CSE 410 Project 3:

Policy Gradient and Actor Critic

Peter M. VanNostrand
Department of Computer Science
University at Buffalo
Buffalo, NY 14261
pmvannos@buffalo.edu

Abstract

In this project we implemented a Policy Learning (PL) agent and an Advantage Actor Critic (A2C) agent. We then applied these learning agents to the OpenAI Gym CartPole-v1 environment which tasks an agent with balancing an inverted pendulum, the titular pole, but moving its base along a track. We evaluated the performance of both algorithms on the environment and have included those results below.

1 Environment

1.1 Description

The environment chosen for this exercise is the OpenAI Gym CartPole-v1 environment. This environment was chosen as it is one of OpenAI's simpler tasks, and as such requires significantly less computational training time than an environment like OpenAI Atari, and therefore can be easily trained on a home computer without need of a compute server.

CartPole-v1 the learning agent is tasked with balancing a vertical pole which is free to rotate in one dimension, by moving its base along a one-dimensional track aligned with its axis of rotation. The pole acts as an inverted pendulum and as such requires continual movement of the base to stay balanced vertically. The base is moved via a cart which the agent controls. A depiction of this environment is shown below in Figure 1.

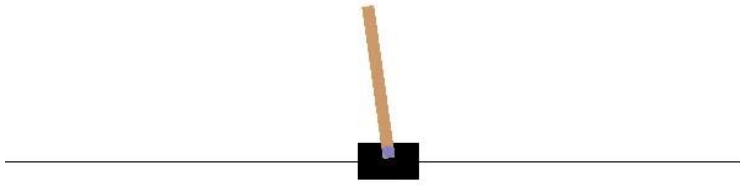

Figure 1: OpenAI CartPole Environment

1.2 Observation Space

The observation space of the CartPole environment is continuous, with physics simulations being performed on both the cart and pole to progress the environment. This continuous space is represented to the agent by four characteristic values. These are the cart's position, the cart's velocity, the pole's angle, and the speed of the pole measured at its tip. The minimum and maximum possible values for these characteristics are listed in Table 1.

Table 1: OpenAI CartPole Observations

Num	Observation	Min	Max
0	Cart Position	-2.4	2.4
1	Cart Velocity	-Inf	Inf
2	Pole Angle	~ -41.8°	~ 41.8°
3	Pole Velocity At Tip	-Inf	Inf

1.3 Action Space

To balance the pole the agent can take one of two actions at any given time step, namely pushing the car to the left or pushing the cart to the right. As the agent is not given an option to leave the cart in its current position, the agent must alternate pushing the cart left and right to maintain a relatively fixed position. Table 2 summarizes these actions.

Table 2: OpenAI CartPole Actions

Num	Action
0	Push cart to the left
1	Push cart to the right

1.4 Reward Function and Termination Conditions

To incentivize the agent to keep the pole standing for as long as possible, the environment gives the agent a reward of +1 at every timestep. This continues until the episode terminates. We report the total cumulative reward at termination as the score. Episodes can end in one of three ways: firstly, if the poll angle exceeds 12° in either direction, secondly if the cart position exceeds the minimum or maximum position, or if the timesteps exceeds 500. For this environment its considered that the agent has solved this task if the average reward is greater than 475 for 100 consecutive episodes.

2 Learning Algorithms

2.1 REINFORCE

The first learning agent implanted uses the REINFORCE algorithm. REINFORCE is a policy gradient reinforcement learning approach. In policy gradient approaches the agent attempts to maximize the total cumulative discounted reward it receives by learning an optimal policy π^{star}. Different to other reinforcement learning techniques we have studied, REINFORCE attempts to do this by representing its policy as π_θ where θ is a set of weights. The agent then performs stochastic gradient ascent on π_θ to learn the optimal value of $\pi^\star$. Note that in this approach we are attempting to directly approximate the optimal policy, rather than approximating the state-value or state-action-value function and choosing a policy based on this value function. REINFORCE is also a stochastic policy system, with the learned policy π_θ used to determine the likelihood of choosing an action such that $\pi_\theta(a, s) = P[a|s; \theta]$. What differentiates REINFORCE from other stochastic policy gradient methods, is its objective function. During stochastic gradient ascent REINFORCE uses G_t, the discounted cumulative future reward from Monte Carlo, as an unbiased sample of $Q^{\pi_\theta}(s_t.a_t)$ for scaling the gradient during its update step.

