IEMS469-HW1

Deniz Sahin

October,15 2025

In this assignment, we implemented the policy gradient algorithm to train a neural network agent for two distinct OpenAI Gym environments: CartPole-v1 and Pong-v5. The objective for the agent in each game was to learn a policy that maximizes the total score. This report covers the implementation of the basic policy gradient algorithm, optimized with Adam and also addresses the issue of high variance by introducing a baseline to the algorithm, aiming for more stable and efficient training. Implementation details, training results, and analysis for both approaches are reported.

1 Cart Pole

1.1 Cart Pole without Baseline

This section details the implementation of the basic REINFORCE algorithm. The agent plays through an entire episode, and at the end, the policy network is updated based on the outcomes. For each action taken, we calculate the discounted cumulative reward from that point until the end of the episode. Actions that led to higher future rewards are encouraged, while actions that led to lower rewards are discouraged.

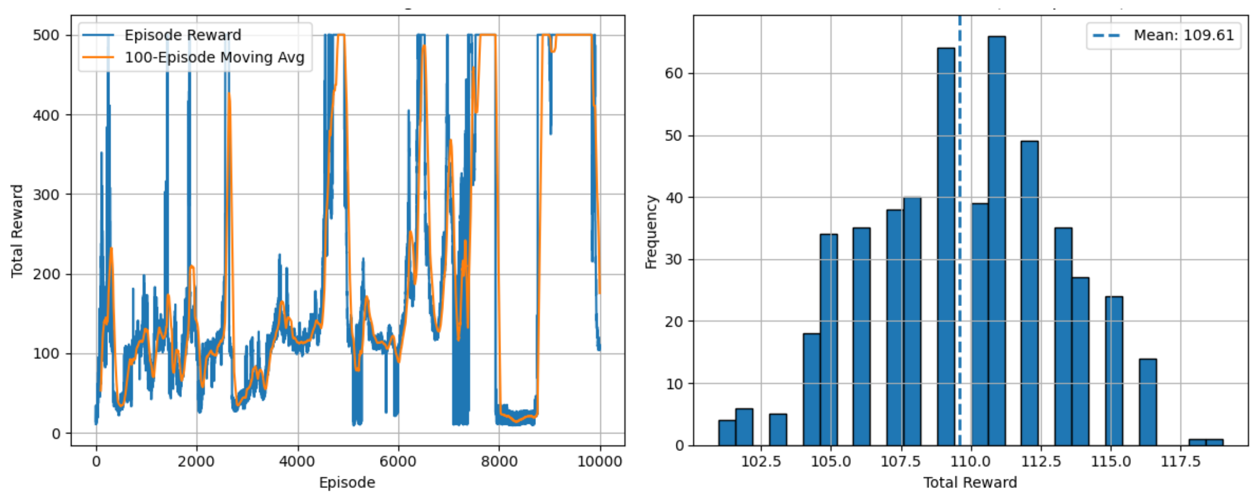

Figure 1: Results for Cart Pole without Baseline

The model was trained for 10,000 episodes using a learning rate of 0.01. Upon evaluation over 500 episodes, the agent achieved a mean reward of 109.61 with a standard deviation of 3.39.

1.2 Cart Pole with a Changing Baseline

To reduce the high variance associated with the basic policy gradient method, we implemented an Actor-Critic model. This model uses a learned, non-constant baseline. The network has two outputs: the Actor, which determines the policy, and the Critic, which estimates the value of the current state. The Critic's estimate serves as a baseline. The Actor is then updated based on the advantage, the difference between the actual return and the Critic's estimate, leading to more stable training.

