IMPALA Implementation for CartPole-v1

Yechan Woo

December 5, 2024

1 Introduction

IMPALA (Importance Weighted Actor-Learner Architecture) is a distributed reinforcement learning framework that efficiently separates acting and learning processes. This implementation applies IMPALA to solve the CartPole-v1 environment, demonstrating both learning efficiency and computational performance. The environment requires an agent to balance a pole on a moving cart, providing a classic test case for reinforcement learning algorithms.

2 Implementation Architecture

The system consists of the following key components:

- **Actors (4)**: Independent processes that interact with CartPole-v1 environments
- **Learner (1)**: Central process that updates the neural network using collected experiences
- **Queue Manager**: Coordinates data flow between actors and learner
- **Neural Network**: Shared architecture producing both policy and value estimates

3 System Configuration and Optimization

3.1 Hyperparameters

After conducting numerous experiments with various configurations, the following set of constants was identified as consistently producing reliable and well-performing results across training:

Category	Parameter	Value
Training	Learning Rate	0.001
	Discount Factor (γ)	0.99
	Gradient Clip	0.7
	Baseline Loss Weight	0.4
	Entropy Cost	0.008
V-trace	$\bar{c}$	0.95
	$\bar{\rho}$	0.95
Architecture	Hidden Size	64
	Queue Size	100
	Batch Size	6
	Unroll Length	4

Table 1: System hyperparameters

3.2 Implementation Optimizations

The implementation employs n-step bootstrapping with an unroll length of 4, allowing the system to effectively propagate value estimates across longer time horizons. This bootstrapping mechanism helps reduce variance in value estimation while maintaining a balance between bias and variance in the learning process. The implementation uses truncated importance sampling ratios ($\bar{\rho} = 0.95$) to ensure stable off-policy learning while preventing excessive variance from importance weights.

4 Evaluation

4.1 Overview

The IMPALA implementation demonstrated effective performance in solving the CartPole-v1 environment. The training process executed approximately 12,000 learner steps. The system began showing signs of policy stabilization around episode 800 (˜43 seconds of training time), with all actors consistently achieving the maximum score of 500 soon after. The final model exhibited stable performance with well-balanced policy between exploration and exploitation. Throughout training, the system maintained consistent processing times, with total computation averaging around 0.004 seconds per step.

4.2 Learning Progression

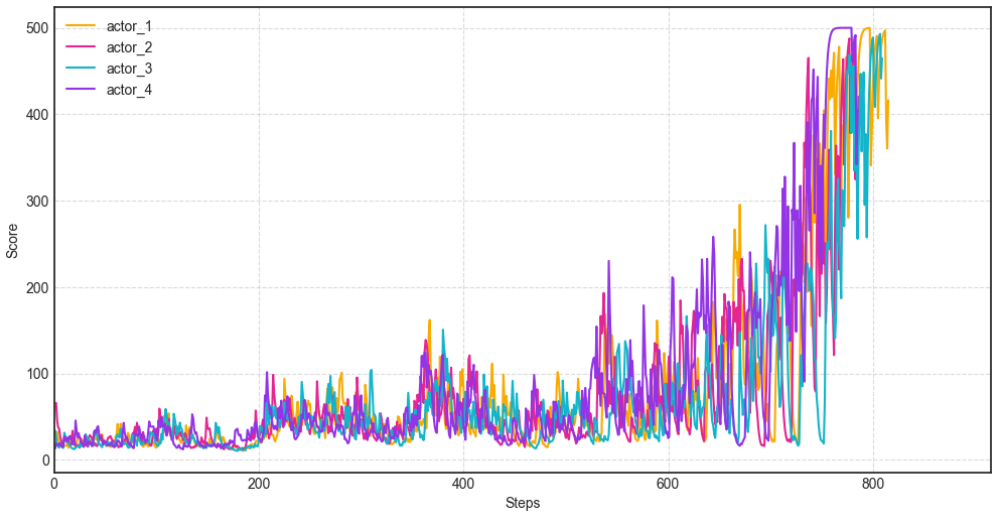
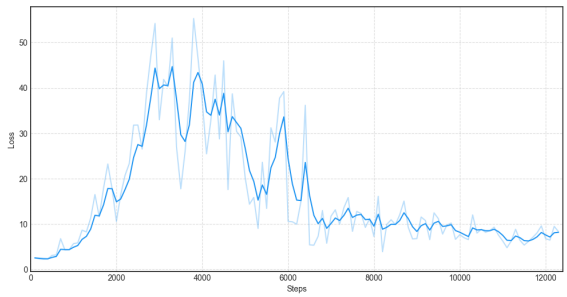


