



CNG 462 Artificial Intelligence Term Project Progress Presentation

Doom Bot using Deep
Convolutional Q-Learning
with Eligibility Trace

Ali Gökdemir

2016004



Outline

- Why I choose this
- Q-Learning
- Deep Q-Learning
- Convolutional Neural Network
- Eligibility Trace
- Videos from game
- Training demonstration
- Future work



Why and How?

- An exciting and more hands-on side of Artificial Intelligence
- Wondered how game bots are coded
- Wanted to learn about Neural Networks
- Wanted to learn how an image from the game can be used to train AI

How:

- PyTorch: For Convolutional Neural Network
- Numpy: For two dimensional arrays
- Ppaquette and Open AI: Doom environment

Q-Learning

- A type of reinforcement learning
- Consists of actions and states
- Matrix named “R” and “Q”

The learning rate, i.e. that extent to which new information overrides old information. This is a number between 0 and 1.

The Q function we are updating, based on state s and action a at time t .

The reward earned when transitioning from time t to the next next turn, time $t+1$.

The value of the action that is estimated to return the largest (i.e. maximum) total future reward, based on all possible actions that can be made in the next state.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha [r_{t+1} + \lambda \max_a Q(s_{t+1}, a) - Q(s_t, a_t)]$$

The arrow operator means update the Q function to the left. This is saying, add the stuff to the right (i.e. the difference between the old and the new estimated future reward) to the existing Q value. This is equivalent in programming to $A = A+B$.

The discount rate. Determines how much future rewards are worth, compared to the value of immediate rewards. This is a number between 0 and 1.

The existing estimate of the Q function, (a.k.a. current the action-value).

Pseudocode for Q-Learning

State	Action					
	0	1	2	3	4	5
0	-1	-1	-1	-1	0	-1
1	-1	-1	-1	0	-1	100
2	-1	-1	-1	0	-1	-1
3	-1	0	0	-1	0	-1
4	0	-1	-1	0	-1	100
5	-1	0	-1	-1	0	100

State	Action					
	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0
5	0	0	0	0	0	0

Set the gamma parameter, environment rewards in R

Initialize matrix Q to zero

For each episode:

 Select a random initial state

 do while the goal state hasn't been reached

 Select one among possible actions for the current state

 Using this action, consider going to next state

 Get maximum Q value for this next state based on all possible actions

 Compute $Q(\text{state}, \text{action})$ using the formula

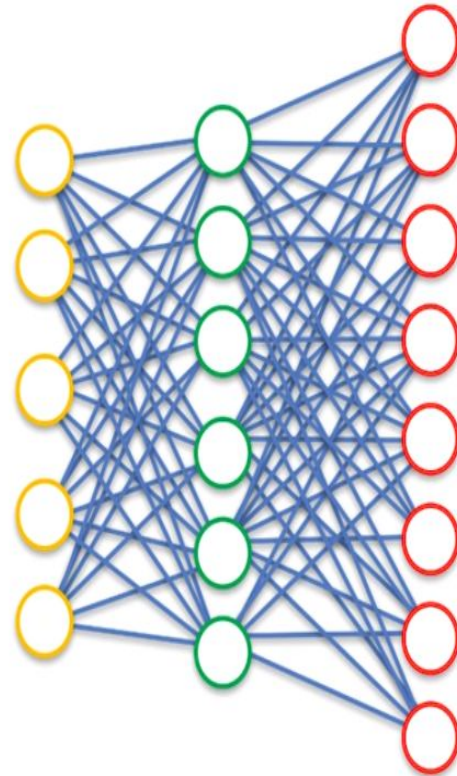
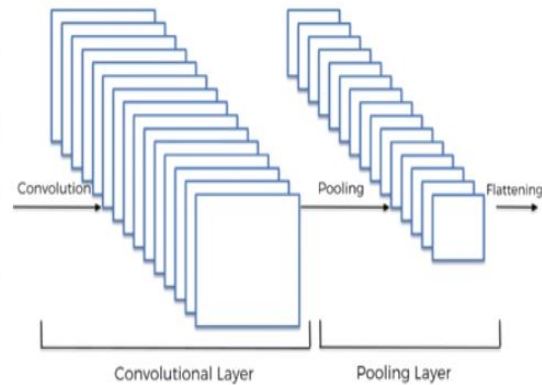
 Set the next state as the current state



Deep Q-Learning

- It is used for problems which have big (state,action) spaces
- Tries to predict the outcome of going to a state rather than manually trying it
- Deals with such big spaces using hidden layers

Convolutional Neural Network





Eligibility Trace

- Without Eligibility trace AI calculates the reward of states after each action
- With eligibility trace the agent takes the steps and then looks back to see which step was eligible to take the agent to its goal or which ones were bad.
- Faster because it doesn't calculate the outcome at each step.



Map Specifications

- Goal
 - Reach the vest
- Rewards
 - Plus distance for getting closer to the vest
 - Minus distance for getting further from the vest
 - Minus 100 pts for getting killed
- Ends when
 - Agent touches vest
 - Player is dead
 - Timeout (1 minute)
- Allowed Actions:
 - Attack, Move Right, Move Left, Move Forward, Turn Right, Turn Left

Videos From Gameplay



Videos From Gameplay



Videos From Gameplay



Videos From Gameplay



Videos From Gameplay



Videos From Gameplay





Training the AI





Future Work

- Training AI for 9 different maps of Doom game
- Trying to enhance the algorithm to learn faster
- Uploading the final version to Github



Thank You
For Listening!