ECE 276 Assignment 3 Report: Policy Gradients

Jingpei Lu
Jacob School of Engineering
University of California, San Diego

Date: November 7, 2019

1 CartPole-v1

1.1 Reinforce Algorithm with Variant Returns

For all the experiments in this section, the policy function $\pi_\theta(s, a)$ is modeled by a neural network with one hidden layer of 128 neurons. I use Adam as my optimizer and a learning rate of 0.002. The policy is trained by policy gradient method with a batch size of 500 steps, for 200 iterations. For training, the reward is discounted by $\gamma = 0.99$, and for evaluating, the average undiscounted return is calculated from the data collected for training.

The figures below show the training performance of the reinforce algorithm with some variations. Figure 1 shows the performance of the vanilla reinforce algorithm. Figure 2 shows the performance of the improved reinforce algorithm, where the returns only depend on the future rewards. Figure 3 shows the performance of the further improved reinforce algorithm, where the returns for each iteration are subtract by an baseline (subtract the mean and divided by the standard deviation). Figure 4 compares the performance of these three reinforce algorithms.

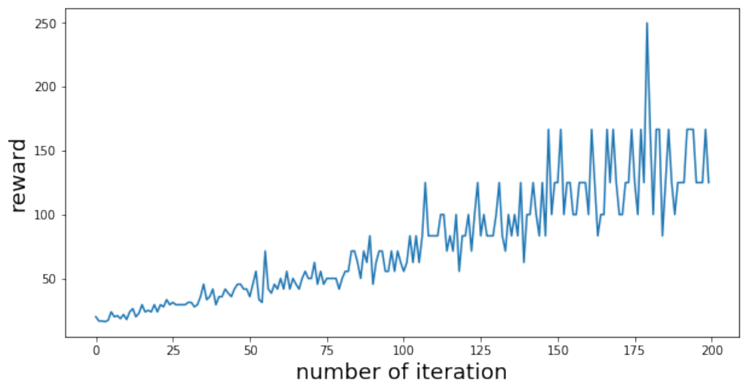

Figure 1: Average returns of the learned policy at each iteration for vanilla reinforce algorithm.

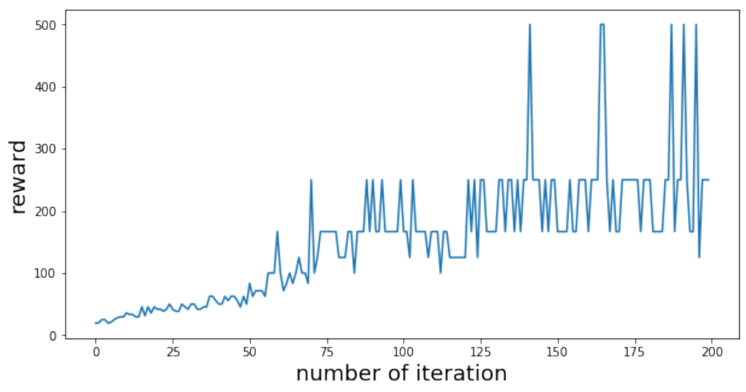

Figure 2: Average returns of the learned policy at each iteration for reinforce algorithm with only future rewards.

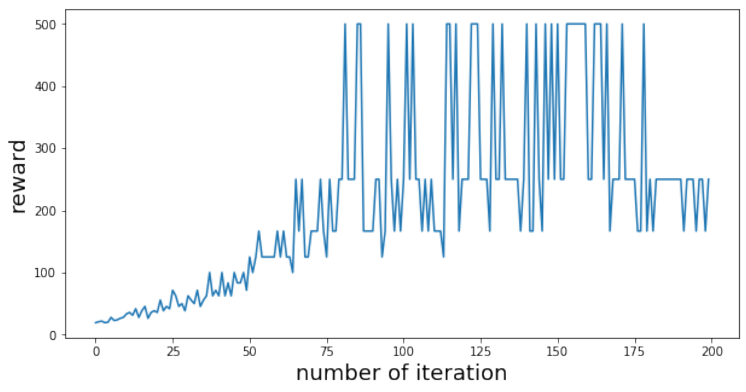

Figure 3: Average returns of the learned policy at each iteration for reinforce algorithm with baseline.