2.2 Advantage Actor Critic (A2C)

The second learning algorithm implemented is Advantage Actor Critic (A2C). Like REINFORCE A2C attempts to learn an optimal policy $\pi^\star$ via stochastic gradient ascent of π_θ. However, A2C does this in a different way as it is an actor critic method. A2C, rather than using G_t as an estimate of $Q^{\pi_\theta}(s_t.a_t)$, A2C introduces a second set of weights w which parametrize an approximation $V_w(s, a) \approx V(s)$. This allows the critic to evaluate an estimated

reward for a given state. This estimated state-value is used to calculate an advantage $A(s,a) = Q_w(s,a) - V(s)$, that represents how much better a given policy performs compared to the average reward for that state. As $Q_W(s,a)$ is not known, A2C approximates that values as $Q_w(s,a) \approx r + \gamma V(s',w)$, where γ is the discount factor and $V(s',w)$ is the approximate reward for the next state. This gives an overall advantage of $A(s,a) \approx r + \gamma V(s') - V(s)$. The critic learns an optimal value of w to approximate $V(s,w) \approx V(s)$ and uses this value to compute the advantage. The actor then uses the advantage to help learn an optimal value of θ by stochastic gradient ascent.

3 Results

3.1 REINFORCE

Using my implementation of the REINFORCE algorithm described above I performed learning on the CartPole environment and recorded the final score at then end of each episode. The results are shown below