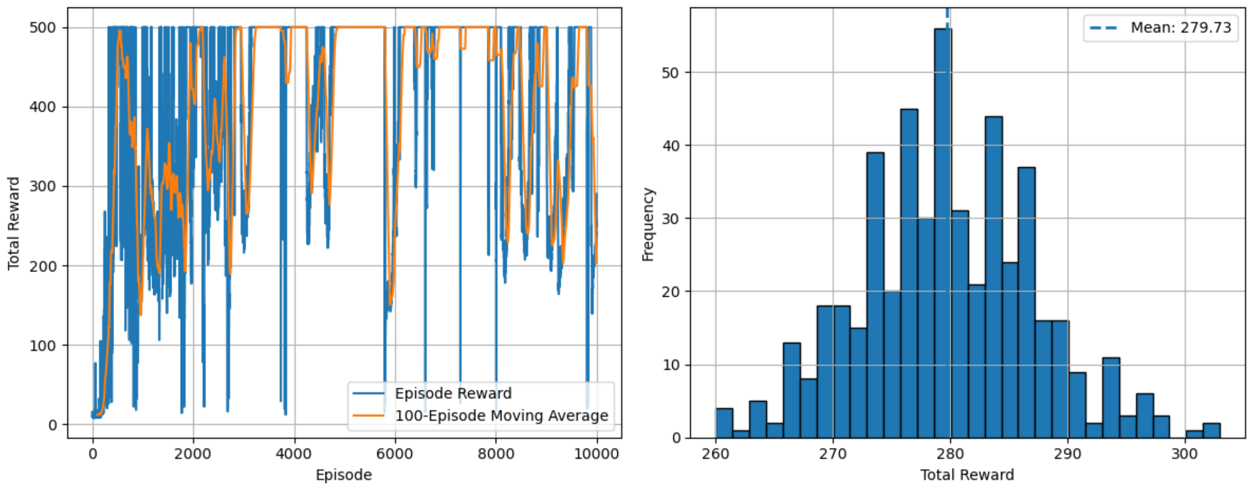

Figure 2: Results for Cart Pole with a Changing Baseline

The model was trained for 10,000 episodes with a learning rate of 0.01. The subsequent evaluation over 500 episodes yielded a mean reward of 279.73 and a standard deviation of 7.47. This result demonstrates a significant performance improvement over the model trained without a baseline. The extended training period was sufficient for the variance-reducing benefits of the baseline to manifest, an outcome that was not observed in the Pong environment, as will be discussed in the next section.

1.3 Cart Pole with a Constant Baseline

In this approach, we use a simpler baseline to reduce variance. After completing an episode, we calculate the discounted returns for every step. The baseline is then computed as the mean of all these returns. By subtracting this average from each return, we normalize the update signal. This means the policy is only strengthened for actions that performed better than the episode's average, which helps to stabilize the learning process.