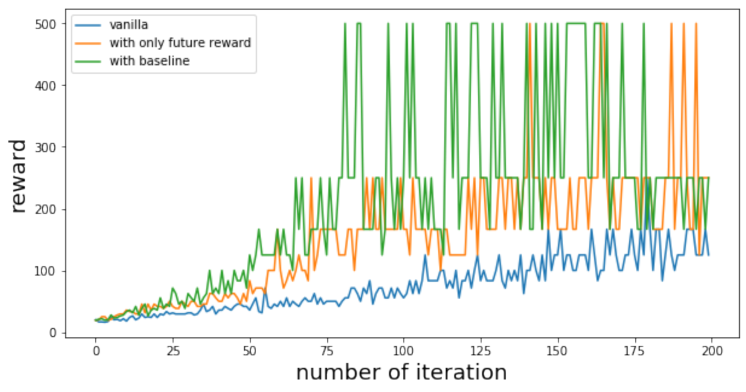

Figure 4: Comparison between the three reinforce algorithm.

From the above figures, we can see that two improved reinforce algorithms converge faster than the vanilla one, where the reinforce with baseline performs the best as it reduces the variance of the estimated returns.

1.2 Reinforce Algorithm with Different Batch Sizes

For all the experiments in this section, the reinforce algorithm with baseline is implemented.

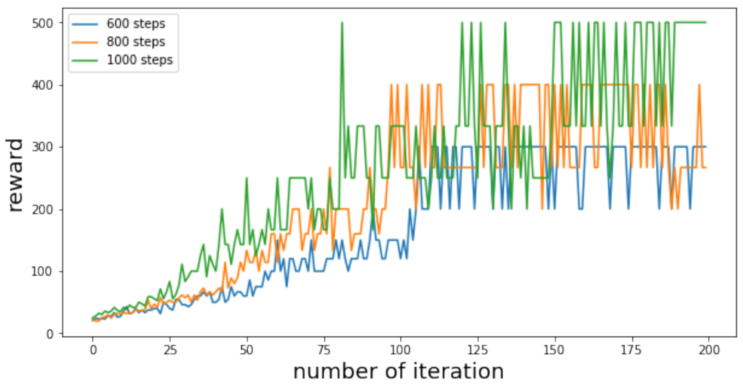

Figure 5: Comparison between the training performance of the reinforce algorithm with different batch sizes. The numbers on y axis indicates the undiscounted average returns, and the x axis indicates the number of iteration. The performance of taking 600, 800, 1000 steps for each iteration are indicated by blue, yellow, green curve correspondingly.

From the above figure, we can see that increasing the batch size dose improve the training. As we collect more samples from a policy, we have better estimation of the policy function.

2 2 Link Arm

For the experiments in this section, the policy function $\pi_\theta(s, a)$ is also modeled by a neural network with one hidden layer of 128 neurons. The Adam optimizer is used and the learning rate is 0.0002. The policy is trained by policy gradient method with a batch size of 2000 steps, for 500 iterations. For training, the reward is discounted by $\gamma = 0.9$. The actions are sampled from a multivariate normal distribution $N(f(s), \Sigma)$, where $f(s)$ is the neural network and covariance is initialized as $\Sigma = I(2) * 0.01$. The average return for each training iteration is plotted below.

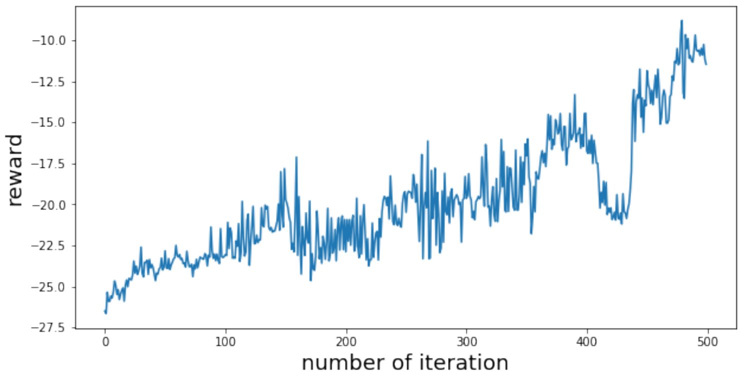

Figure 6: Average returns of the learned policy at each iteration.