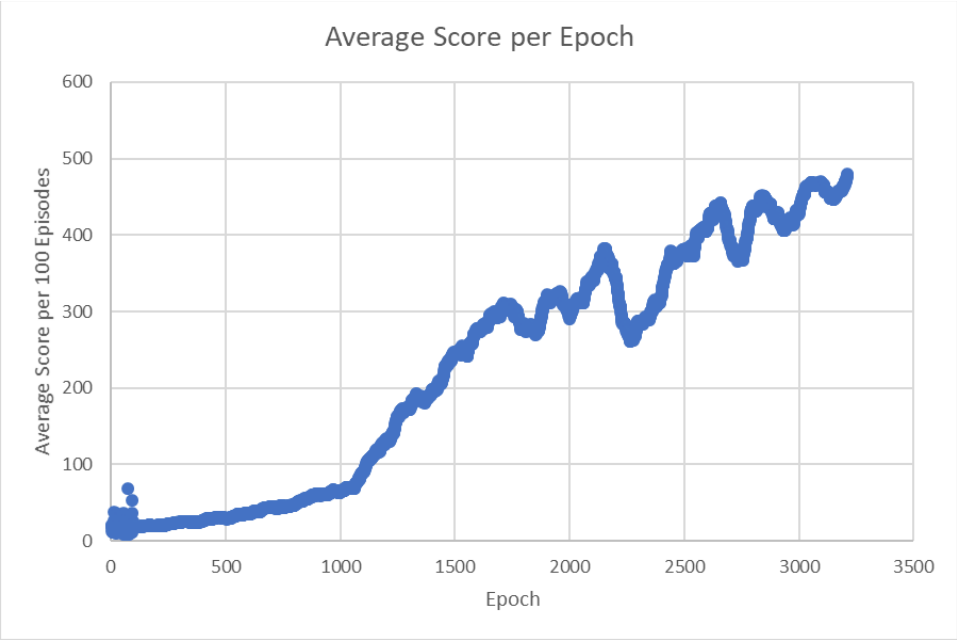

Figure 2: Average Score of REINFORCE Algorithm

Above is shown the score, that is the total cumulative reward, per training episode average over a 100-episode sliding window. As REINFORCE is a stochastic environment, averaging was used to make this trend easier to see among the stochastic noise. Without this the combined connecting lines between intermittent outliers consumed a surprising amount or real estate and quickly made the plot difficult to read. This window size was selected as the environment is considered solved after 100 consecutive episodes with an average above 475. We can see from this plot that the REINFORCE learning agent has solved the task at around 3250 episodes. Referencing the console the last recorded training epoch is 3,207 meaning that after this time training was terminated due to a solved environment. We can observe from this trend that a stead increase in performance was recorded with few prolonged periods of decrease.

3.2 AC2

Below are shown the results of the implemented AC2 learning agent

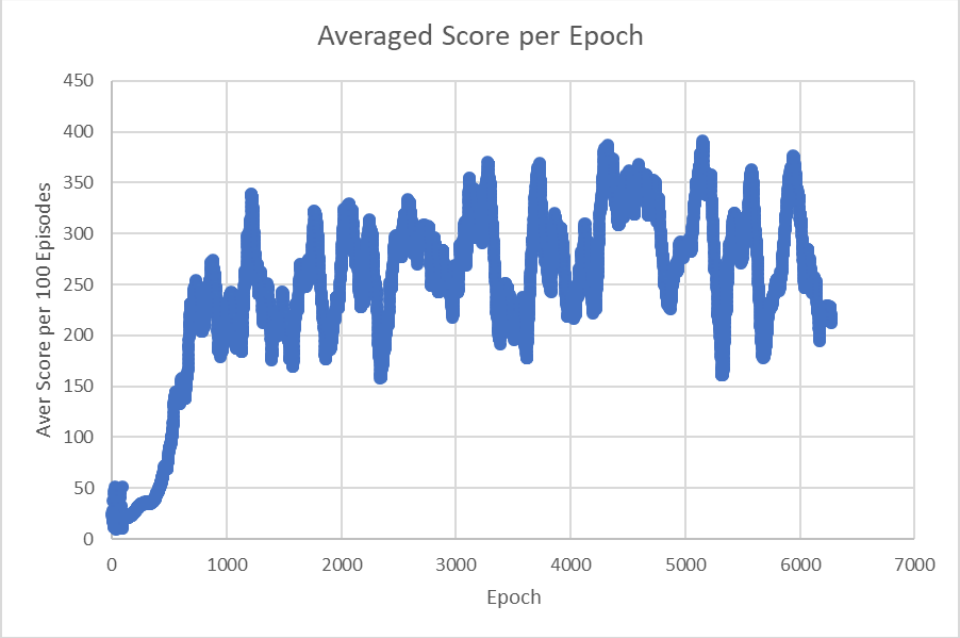

Figure 3: Average Score of AC2 Algorithm

Here the same averaging window was applied for clarity and comparability. From the plot we can observe that the AC2 algorithm experienced a much faster initial learning curve, but ultimately achieved a lower overall score, and was not able to solve the environment within ~6250 epochs. This is likely a result of less optimally selected learning rates. For simplicity of development the AC2 algorithm was developed using keras with a tensorflow backend to perform learning and gradient operations, this requires significant computation during training and led to long training times, therefore I was able to run fewer experiments to select an ideal value for the learning rates. As the REINFORCE algorithm performed manual backpropagation with a manual single softmax layer, the training time was much quicker, and therefore those results are better tuned. Additionally, the presence of two separate learning rates in the AC2 algorithm, as well as a greater number of overall weights likely contributed to this problem. I feel that based on the early results of both algorithms it is likely that AC2 would outperform REINFORCE given appropriate training and calibration. Nevertheless AC2 performed well reaching an average score of 250-300 after approximately 1250 epochs.