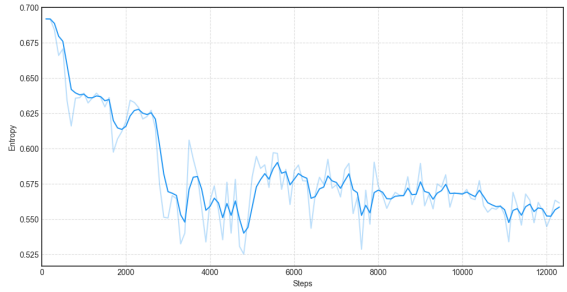
Figure 1: Episode scores showing learning progression across all actors

The training process showed three distinct phases. Initially (Episodes 0-400), the agent focused on exploration, leading to volatile scores below 100. This was followed by a rapid improvement phase (Episodes 400-800), where scores increased consistently, indicating significant policy refinement. Finally, during the convergence phase (Episodes 800+), all actors achieved the maximum score of 500, demonstrating stable and optimal performance.

4.3 Training Metrics



(a) Critic Loss



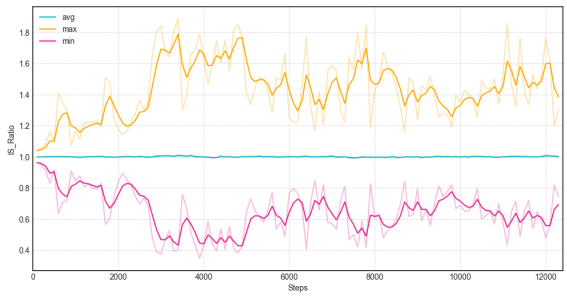
(b) Entropy

Figure 2: Core training metrics

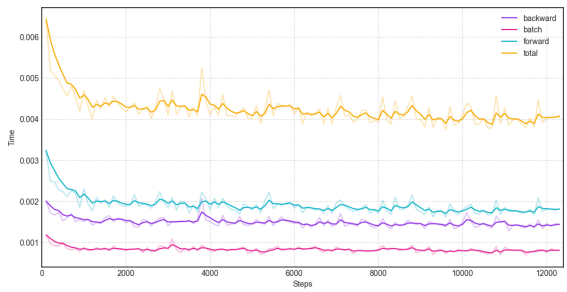
The critic loss progression revealed several key phases in the learning process. Initially, the loss experienced a significant spike, reaching approximately 50 around step 5000, indicating substantial policy updates during this period. This was followed by a decrease after step 6000, suggesting policy convergence as the agents learned effective strategies. Finally, the loss stabilized around 5-10, maintaining consistent performance throughout the remainder of training.

The entropy measurements provided insights into the exploration-exploitation balance. Starting at about 0.7, the entropy gradually decreased to 0.525, demonstrating a controlled reduction in policy uncertainty. This gradual decline indicates that the agents maintained sufficient exploration capability while progressively focusing on optimal actions, achieving an effective balance between exploration and exploitation.

4.4 System Performance



(a) Importance sampling ratios



(b) Component timing analysis

Figure 3: System performance metrics

The importance sampling ratios demonstrated robust off-policy learning characteristics throughout the training process. The average ratio maintained stability at 1.0, which is theoretically correct. Maximum ratios showed controlled variation between 1.25 and 1.8, but clipped by the hat for computational efficiency. The minimum ratios consistently remained around 0.5, ensuring that no experiences were excessively down-weighted during learning.

The computational performance analysis revealed stable processing times across all components. The forward pass consistently averaged 0.0018 seconds, demonstrating stable inference time across varying input states. The backward pass maintained efficient gradient computation at around 0.0015 seconds, while batch processing showed remarkable efficiency at 0.0008 seconds. The total processing time averaged approximately 0.004 seconds per step, indicating well-optimized computational resource utilization. The forward pass,

averaging 0.0018 seconds, emerged as the primary computational bottleneck in the system. Optimizing the model architecture or employing hardware acceleration (e.g., GPUs) could further reduce this bottleneck.