Snake game

Agent:

* Game
  + Use pygame

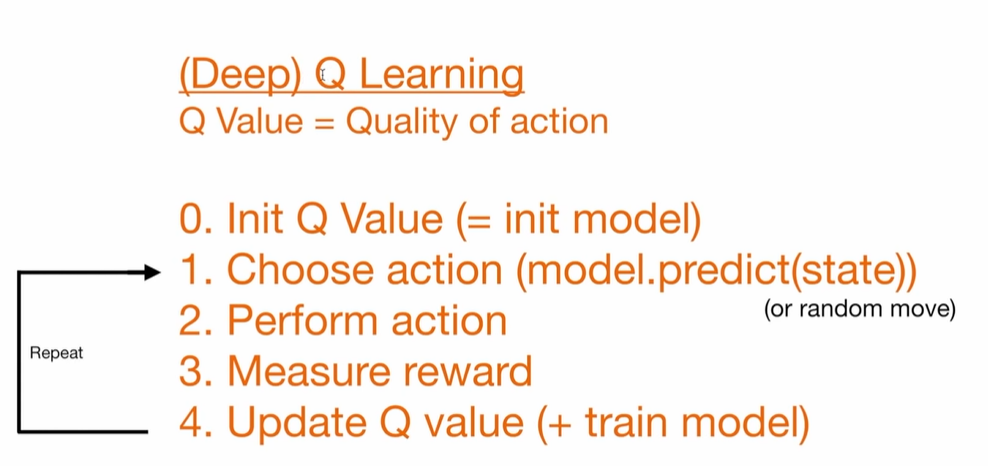
Play\_step(action)

* + - Reward, game\_over, score
* Model (PyTorch)
  + Linear\_QNet (DQN)

Model.predict(state)

* + - Action
* Training:
  + State = get\_state(game)
  + Action = get\_move(state):
    - Model.predict()
  + Reward, game\_over, score = game.play\_step(action)
  + New\_state = get\_state(game)
  + Rêmmber
  + Model.train()
* Reward:
  + Eat food: +10
  + Game over: -10
  + Else: 0
* Action:
  + [1, 0, 0]: Straight
  + [0, 1, 0]: right turn
  + [0, 0, 1]: left turn
* State: tell the snake information about the game that it knows about the environment
  + Have 11 values:
  + Danger straight, danger right, danger left
  + Direction left, direction right, direction up, direction down
  + Food left, food right, food up, food down

(Deep) Q Learning



Bellman Equation

* : new Q value for that state and that action
* : current Q value
* : learning rate
* : reward for taking that action at that state
* : discount rate
* : max expected future reward given the new s’ and all possible actions at that new state

Q Update rule simplified

Loss function

Game

* Conda create -n snake\_game python=3.10
* Conda activate snake\_game