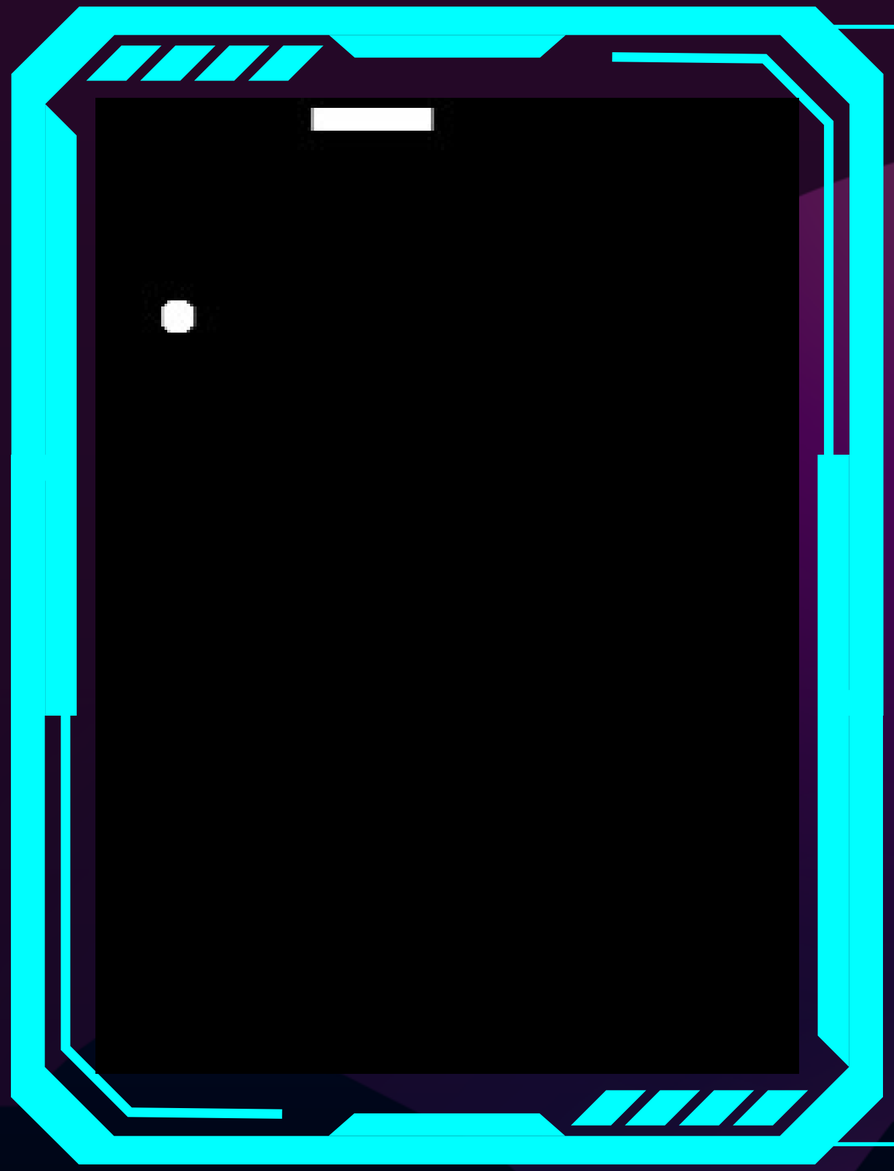
SNAKE AI GAME AGENT

time to eat some berries



FIRST THING FIRST

CAN YOU BEAT AN AI AGENT?

STEP 01

WHAT IS A SNAKE GAME?

The snake game has existed since 1976, and since has evolved with generations to bring more entertainment from a very simple game concept, still being a trend nowadays, in this project we're gonna understand the implementation of an AI Agent that will be able to play the snake game and learn through reinforcement learning how to get higher scores.

STEP 02

REINFORCEMENT LEARNING

Reward: when the snake eats a fruit it gets rewarded with 10 point

Punishment: when the snake hits a wall or bites itself it gets removed 10 points

Maintain: with any other movement that does not provoke a game over or higher the score, it doesn't give or take away any point

