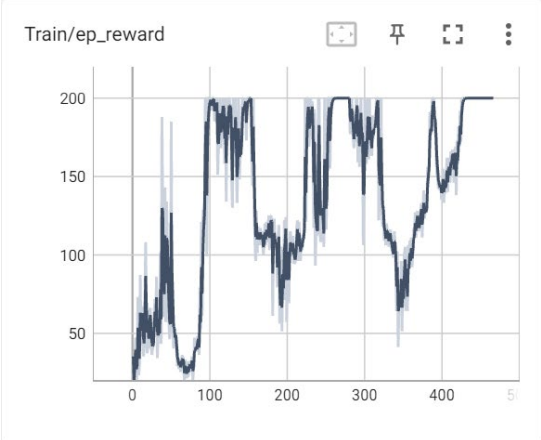
RL HW1 Report

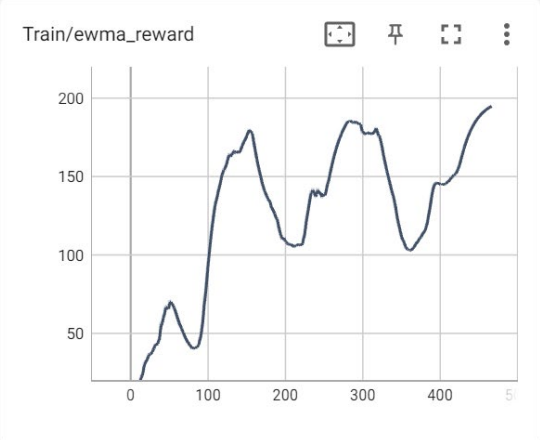
311551059 陳昱丞

(a)Vanilla REINFORCE

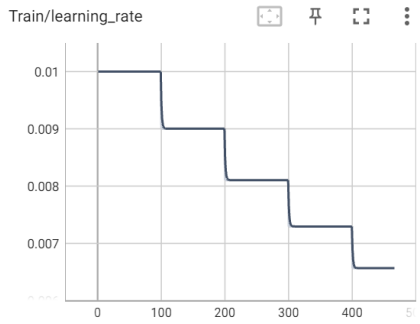
- Results:
  - Task “CartPole-v0” can be solved in episode 466. The episodic reward, ewma reward, and testing results are as follow:

```
Episode 458 length: 199 reward: 200.0    ewma reward: 192.49777592912247
Episode 459 length: 199 reward: 200.0    ewma reward: 192.87288713266634
Episode 460 length: 199 reward: 200.0    ewma reward: 193.229242776033
Episode 461 length: 199 reward: 200.0    ewma reward: 193.56778063723135
Episode 462 length: 199 reward: 200.0    ewma reward: 193.88939160536978
Episode 463 length: 199 reward: 200.0    ewma reward: 194.19492202510128
Episode 464 length: 199 reward: 200.0    ewma reward: 194.4851759238462
Episode 465 length: 199 reward: 200.0    ewma reward: 194.7609171276539
Episode 466 length: 199 reward: 200.0    ewma reward: 195.02287127127119
Solved! Running reward is now 195.02287127127119 and the last episode runs to 199 time steps!
Episode 1   Reward: 200.0
Episode 2   Reward: 200.0
Episode 3   Reward: 200.0
Episode 4   Reward: 200.0
Episode 5   Reward: 200.0
Episode 6   Reward: 200.0
Episode 7   Reward: 200.0
Episode 8   Reward: 200.0
Episode 9   Reward: 200.0
Episode 10  Reward: 200.0
```

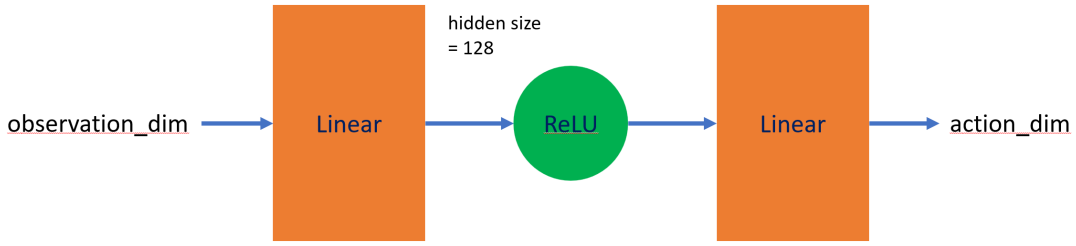




- Learning Rate:



