CS 6955 HW5: Policy Gradients

Kutay Eken
u1322888

March 2025

1 Part 1:

1.a

Table 1: Training Progress Over Epochs (Simple Policy Gradient)

Epoch	Loss	Return	Episode Length
0	23.266	26.962	26.962
1	27.220	28.913	28.913
2	30.594	34.819	34.819
3	33.969	37.170	37.170
4	36.189	42.050	42.050
5	36.420	44.088	44.088
6	35.883	47.349	47.349
7	40.790	53.083	53.083
8	40.645	54.677	54.677
9	46.715	58.871	58.871
10	48.606	63.418	63.418
11	44.806	65.221	65.221
12	52.913	70.125	70.125
13	53.855	74.672	74.672
14	61.292	85.034	85.034
15	67.896	86.793	86.793
16	70.780	97.808	97.808
17	58.391	84.148	84.148
18	70.867	100.980	100.980
19	77.955	104.854	104.854
20	95.943	121.405	121.405
21	93.170	148.059	148.059
22	83.233	132.684	132.684
23	95.259	149.412	149.412
24	106.669	165.129	165.129
25	118.681	169.300	169.300
26	97.779	148.206	148.206
27	106.356	159.406	159.406
28	108.416	158.250	158.250
29	152.870	217.542	217.542
30	159.641	239.136	239.136
31	187.827	296.000	296.000
32	195.061	302.778	302.778
33	220.118	317.812	317.812
34	207.313	306.941	306.941
35	225.969	372.000	372.000
36	196.069	317.812	317.812
37	219.938	380.143	380.143
38	207.311	333.062	333.062
39	221.952	363.929	363.929
40	231.619	394.923	394.923
41	245.068	440.833	440.833
42	236.876	411.000	411.000
43	252.776	447.417	447.417
44	255.334	463.727	463.727
45	249.651	461.636	461.636
46	261.270	482.273	482.273
47	259.379	449.167	449.167
48	245.973	456.167	456.167
49	241.055	445.667	445.667

From the results above, it can be seen that the agent is indeed learning, as the average reward increases with each epoch. A similar trend is observed in the episode length, which suggests that the agent is performing better over time—especially relevant for the CartPole task, where longer episodes indicate improved balance.

However, this improvement is not strictly monotonic. The return values fluctuate at certain points. For example, examining epochs 16, 17, and 18 reveals a dip in reward at epoch 17, followed by an increase at epoch 18. This kind of non-monotonicity is expected in reinforcement learning due to factors such as exploration, variations in training batches, and gradient estimates.

Regarding the loss values, we observe that they generally increase on average. This does not imply that the agent is not learning, due to the way loss is defined in our setup. In our case, the loss is given by:

$$\text{Loss} = -(\log\_prob \times \text{reward}).\text{mean}() \tag{1}$$

An increase in this loss indicates that the agent is being trained on more valuable actions, rather than reflecting poor performance.

1.b

From my observations while visualizing the learning policy over time, I found that the agent's actions are not meaningful during the early epochs. These actions are also quite abrupt, causing the pole to lose balance quickly. However, as training progresses, the agent's behavior becomes more purposeful—for example, attempting to counterbalance the pole by moving in the direction it is leaning. Additionally, the agent's movements become smoother, resulting in improved performance. Overall, the agent's actions become increasingly meaningful and refined as training continues.