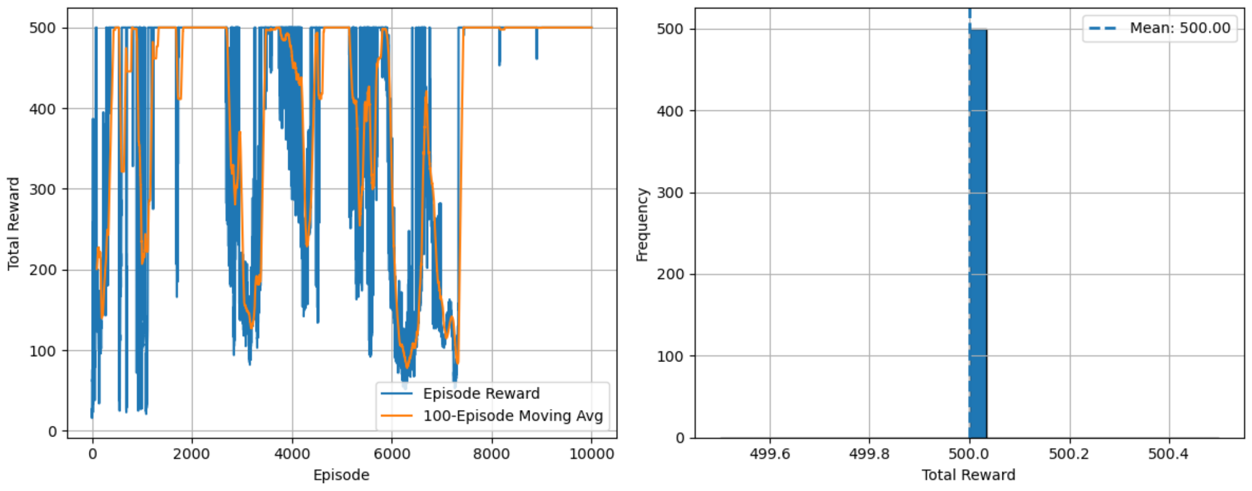

Figure 3: Results for Cart Pole with a Constant Baseline

The model was trained for 10,000 episodes with a learning rate of 0.01. This approach proved highly effective, as the agent achieved the maximum possible score in all 500 evaluation episodes. While this perfect performance is acknowledged to be sample-dependent, it nonetheless demonstrates that the training duration was sufficient for the constant baseline method to master the environment.

2 Pong

2.1 Pong without Baseline

For the Pong environment, the state is represented by a pre-processed image of the game screen. Specifically, we use the difference between consecutive frames to capture motion. The basic REINFORCE algorithm is applied here, where the policy network is a Convolutional Neural Network (CNN) that processes these images. The network is updated after a batch of episodes based on the discounted future rewards.

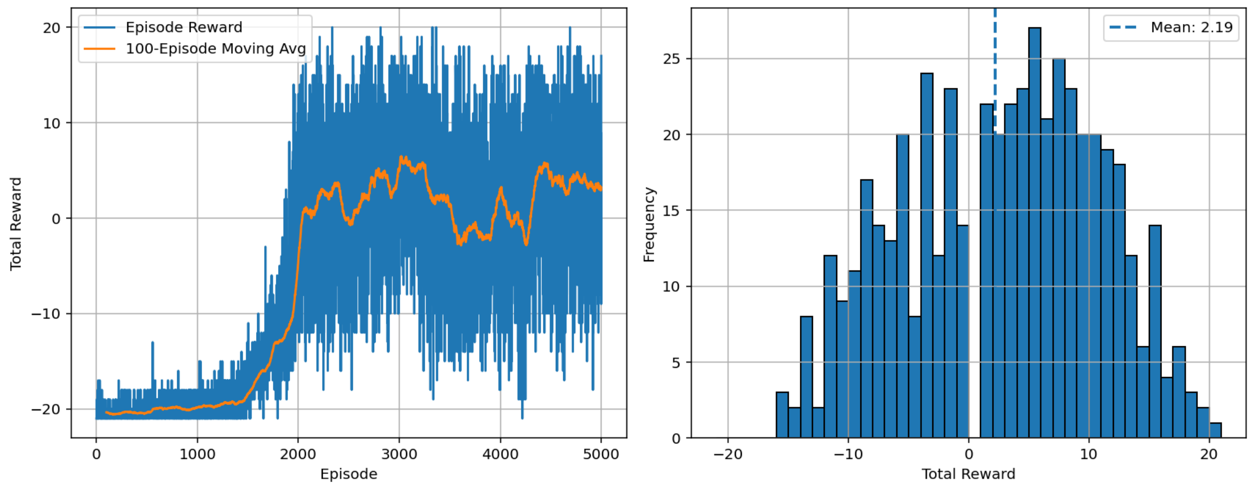

Figure 4: Results for Pong without Baseline

Due to computational constraints (servers), the model was trained for 5,000 episodes. A learning rate of 0.001 was selected to ensure training stability and prevent the divergence. To further stabilize the process, the policy was updated in batches, with gradients accumulated over 5 episodes before each optimization step. Given the high complexity of the Pong environment, this approach resulted in moderate performance. The learning process was characterized by a slow initial phase which gradually improved as training progressed. The final evaluation over 500 episodes yielded a mean reward of 2.19 with a standard deviation of 8.20.

2.2 Pong with a Changing Baseline

Similar to the CartPole experiment, we implement an Actor-Critic model for Pong. A shared CNN processes the game frames, which then feeds into two separate heads: an Actor head for the policy and a Critic head for the state-value estimate. This learned state-value acts as a dynamic baseline, helping to stabilize the gradient updates and improve the learning efficiency in the high-dimensional Pong environment.