- NN architecture:

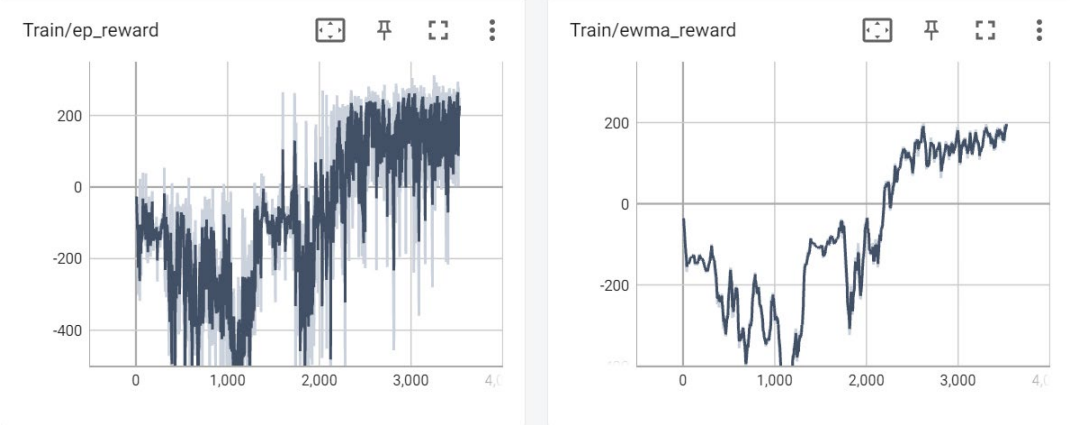


- Other hyperparameters:
  - discount factor gamma: 0.999
  - optimizer: Adam

(b) REINFORCE Baseline

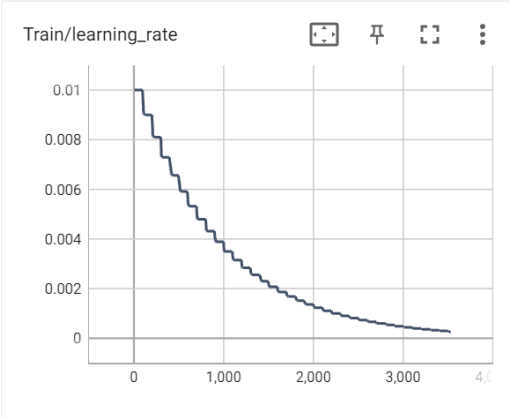
- Results:
  - Task “LunarLander-v2” can be solved in episode 3526. The episodic reward, ewma reward, and testing results are as follow:

```
Episode 3515    length: 320 reward: 277.41201279805455    ewma reward: 183.9011453985534
Episode 3516    length: 301 reward: 308.41666582542854    ewma reward: 190.12692141989717
Episode 3517    length: 325 reward: 291.9197516430611     ewma reward: 195.21656293105534
Episode 3518    length: 323 reward: 208.41908705870378    ewma reward: 195.87668913743775
Episode 3519    length: 460 reward: 194.13316462011227    ewma reward: 195.78951291157145
Episode 3520    length: 796 reward: 206.21527768987838    ewma reward: 196.31080115048678
Episode 3521    length: 550 reward: 206.20422513058634    ewma reward: 196.80547234949177
Episode 3522    length: 140 reward: -20.418626814134115   ewma reward: 185.94426739131046
Episode 3523    length: 375 reward: 267.45916889412837    ewma reward: 190.02001246645136
Episode 3524    length: 492 reward: 251.76190220166237    ewma reward: 193.1071069532119
Episode 3525    length: 300 reward: 292.85712289008904    ewma reward: 198.09460775005576
Episode 3526    length: 423 reward: 249.3591027715831     ewma reward: 200.6578325011321
Solved! Running reward is now 200.6578325011321 and the last episode runs to 423 time steps!
Episode 1   Reward: 242.45115202216672
Episode 2   Reward: 193.5658272519674
Episode 3   Reward: -4.836280858961359
Episode 4   Reward: 252.07850453359362
Episode 5   Reward: 256.6295832635018
Episode 6   Reward: 262.7768337363356
Episode 7   Reward: 198.75673999662587
Episode 8   Reward: 252.87552025188154
Episode 9   Reward: -5.329620886832572
Episode 10  Reward: 14.595839790556113
```

