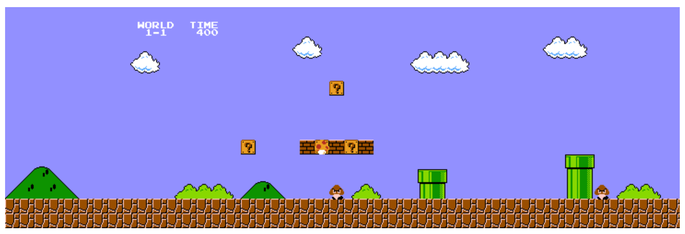
Optimizing Learning Rates in Proximal Policy Optimization: An Exploration Within Super Mario Bros

Moe Khalil
Tom Adamo
Walker Stewart

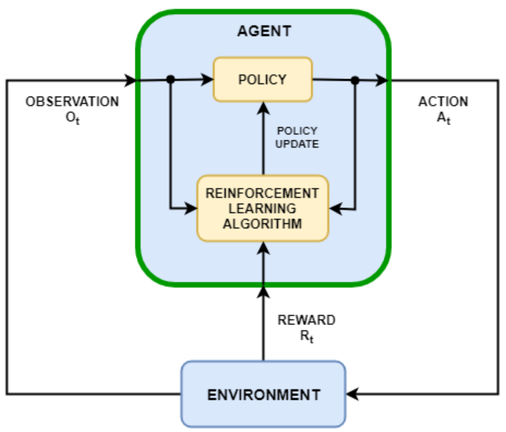
Abstract

This paper explores the impact of different learning rates on the PPO algorithm's performance in Super Mario Bros, finding that a learning rate of 3e-4 yields the most consistent results.



Introduction

In this study, we sought to refine the application of a popular reinforcement learning algorithm, Proximal Policy Optimization (PPO), which often requires meticulous tuning of hyperparameters for different problem domains. Among these, **the learning rate** plays a crucial role in algorithm performance, influencing the speed and stability of learning. This refining was done in the context of a Super Mario Bros game-playing agent.

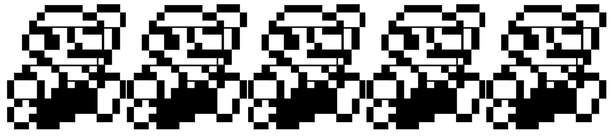


Related Work

Previous projects have employed a variety of approaches to tackle the challenge of applying reinforcement learning to Super Mario Bros. For example, Liao, Yi, and Yang's 2012 project and Klein's 2016 project both used traditional Q-Learning techniques.

Method - Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO) is a reinforcement learning algorithm that enhances stability and efficiency by maintaining modest updates to the policy, preventing large deviations from the previous policy and thus improving the sample efficiency. It does so by utilizing a penalty term in its objective function to discourage large deviations.



Method - Value Function

Velocity (change in x-value over time)

Death penalty

$$r = v + c + d$$

Clock difference (time spent)

Method - Pre-processing

GrayScaleObservation

DummyVecEnv

VecFrameStack

Method - Environment

We leveraged the gym_super_mario_bros library to craft our training environment. Through callbacks and monitoring tools, we effectively tracked and enhanced our RL agent's learning journey.

Experiments - Metrics

We closely monitor our model's

Reward Function Over Time

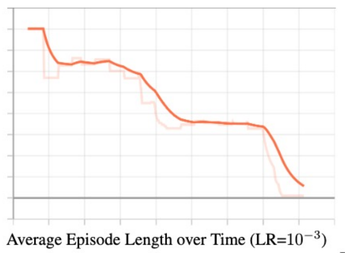
and

Average Episode Length.

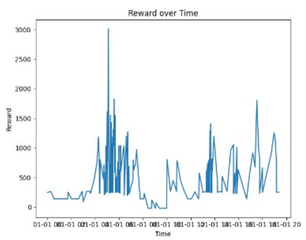
These metrics serve as a mirror, reflecting our agent's learning progress and its increasing proficiency in navigating the game environment.

Experiments - Learning Rate

Four different learning rates were tested to understand their impact on the performance of the Proximal Policy Optimization algorithm in the Super Mario Bros environment.
10e-3: Agent demonstrated inconsistency in policy improvement, as indicated by the oscillation of the value function output.
10e-4/3*10e-4: Oscillations in value function were observed but with an overall uptrend, suggesting a more consistent learning process.
10e-5: Agent displayed notable reward function oscillations and opted to optimize reward through fewer steps, indicating limited progression.



Average Episode Length over Time (LR=10^{-3})



Reward Function over Time (LR=10^{-3})

Experiments - Conclusion

Learning rate was found to be a crucial factor influencing the performance of the PPO algorithm, with the rate of 3*10e-4 delivering the most consistent results in the Super Mario Bros gaming environment.