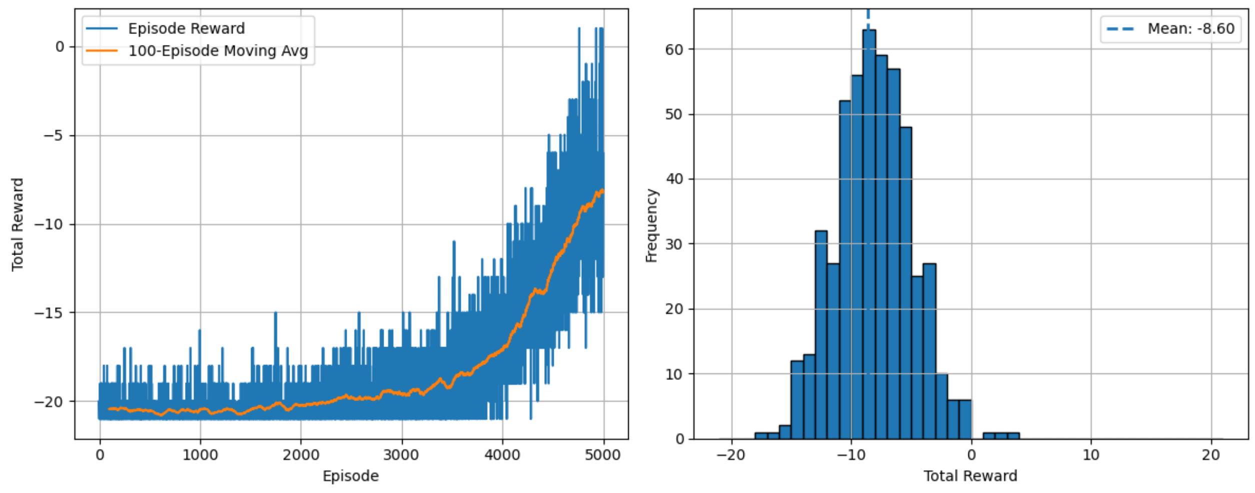

Figure 5: Results for Pong with a Changing Baseline (5000 Episodes for Training)

The model was again trained for 5,000 episodes with a learning rate of 0.001. As illustrated Figure 5, the introduction of a baseline was effective in reducing training variance. Nevertheless, this training duration proved insufficient for the agent to achieve a high level of performance. The subsequent evaluation over 500 episodes yielded a mean reward of -8.60 and a standard deviation of 3.21.