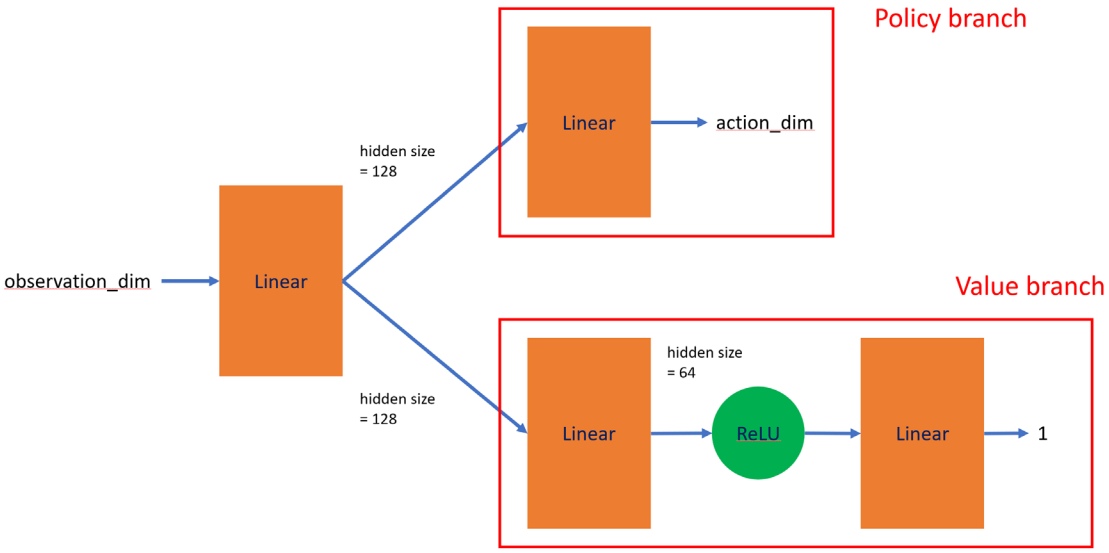


- The design of baseline function:
  - I used the value function for baseline. So for loss calculation, in addition to calculating the policy loss, I had to calculate the value loss. The objective of value function is to let v close to the discounted total reward as much as possible.

- Learning Rate:



- NN architecture:

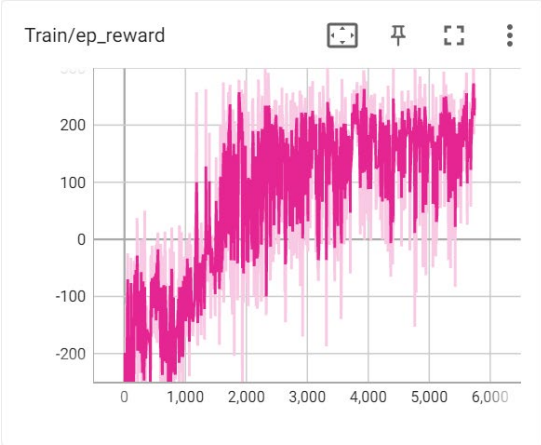


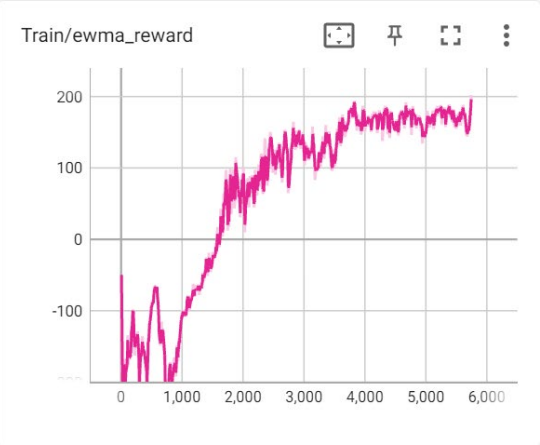
- Other tricks:
  - I performed standardization on the returns because the range of discounted total reward in this case is large. Standardize the returns can make training more easily.
- Other hyperparameters:
  - discount factor gamma: 0.999
  - optimizer: Adam
  - value loss: MSE

