lab14 - report

April 25, 2021

1 Lab14 - st121413

Independent exercise Recall that Mnih et al. (2015) obtained superhuman playing ability for some of the Atari games by combining DQN with experience replay and a more sophisticated state representation that what we've seen so far: they stack successive frames as the input state representation so as to give the agent some "velocity" input rather than a static snapshot of the scene.

Try combining the techniques we've developed in the lab with the frame history as state, and get the best Space Invaders player you can. What's your agent's average and best score over 100 games?

I continue from Prioritized experience replay. From there, I notice that to get the stack of observation, I only need to modify the get_state2 a bit.

First, I transform the observation to 1-channel grayscale. Then, I stack them up in a shape of (queue,width,height). The queue is max to length of 3 and init with all the first frame. Later on, I pop one out and put new observation in.

This way, I can just put all 3 observations as an input without modifying the DDQN

[ ]:

```
image_size = 84
transform = T.Compose([T.ToPILImage(),
                       T.Grayscale(num_output_channels=1),
                       T.Resize((image_size, image_size), interpolation=Image.
 ↪CUBIC),
                       T.ToTensor()])
```

[ ]:

```
import queue
state_buffer = queue.Queue()
def get_state3(observation):

    # First time, repeat the state for 3 times
    if(state_buffer.qsize() == 0):
        for i in range(3):
            state = get_state2(observation)
            state_buffer.put(state)
        # print(observation.shape, state.shape)
    else:
        state_buffer.get()
        state = get_state2(observation)
        state_buffer.put(state)
    # for i in state_buffer.queue:
    #     print(i.shape)
    rep = torch.cat(list(state_buffer.queue), dim=0)
    # print("rep=====",rep.shape)
    return rep

def get_state2(observation):
    state = observation.transpose((2,0,1))
    state = torch.from_numpy(state)
    state = transform(state)
    return state
```

1.1 result

First, I trained the modal for 200,000 epochs and found that the performance is not any better than what we have done so far. - Mainly, the agent only shooting at the start spot. It got 215 at best.

I suspected the number of epoches is to little so I train for 1,000,000 epochs and found a little improvement. - The agent move the ship to the right and shooting from there. Now, it got 285.

Then, I trained more to 50,000,000 epoches. - The agent act the same as the 1M one. But, it able to shoot the purple UFO and got 800 scores at the end. - Unfortunately, without the UFO, the agent got 285 score. The only different from the 1M agent is it wiggles (move slightly left and right) when the laser is about to hit the spaceship.