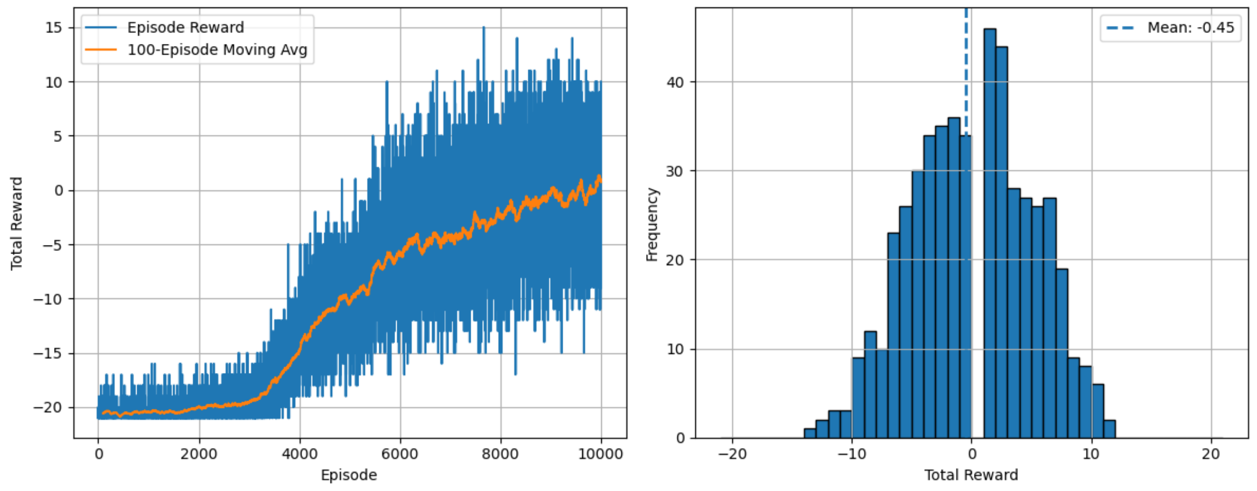

Figure 6: Results for Pong with a Changing Baseline (10000 Episodes for Training)

Figure 6 highlights the critical role of training duration. By extending the training from 5,000 to 10,000 episodes, the agent's performance improved substantially. The evaluation over 500 episodes yielded a mean reward of -0.45 with a standard deviation of approximately 5.0, demonstrating that sufficient training allows the baseline method to effectively learn a competent policy.

2.3 Notes

Figure 7 illustrates training divergence or catastrophic forgetting. This phenomenon was observed during an initial training run where the learning rate was set to 0.001. A single large, unfavorable update to the policy network pushed the agent into a suboptimal state from which it could not recover, resulting in a sudden and permanent collapse in performance. The problem was resolved by reducing the learning rate to 0.0001, which stabilized the training (Figures 8 and 9). However, this adjustment came at the cost of a significantly slower learning process, highlighting the critical trade-off between training stability and convergence speed. Luckily, in another try with the learning rate 0.001, the same problem didn't happen again.

