Solving SpaceInvaders and Breakout environment with Proximal Policy Optimization

Gaurav Madhav Pai **Keshav Jethaliya**
Department of Computer Science
State University of New York at Buffalo
Buffalo, NY 14260
gmadhavp@buffalo.edu
keshavje@buffalo.edu

Abstract

Here in this report I present 2 different environments, *SpaceInvaders* and *Breakout* by Atari, taken from openAI gym. PPO algorithm is applied to both the environments, in which the agent is trained to play the game and learn to win. The agent trains and improves itself by playing a couple of times.

1 Introduction

Reinforcement Learning

Reinforcement learning (RL) is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward. Reinforcement learning is one of three basic machine learning paradigms, alongside supervised learning and unsupervised learning.

It differs from supervised learning in that labelled input/output pairs need not be presented, and sub-optimal actions need not be explicitly corrected. Instead the focus is finding a balance between exploration (of uncharted territory) and exploitation (of current knowledge).

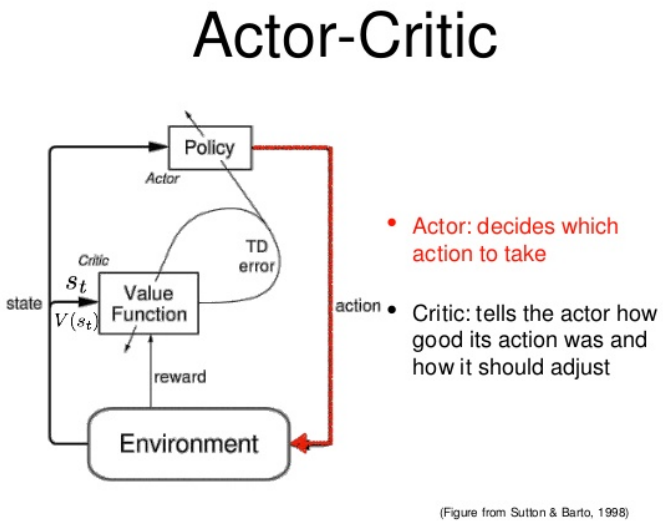
Deep Learning

Deep learning is a class of machine learning algorithms that uses multiple layers to progressively extract higher level features from the raw input. For example, in image processing, lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as digits or letters or faces.

Actor Critic

The Actor Critic model is a better score function. Instead of waiting until the end of the episode as we do in Monte Carlo, we make an update at each step (TD Learning).

1. The “Critic” estimates the value function. This could be the action-value (the *Q value*) or state-value (the *V value*).
2. The “Actor” updates the policy distribution in the direction suggested by the Critic (such as with policy gradients).



2 Architecture

The following components are used in the architecture of the model.

2.1 Advanced Policy Gradient

The **policy gradient** methods target at modeling and optimizing the policy directly. The policy is usually modeled with a parameterized function respect to θ, $\pi_\theta(a|s)$. The value of the reward (objective) function depends on this policy and then various algorithms can be applied to optimize θ for the best reward.

2.2 Proximal Policy Optimization (PPO)

One of the Policy Gradient methods is Proximal Policy Optimization. PPO strikes a balance between ease of implementation, sample complexity, and ease of tuning, trying to compute an update at each step that minimizes the cost function while ensuring the deviation from the previous policy is relatively small.

$$L^{CLIP}(\theta)=E\hat{}_t[min(r_t(\theta)A\hat{}_t,clip(r_t(\theta),1-\varepsilon,1+\varepsilon)A\hat{}_t)\ ]$$

- the policy parameter
- E^t$E\hat{}t$ denotes the empirical expectation over timesteps
- rtrt is the ratio of the probability under the new and old policies, respectively
- A^t$A\hat{}t$is the estimated advantage at time tt
- $\varepsilon\varepsilon$ is a hyperparameter, usually 0.1 or 0.2

```
#surrogate losses (for actor)
surrogate_loss_1 = log_ratio * advantage
surrogate_loss_2 = torch.clamp(log_ratio, 1.0 - clip_param, 1.0 + clip_param) * advantage

actor_loss  = - torch.min(surrogate_loss_1, surrogate_loss_2).mean()
critic_mse_loss = (return_ - state_value).pow(2).mean()

#Combined loss function to update both actor and critic network
combined_loss = 0.5 * critic_mse_loss + actor_loss - 0.001 * entropy
```

We use the combined_loss to train both actor network and critic network together.

In PPO, if a policy gives us better reward than expected, then we increase the probabilities of the actions associated with it, and decrease the probabilities if it gives us worse reward than expected.

