LEARNING IN AUTONOMOUS SYSTEMS

PING PONG GAME (Neeha chowdary)

A ping pong game is where there are two paddles and a ball in between, have to try and get ball pass between the paddles, this game has been build from scratch and train a Neural Network to get it better over time.

Build a Deep Q Network to read it in pixel data from the game of pong .where Deep mind is to read in the pixel data and the score, these two are its inputs to the DQ Network, based on those two it gets better over time, reinforcement learning.

There is one paddle the one which we want to beat, the other paddle would be us or player.

BUILDING A PONG GAME

Starting the game building by importing pygame as this is going to help in making a graphical user interface game in python, random is going to help in defining in which direction the ball is going to go in.

Define some variables for the game like framerate for how fats the game is going to move, the size of the window in which the game is going to take place, size of the paddle which use for playing, size of the ball as it is not completely a circle it’s a tiny small square, speed of the ball and paddle, color spectrum is used to color the paddle, ball and the background.

Initializing the screen using the pygame module and use setmode to display the size. The variable which have been defined abve like the width and height of the window will used for defining the screen. To define the ball draw as it is a small rectangle the two parameters are going to be the X and Y position where the ball must be drawn. The rect function of pygame is used to create the ball rectangle. Using the screen to define it and later color the ball.

Drawing the paddle as the first paddle would be player its is going to use the first Y position as the parameter, using the pygame rect method the paddle buffer as the first parameter then Yposition, width, height are used to tell how big and where its going to begin. Also need to take care that the paddle doesn’t hit the edge of the screen. So as the same for paddle 2.

To update this ball, take parameters as the position of both the paddles and the direction the ball is going to. Checking of a collision also needs to be done, if the ball hits the left side then the learning agent will switch to other direction or the ball is going to switch direction. To change the direction of the ball the score would be negative 1 then to return all that has been calculated.

To return the score both paddles position has to be returned and the ball position as well. Checking if it hits the other direction. Switching the direction if it hits the other side and counting for both sides, ball in X direction is negative 1 and if it doesn’t hit the ball position is greater than equal to 1. Checking for collision and switching the direction happens and also if the ball hits the top it will return the last score.

The direction where the position of the paddle must be updated would be according to going up or down so the action would be an array of where its going. When it moves down the paddle position is movedown and when it is up the position would be movie up, also should take care as it doesn’t move out of the screen. As it is taken care for all the possible scenarios and it would be the same for paddle 2.

Now it is ready to define the pong class which is the game of pong. Saying a random number for the initial direction of the ball to go, define a random number using some some random functions starts from 0 to 9. In order to keep the score tally variable has been defined. Initialize the positions of the paddles and positions of the path. Also defining the ball direction.

A present frame has been defined to get all these right functions which means the current frame. To feed the reinforcement algorithm the pixels, in order to get the pixels from the game. Repaint the window is used to repaint and run anything in it. The background would be black so fill in function is used. Drawing the paddle2 position at Y would be same and as the ball has been drawn before just need to call that function. Initialize at X and Y position.

The ball, paddles and the screen is setup. Taking the pixels from the entire game and returning that by defining a variable called getpixels, image data are used as array 3d functions of pygame, surfaway is also done for this. Updating the window to return the screen, image data.

One more function to get the next frame is given for self, action as the action is like what direction to move in, to define this later the reinforcement learning algorithm is created.

Starting the same way as filling in the background screen and updating the paddle as the previous method action is going to fed back, drawing the paddle and updating the positions of both over time. So as for drawing the ball and get back the surface of the data, then a picked window so pygame .flip and return, to record the tally the scores are going what we had according to self.tally . To return the scores and the image data the above two are used as inputs which are going to be feed in the reinforcement algorithm, the pixels on the screen and the score. The pong class is done.

REINFORCEMENT LEARNING ALGORITHM IN TENSOR FLOW

This begins by importing the tensorflow. Tensorflow is to do machine learning and CV2 is openCV that’s going to help in formatting the pixel data for its better reading into the tensor flow graph, import the pong class which has been created.

Importing numpy would help in math, importing the collection libraries are going to import DQ, it is the data structure which is going to be stored in the experience replay. Memory is going to be stored in Q function.

Hyper parameters are going to start the action variables, learning rate is going to be gamma which would be 0.99. Initial epsilon would be equal to 1.0 this is where the gradient update is done or the training over time and the final epsilon where it need to be would be 0.05. There would be 500k frames for exploring and 50k for observing. Replay is going to be 50k in the batch size is to be 100.

Creating a tensor flow graph. This is a Convolutional neural network, it is a type of 5 layer CNN. In tensor flow to feed the pixel data and the score into this CNN. which is going to help in Deep reinforcement learning and Deep mind, they had given many Atari games which would look at the pixels and become amazing over time.

First layer using tensor flow variables and we are going to define the size of it and to say these sizes can be changed to the any numbers. Defining the first CN layer using TF. Zeros is a functions to define those layers, it doesn’t create a rate of an empty tensor which is good as CNN is going to begin empty and fill it data over time. To define the size of it , a B convolutional TF as bais vector will help in the data flow like in what direction . so, bias will help us in defining like what part of data will flow in it.

The other layers would be followed by the same way and as we created the 5 layers and each of them were well defined the size and the input size, now where does the data needs to go , create an input a placeholder where the data feeds into the network , the pixel data is going to be feed into this. This is where the data is going to flow in.

To the actual activation function which is going to be rectified linear unit, every time to feed data into this at each layer, need to perform some kind of computation need to be done to that data. At every layer we are going to perform the rectified linear unit activation function its going to take to that data and its going to turn into a probability.

Every time the computation is done using the potential flows which are built in the RELU function.

Taking the convolutional DQ network the stride is for how the channels want the data to flow true. It would be a valid variable.

Taking all the data that has been computed in one layer and sending it to the next layer. The last convolutional layer is going to TF and reshape it and also going to take the convolutional layer and say negative 1.

Now the matrix multiplication is been performed , this function to make sure that is working and going to reshape. Now the last layer is going to take everything that was got from before layer to perform the multiplication we need our output tensor. Fc5 would be the output tensor that is going to be our result and we are going to feed that back in the network so return the input tensor and output tensor.

The main method here is to start by creating a session , in the TF session , interact session where this is the method that is used to initialize a session and then to define the input layer and output layer that is we just defined a method to do that which is create graph method, helps is creating out input and output layers for us. Where they are going to feed those in our training graph method which is going to take both of those and its going to run a reinforcement algorithm.

CONCLUSION

We have defined a pong from scratch , defined a bunch of variables for defining how the game is going to look like and draw the ball, both of our paddles, update the position of our ball then update the position of our paddles, where they are going and then we are going to define a class for pong game where we are going to say to get the present frame and the next frame, we are going to take that pixel data and then we are going to feed it into the CNN and deep Q NN , it’s a 5 layer CNN and we built these with Tensorflow , we are going to input that using this placeholder variable that’s where the data flows into the network and once we do then we are going to apply the RELU function in each layer of that network. By the end its going to be using the agent environment loop in the training graph method. We are able to train our network to get better over time.