ECE 276C Assignment 3 Report

Mingwei Xu A53270271

Fall 2019

1 Question 1 - Cartpole-v1

1.1 Vanilla Reinforce Algorithm

Using the vanilla reinforce algorithm, which updates gradient by:

$$\nabla J(\theta) \approx \frac{1}{n}\sum_{i=0}^{n} G(\tau)\left(\sum_{t=0}^{T}\nabla_\theta \log \pi_\theta\left(a_t|s_t\right)\right) \text{ ,where } G(\tau)=\sum_{t=0}^{T}\gamma^t r(t) \tag{1}$$

Here we use a 3 layer fully connected neural network with a 48-node hidden layer to estimate policy. The optimizer we use is Adam.

Set batch size of 500 steps for 200 iterations, using learning rate $\alpha = 0.01$ and discount factor $\gamma = 0.99$, we obtained the result shown in "Fig. 1": (Average returns in figure are not discounted)

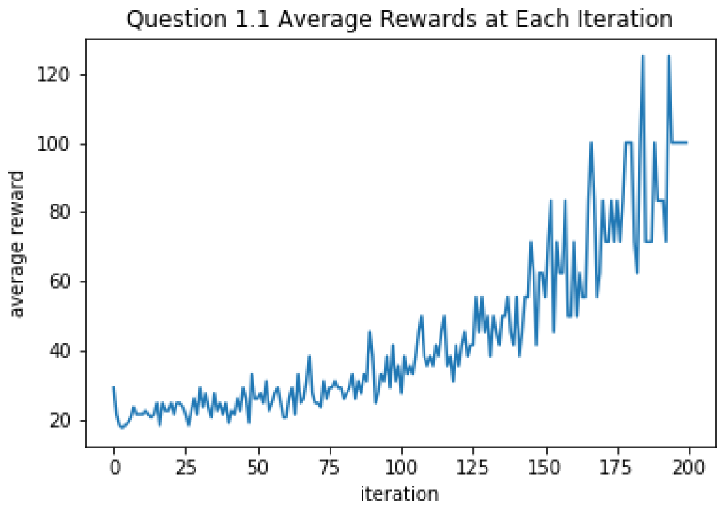

Figure 1: Question 1.1 Average Returns at Each Iteration

From the result we can see the average returns increase as iteration increases, which means our policy gradient algorithm converges.

1.2 Modified Reinforce Algorithm

Here we use a modified reinforce algorithm which updates gradient by:

$$\nabla J(\theta) \approx \frac{1}{n} \sum_{i=0}^{n} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta \left(a_t | s_t\right) \sum_{t'=t}^{T} \gamma^{t'-t} r\left(t'\right) \tag{2}$$

All the other settings remain the same. The result is shown in "Fig. 2": (Average returns in figure are not discounted)

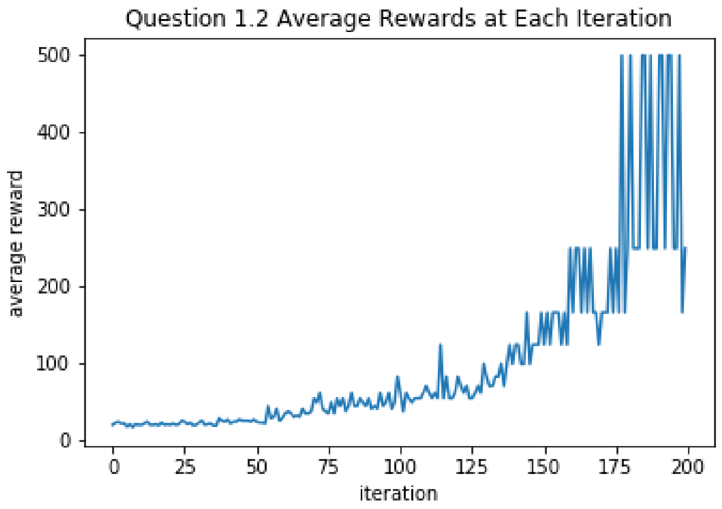

Figure 2: Question 1.2 Average Returns at Each Iteration

The result is much better than we got in question 1.1. It shows that the gradient is only dependent on the future rewards, hence the modified reinforce algorithm converges faster.

1.3 Baseline

Here we use reinforce algorithm with baseline:

$$\nabla J(\theta) \approx \frac{1}{n} \sum_{i=0}^{n} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta \left(a_t | s_t\right) \sum_{t'=t}^{T} \left(\gamma^{t'-t} r\left(t'\right) - b\right) \tag{3}$$

where b is the mean of batch returns, so after the subtraction the mean of modified returns remains 0. The baseline method is supposed to improve sample efficiency and improve numerical stability.

The result of reinforce with baseline is shown in "Fig. 3": (Average returns in figure are not discounted)

From the result we can see the result with baseline is better than our previous results, as it converges faster and gets higher rewards more consistently among the final batches. The result meets our expectation.

