribution**  output meaning its greater for classification.

This also means that the softmax activation function can take any number of classes.

Correctly peaking the ReLu can only handle 2 classes, (sigmoid/ReLu) which could be a problem with classification of multiple class objects

6400X1 input vector for the input vector representing the frame of the game (technically where the ball is)

Binary classification, one of two actions up or down

dL(i)/df(j) = y(I,j) – sigma(f(j))

policy gradients

Discounting, record actions that we take towards the end of the episode more heavily than the first, think about the last action before the one match was over(episode/when computer or ai misses ball) that is more important than the first move