(c) REINFORCE GAE

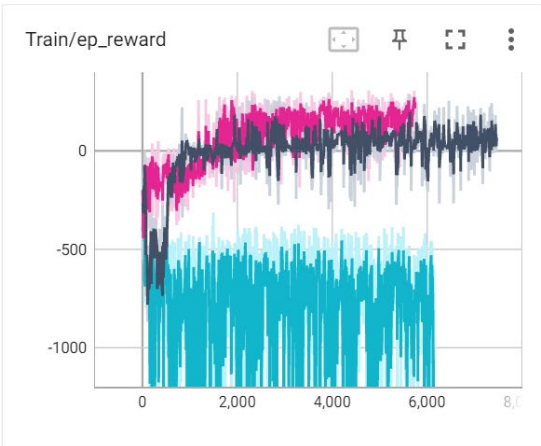
- Results:
  - Task “LunarLander-v2” can be solved in episode 5739. The episodic reward, ewma reward, and testing results are as follow:

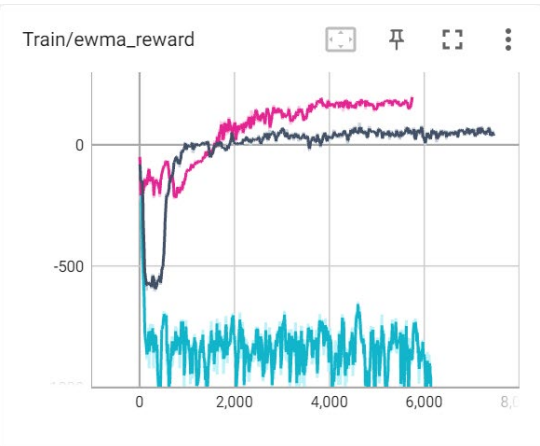
```
Episode 5730    length: 485 reward: 261.44975247050735    ewma reward: 189.37950675916468
Episode 5731    length: 788 reward: 187.57818247874712    ewma reward: 189.28944054514378
Episode 5732    length: 631 reward: 174.77711207786894    ewma reward: 188.56382412178004
Episode 5733    length: 584 reward: 226.34041560395343    ewma reward: 190.45265369588873
Episode 5734    length: 657 reward: 212.4202225374064     ewma reward: 191.5510321379646
Episode 5735    length: 742 reward: 154.6056716052268     ewma reward: 189.7037641113277
Episode 5736    length: 279 reward: 311.756734442982      ewma reward: 195.8064126279104
Episode 5737    length: 435 reward: 243.24626340830528    ewma reward: 198.17840516693013
Episode 5738    length: 696 reward: 200.6772858903267     ewma reward: 198.30334920309997
Episode 5739    length: 373 reward: 272.11424832291596    ewma reward: 201.99389415909076
Solved! Running reward is now 201.99389415909076 and the last episode runs to 373 time steps!
Episode 1   Reward: 182.61867433955527
Episode 2   Reward: 181.1180542881986
Episode 3   Reward: 243.63549859028112
Episode 4   Reward: 29.406034107983686
Episode 5   Reward: 38.67357243330474
Episode 6   Reward: 206.6996748434441
Episode 7   Reward: 247.41143546095847
Episode 8   Reward: 19.694487162202677
Episode 9   Reward: 53.76492446869145
Episode 10  Reward: 182.37281920466725
```





- Three different values of lambda and compare:





- The mapping between colors and values are as follow:
  - Black: lambda=0.99
  - Pink: lambda=0.9
  - Blue: lambda=0.75

    In picture above, we can conclude that 0.9 is the best choice of lambda. It solved this problem in episode 5739. Another feasible choice is 0.99, but the performance looks not as good as 0.9 and failed to solve this problem. And 0.75 is not a good choice since it can't get positive reward and the reward doesn't increase.