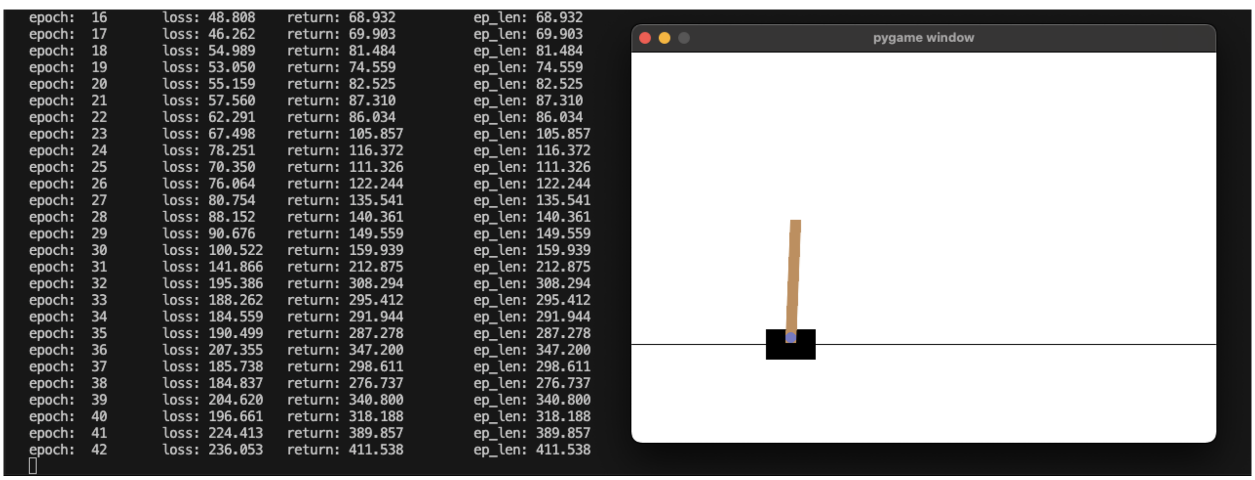

Figure 1: CartPole Visualization.

2 Part 2:

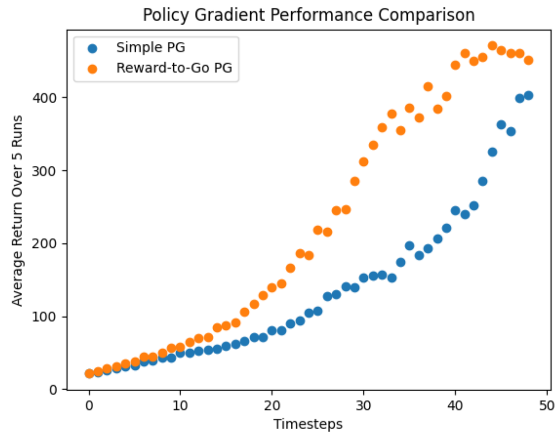

Figure 2: Policy Gradient Performance Comparison

Based on my observations and the plot above, it can be concluded that the "reward-to-go" approach learned more effectively and converged faster than the "simple" policy gradient. From the graph, it is clear that it reaches higher scores more quickly. I also noticed this visually: the agent using the reward-to-go method was able to balance the pole for much longer and achieved high scores, even reaching 500, which the simple approach never managed to achieve.

The reason for this is the nature of the reward-to-go policy gradient. Compared to the simple policy gradient, the reward-to-go policy assigns more realistic credit to each action by ignoring unnecessary ones that do not affect the current outcome. This makes the feedback more accurate and fair, leading to faster convergence and better performance.