2.3 Generalized Advantage Estimator (GAE)

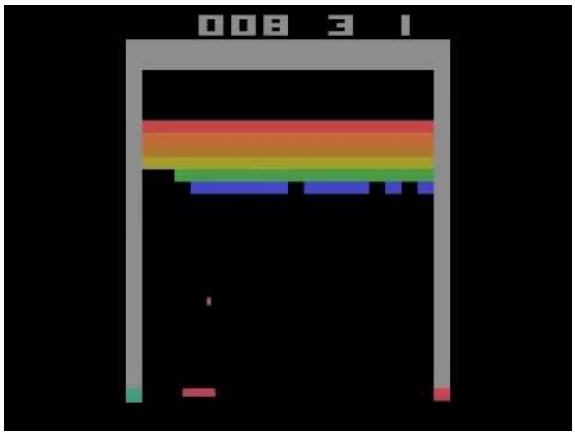
A family of policy gradient estimators is proposed that significantly reduce variance while maintaining a tolerable level of bias. We call this estimation scheme, parameterized by $\gamma \in [0, 1]$ and $\lambda \in [0, 1]$, the generalized advantage estimator (GAE).

2.3 SpaceInvaders-v0

In this environment, the observation is an RGB image of the screen, which is an array of shape (210, 160, 3) Each action is repeatedly performed for a duration of k frames, where k is uniformly sampled from {2,3,4}{2,3,4}. The goal is to shoot all the space invaders to maximize the score, while saving the agent.

2.4 Breakout-v0

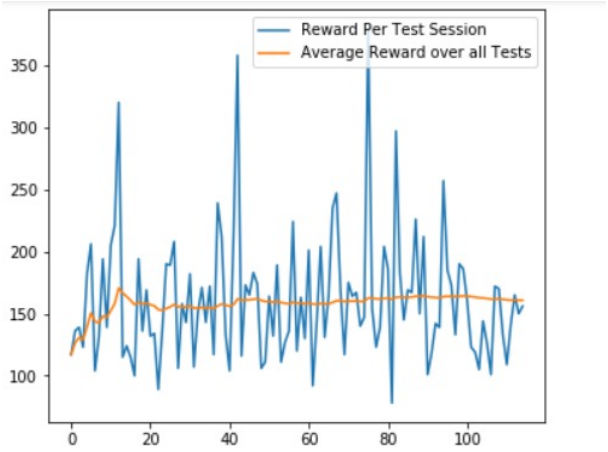
In this environment, the observation is an RGB image of the screen, which is an array of shape (210, 160, 3) Each action is repeatedly performed for a duration of k frames, where k is uniformly sampled from {2,3,4}{2,3,4}. The goal of the agent is to clear all the bricks by hitting then with the ball, while saving the ball from falling.



3 Results

SpaceInvaders:

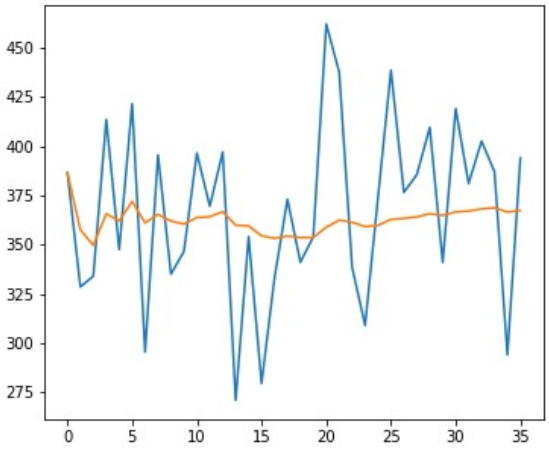
Following are the rewards, given by the PPO model to the agent after each episode, taking some action:



Testing the policy in the environment for every 125 of 15000 steps of training with learning rate for actor critic network of 1e-4

Breakout:

Following are the rewards, given by the PPO model to the agent after each episode, taking some action:



Testing the policy in the environment for every 400 of 14000 steps of training with learning rate for actor critic network of 1e-4

4 Conclusion

In conclusion, seeing the above graphs, we were at a disadvantage when we used artificial neural networks for actor and critic. Due to limited computational power, we could not use convolutional neural networks which would have given better results on both Atari SpaceInvaders and Breakout. Further, we used a single learning rate for both actor and critic networks. Using a higher learning rate for critic to help it learn faster than actor would also have led to better conversions.

References

[1] https://towardsdatascience.com/introduction-to-various-reinforcement-learning-algorithms-part-ii-trpo-ppo-87f2c5919bb9

[2] https://www.youtube.com/watch?v=WxQfQW48A4A&list=WL&index=2&t=7s

[3] Lecture Slides by Prof. Alina Vereshchaka

[4] http://wikipedia.org

[5] http://rail.eecs.berkeley.edu/deeprlcourse/static/slides/lec-9.pdf