- Implementation of GAE:

  The advantage of each state can be derived recursively as follow:

$$A_0^{GAE} = \delta_0 + (\lambda\gamma)\delta_1 + (\lambda\gamma)^2\delta_2 + \ldots + (\lambda\gamma)^{n-1}\delta_{n-1}$$

$$A_1^{GAE} = \delta_1 + (\lambda\gamma)\delta_2 + (\lambda\gamma)^2\delta_3 + \ldots + (\lambda\gamma)^{n-2}\delta_{n-1}$$

  Therefore,

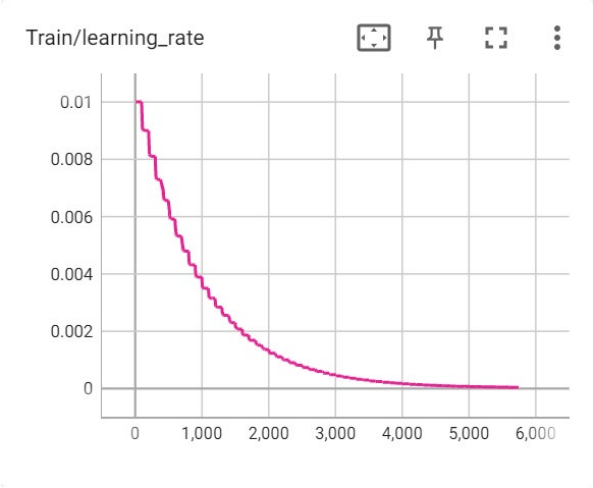
$$A_0^{GAE} = \delta_0 + (\lambda\gamma)A_1^{GAE}$$

  Starting from the last state, we calculate the TD error and store the advantage. The advantage will then be used to calculate next advantage. Doing this iteratively, we can get all advantages in a trajectory. Then, reverse the advantages array to get the right order.

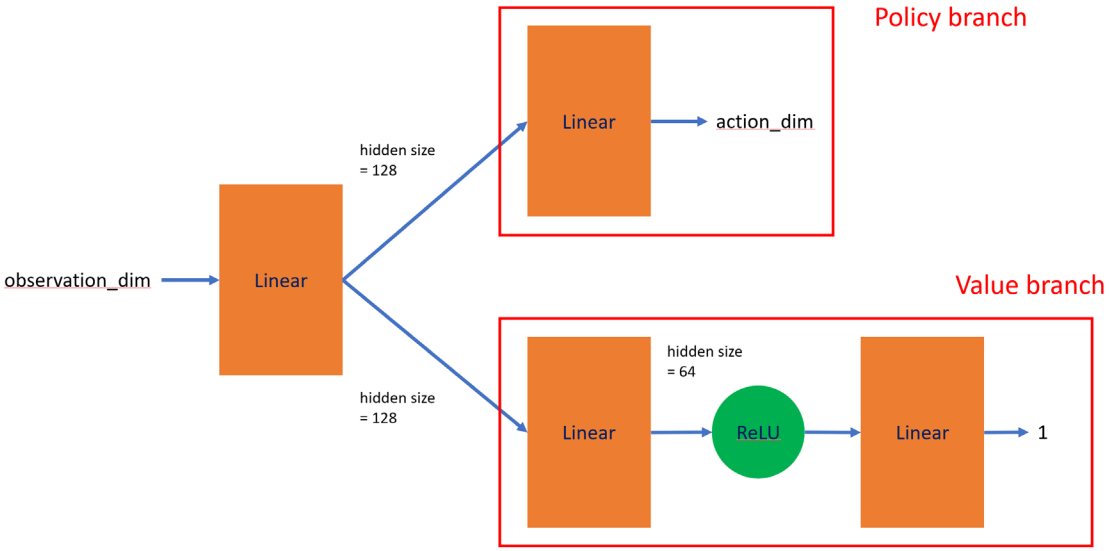
```
########## YOUR CODE HERE (8-15 lines) ##########
advantages = []
advantage = 0
next_value = 0

for r, v in zip(reversed(rewards), reversed(values)):
    td_error = r + next_value * self.gamma - v
    advantage = td_error + advantage * self.gamma * self.lambda_
    next_value = v
    advantages.append(advantage)
advantages.reverse()
return advantages
########## END OF YOUR CODE ##########
```

  Finally, by using the advantage of each state to calculate the policy loss, we get the whole REINFORCE GAE algorithm.

- Learning Rate:



- NN architecture:



- Other hyperparameters:
  - discount factor gamma: 0.999
  - lambda: 0.9
  - optimizer: Adam
  - value loss: MSE