3 Part 3:

I chose the `InvertedPendulum-v5` environment from Gymnasium to test my implementation on a continuous action space. Initially, I ran the training without modifying any parameters and obtained the following results:

```
epoch:   0      loss: 10.392    return: 9.066   ep_len: 10.066
epoch:   1      loss: 12.634    return: 10.802          ep_len: 11.802
epoch:   2      loss: 14.348    return: 12.462          ep_len: 13.462
epoch:   3      loss: 15.737    return: 14.350          ep_len: 15.350
epoch:   4      loss: 18.219    return: 16.396          ep_len: 17.396
epoch:   5      loss: 19.923    return: 18.277          ep_len: 19.277
epoch:   6      loss: 20.120    return: 19.445          ep_len: 20.445
epoch:   7      loss: 22.850    return: 22.227          ep_len: 23.227
epoch:   8      loss: 24.804    return: 24.607          ep_len: 25.607
epoch:   9      loss: 27.101    return: 27.787          ep_len: 28.787
epoch:  10      loss: 28.086    return: 29.321          ep_len: 30.321
epoch:  11      loss: 25.765    return: 28.128          ep_len: 29.128
epoch:  12      loss: 27.155    return: 30.217          ep_len: 31.217
epoch:  13      loss: 29.730    return: 32.818          ep_len: 33.818
epoch:  14      loss: 30.252    return: 33.799          ep_len: 34.799
epoch:  15      loss: 31.836    return: 36.178          ep_len: 37.178
epoch:  16      loss: 32.272    return: 36.924          ep_len: 37.924
epoch:  17      loss: 30.396    return: 36.433          ep_len: 37.433
epoch:  18      loss: 35.614    return: 39.797          ep_len: 40.797
epoch:  19      loss: 36.092    return: 43.096          ep_len: 44.096
epoch:  20      loss: 39.874    return: 46.781          ep_len: 47.781
epoch:  21      loss: 40.069    return: 46.733          ep_len: 47.733
epoch:  22      loss: 39.422    return: 46.299          ep_len: 47.299
epoch:  23      loss: 40.311    return: 49.828          ep_len: 50.828
epoch:  24      loss: 39.411    return: 47.592          ep_len: 48.592
epoch:  25      loss: 41.725    return: 49.576          ep_len: 50.576
epoch:  26      loss: 45.408    return: 53.446          ep_len: 54.446
epoch:  27      loss: 49.159    return: 57.488          ep_len: 58.488
epoch:  28      loss: 49.319    return: 58.571          ep_len: 59.571
epoch:  29      loss: 49.364    return: 58.857          ep_len: 59.857
epoch:  30      loss: 46.512    return: 55.708          ep_len: 56.708
epoch:  31      loss: 51.308    return: 60.765          ep_len: 61.765
epoch:  32      loss: 50.085    return: 59.036          ep_len: 60.036
epoch:  33      loss: 53.808    return: 63.538          ep_len: 64.538
epoch:  34      loss: 55.878    return: 66.280          ep_len: 67.280
epoch:  35      loss: 62.902    return: 74.833          ep_len: 75.833
epoch:  36      loss: 59.217    return: 69.083          ep_len: 70.083
epoch:  37      loss: 66.233    return: 78.766          ep_len: 79.766
epoch:  38      loss: 68.999    return: 80.032          ep_len: 81.032
epoch:  39      loss: 69.333    return: 79.048          ep_len: 80.048
epoch:  40      loss: 77.750    return: 90.786          ep_len: 91.786
epoch:  41      loss: 75.495    return: 86.982          ep_len: 87.982
epoch:  42      loss: 70.087    return: 83.817          ep_len: 84.817
epoch:  43      loss: 75.969    return: 91.704          ep_len: 92.704
epoch:  44      loss: 83.023    return: 98.431          ep_len: 99.431
epoch:  45      loss: 85.789    return: 102.449         ep_len: 103.449
epoch:  46      loss: 88.062    return: 108.326         ep_len: 109.326
epoch:  47      loss: 86.701    return: 106.340         ep_len: 107.340
epoch:  48      loss: 82.386    return: 102.040         ep_len: 103.040
epoch:  49      loss: 84.624    return: 101.720         ep_len: 102.720
```

Figure 3: Inverted Pendulum Training Results

From Figure 3, it can be observed that the agent learns over time, achieving higher rewards as training progresses. This behavior is consistent with the trend observed in **Part 1**, where the reward values do not increase monotonically but demonstrate an overall upward trajectory across epochs.

To investigate whether the agent could reach even higher performance, I increased the number of training epochs from 50 to 100.

With this single change, I observed a significant improvement: starting from epoch 92, the agent consistently achieved a reward of **1000**, which is the maximum possible return for the *Inverted Pendulum* environment.