STEP 03

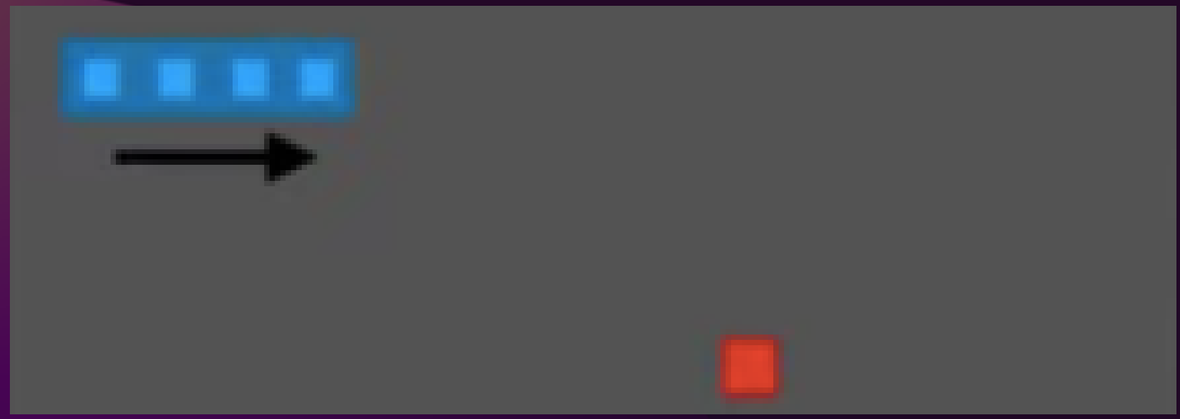
ACTION SYSTEM

- Straight [1,0,0]: the snake will keep the same way its heading and won't change direction
- Right [0,1,0]: the snake will make a right turn depending on the original direction it was taking
- Left [0,0,1]: the snake will a left turn depending on the original direction it was taking

STEP 04

STATUE VALUE

- Danger: straight, right and left
- Direction: left, right, up and down
- Food: left, right, up and down



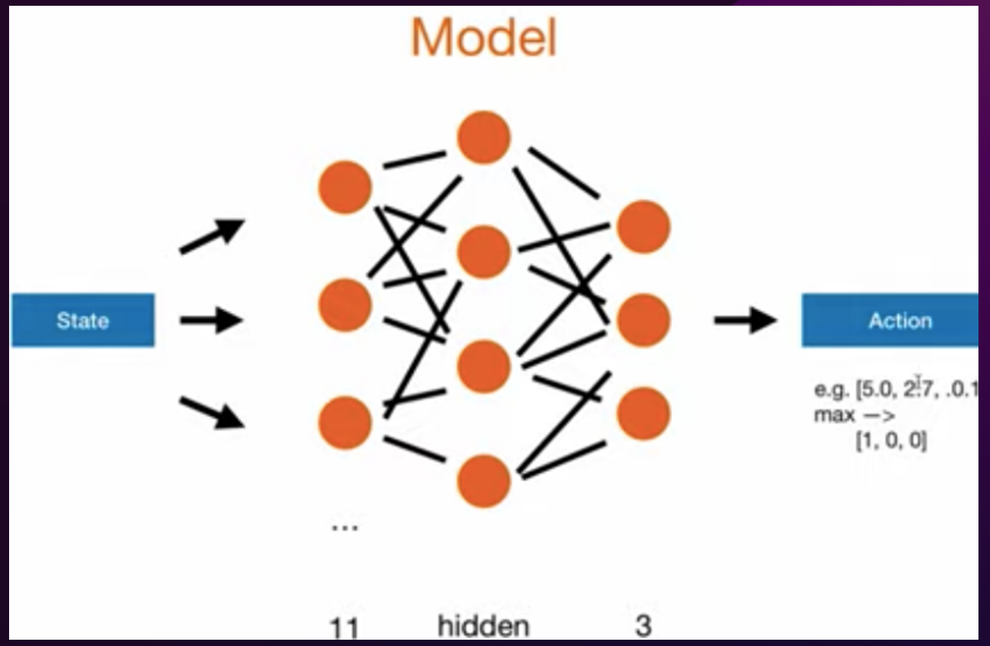
In this case, the state values we would receive to help the snake predict its next movement would be:

[0 , 0 , 0,
 0 , 1 , 0, 0,
 0 , 1 , 0 , 1]

This would mean that there is no immediate danger coming from either left, right or straight, the direction the snake is heading in the right direction inside the environment, and the fruit is to the right and down.