1.4 Different Batch Sizes

The results of reinforce with baseline with batch sizes $\{600, 800, 1000\}$ is shown in "Fig. 6": (Average returns are not discounted)

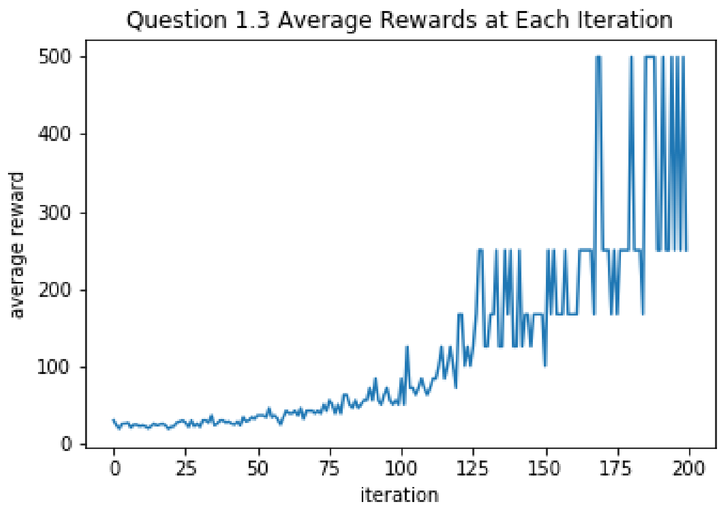

Figure 3: Question 1.3 Average Returns at Each Iteration

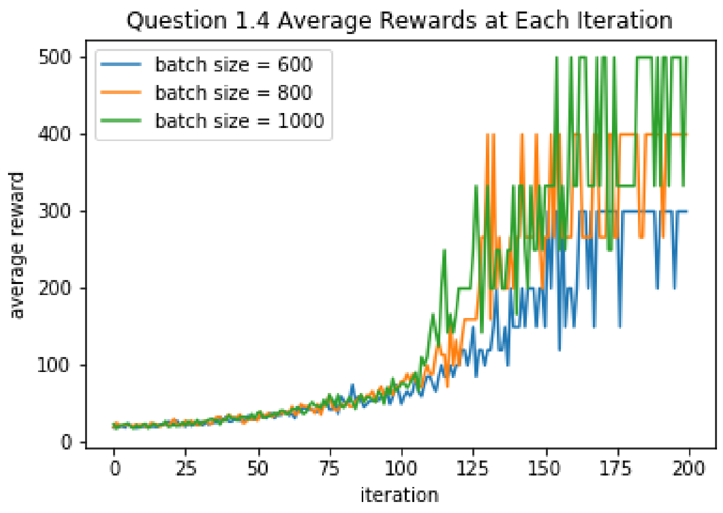

Figure 4: Question 1.4 Average Returns at Each Iteration

From the results we can observe that the result improves as batch size increases, and batch size 1000 gives the best result. Also, the algorithm tends to converge faster as batch size increases.

We know that reinforce algorithm needs a large amount of training data, here larger batch size gives us more training data, therefore we have better results.

2 2 Link Arm

In this problem, we use a 4-layer fully connected neural network to learn to policy. Two hidden layers have 128 and 64 nodes respectively, and activate function is tanh.

Covariance of multi-variable Gaussian is initialized as diag$[0.2, 0.2]$. Using learning rate $\alpha = 0.01$ and discount factor $\gamma = 0.9$, we are able to train the policy with batch size 1000 within 500 iterations. The policy is able to reach the goal in randomized environment consistently.

Below here shows the average rewards over iteration and average trajectory steps over iteration. At iteration 199 we terminated the training, since the policy was good enough.

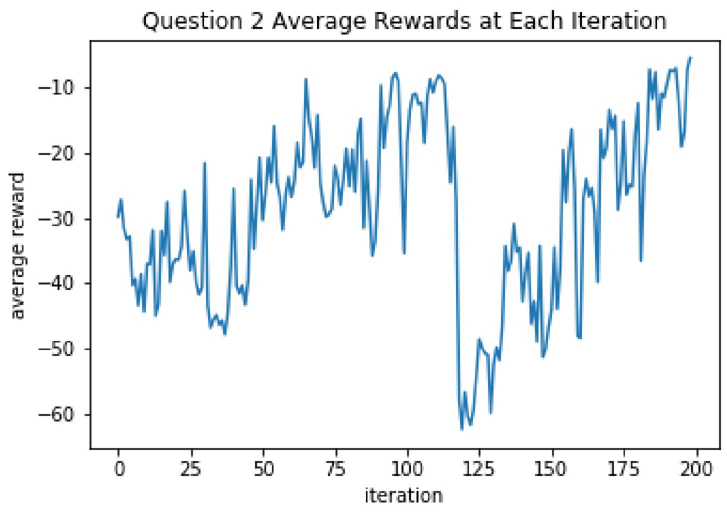

Figure 5: Question 2 Average Returns at Each Iteration

For the GIF please see attached file.

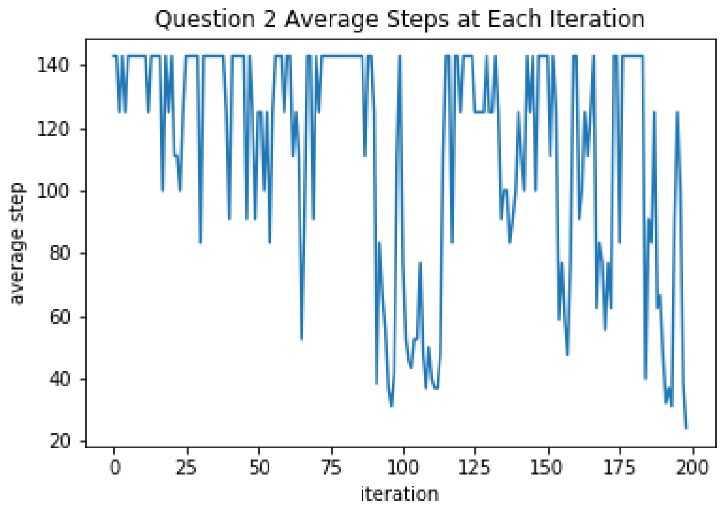

Figure 6: Question 2 Average Trajectory Steps at Each Iteration