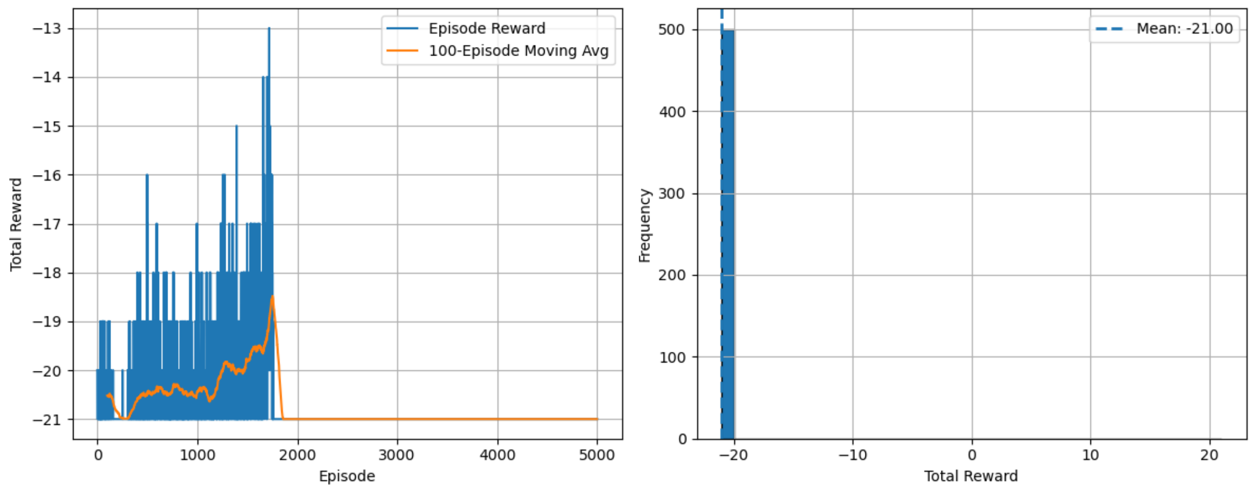

Figure 7: Training Divergence

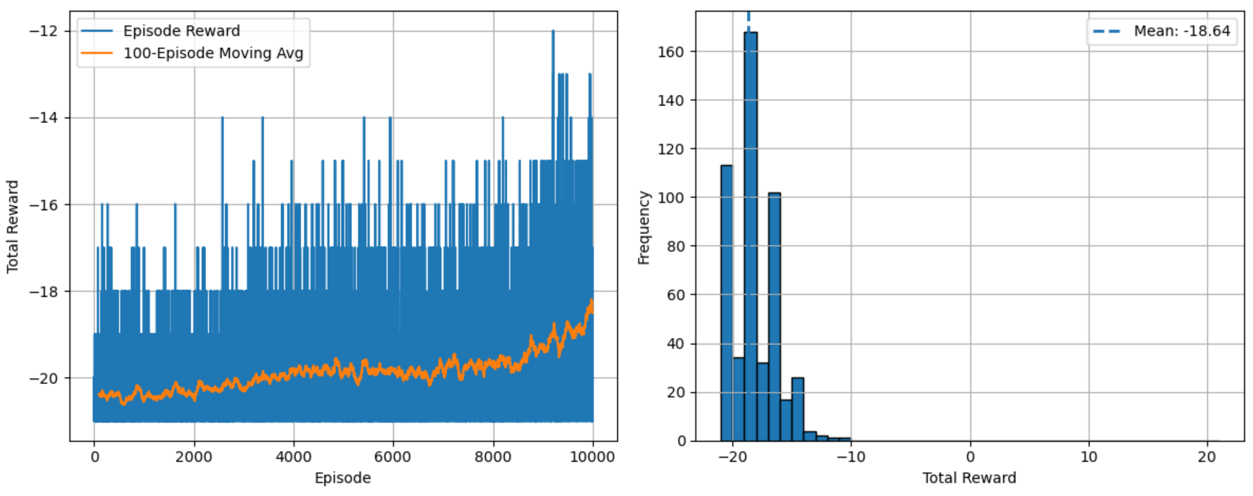

Figure 8: Results for Pong without Baseline (Learning Rate 0.0001)

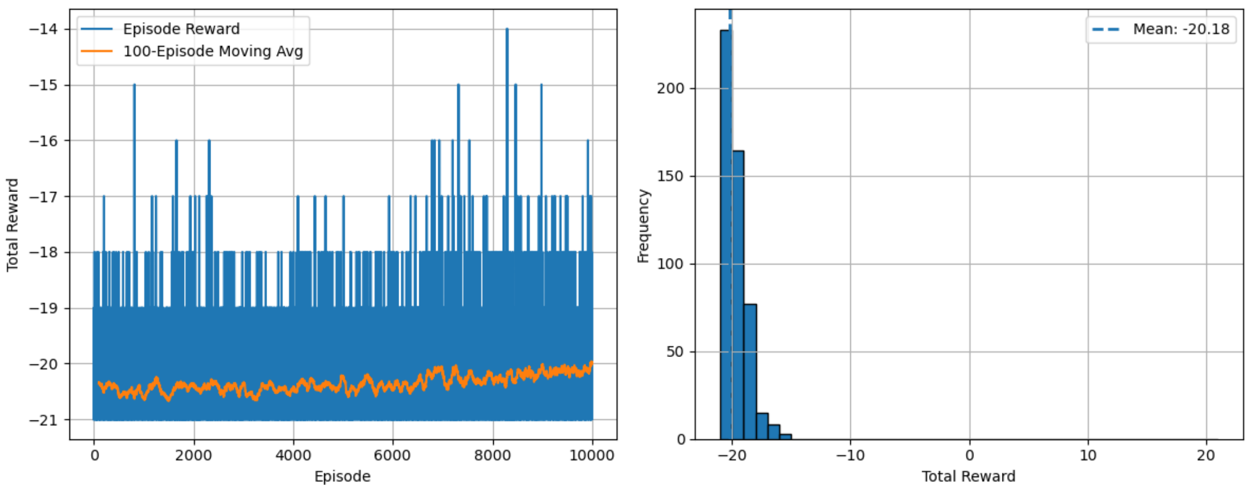

Figure 9: Results for Pong with a Changing Baseline (Learning Rate 0.0001)