STEP 05

MODEL TRAINING



(Deep) Q Learning

Q Value = Quality of action

0. Init Q Value (= init model)
1. Choose action (model.predict(state))
2. Perform action
3. Measure reward
4. Update Q value (+ train model)

(or random move)

Repeat

Feedforward neural network & Deep Q Learning

STEP 03

$$NewQ(s,a) = Q(s,a) + \alpha[R(s,a) + \gamma max Q'(s',a') - Q(s,a)]$$

New Q value for that state and that action

Current Q value

Learning Rate

Reward for taking that action at that state

Discount rate

Maximum expected future reward **given the new s' and all possible actions at that new state**

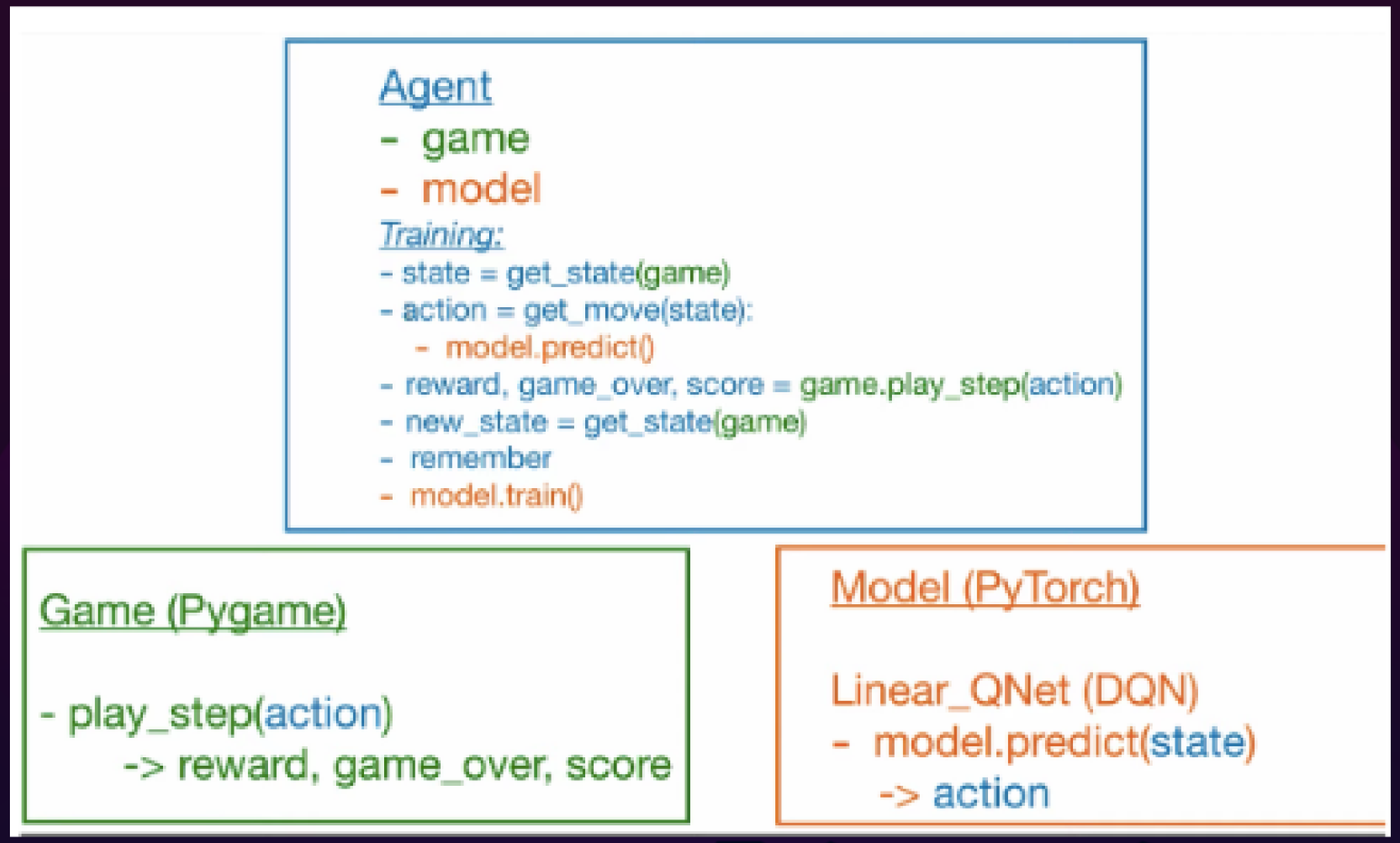
$$Q = model.predict(state_0)$$

$$Q_{new} = R + \gamma \cdot max(Q(state_1))$$

$$loss = (Q_{new} - Q)^2$$

Mean Squared Error

BELLMAN EQUATION & LOSS FUNCTION



SEGMENTATION


THE HIGHEST SCORE FOR THIS AI WAS 64 POINTS