Enhancing Game Control Through Hybrid Reinforcement Learning

Danhua Yan

Department of Computer Science, Stanford University

Introduction

- **Motivation** Many environments have high-dimensional state spaces, sparse rewards, and complex dynamics, making pure exploration inefficient. Limited or costly exploration can prevent RL agents from learning usable policies.
- **Objective** This project explores hybrid RL (HRL) by combining offline human demonstrations with online agent explorations to enhance game control through guided exploration.

Background

- **Super Mario Bros.** The NES game Super Mario Bros has challenges including sparse rewards, precise timing for jumps, and strategy for finishing within time limits. The problem is non-Markovian, requiring temporal local structure.
- **Behavior Cloning (BC)** A supervised learning approach to learn a policy from human demonstrated (s, a) pairs.
- **PPO** Proximal Policy Optimization (PPO) is an RL policy gradient algorithm for training intelligent agents.
- **Bootstrapping PPO with Guidance** PPO starts with a pre-trained policy or is guided by demonstrations early in training.

Methods

Data and Environment

- **Human Demonstration** Script to record trajectories in the same format as the RL agent, capturing controller actions and game states.
- **Customized Environment** Actions: 3 movements. Termination: Single life and timeout terminations. Rewards: Combination of scores, time, milestones, movement to give dense rewards. Sampling: Sampled game states at 15fps to reduce trajectory length.

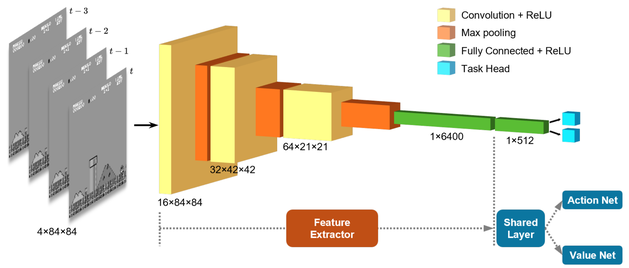


Figure 1. **PPO agent architecture for playing Super Mario Bros.** Four downsampled gameplay frames are stacked to represent the current state s_t, as input to the CNN actor-critic PPO networks.

Reinforcement Learning Models

- **State Representation** Grayscale and downsampled gameplay frames. State s_t at time t is represented by four stacked frames, $f_{t-3}, \cdots, f_t$, preserving local temporal dynamics (see Figure. 1).
- **Policy Network Architectures** A three-layer CNN produces a dense feature representation. The PPO actor-critic shares a hidden layer, followed by separate action and value heads. The BC network combines the feature extractor with the action head. (see Figure 1)
- **Baselines** Offline: BC policy on human data; Online: PPO and DQN with decaying ϵ-greedy.
- **Hybrid Reinforcement Learning (HRL)**
  - **PPO Weights Pre-Training** (`HRL1`) Pre-train the policy network via BC on human data, then use those weights for PPO. Evaluate different parts of the actor-critic policy network transferred from BC policy (`HRL1-MLP`: action net; `HRL1-FEAT`: feature extractor; `HRL1-ALL`: all weights except value head).
  - **Assisted Explorations** (`HRL2`) Use a single human play trajectory to reset PPO rollouts strategically from near the winning state back to the initial state [1]. Propose an exponential decay schedule where earlier states get more rollouts (`HRL2-ER`).

Experiments

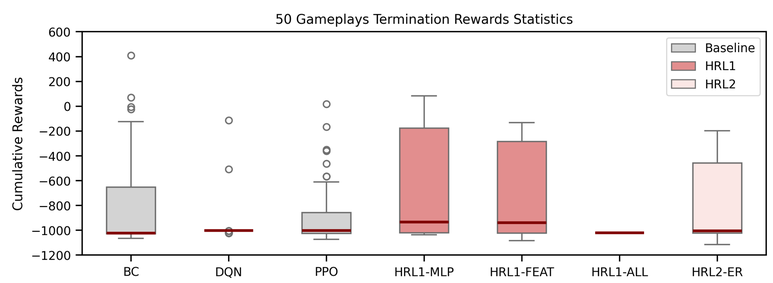


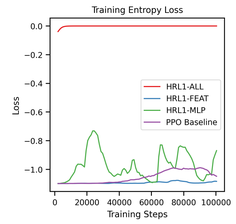
Figure 2. **Gameplay statistics of different policies.**

All online policies were trained for 250 rollouts to compare efficiency. After training, each policy was evaluated as a stochastic policy. We conducted 50 gameplays for each policy and compared cumulative rewards.

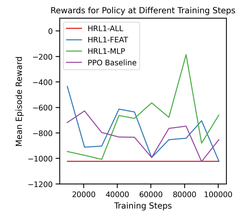
From Figure 2, we conclude: 1) Behavior cloning is a strong starting point, training faster; 2) Transfer learning (`HRL1`) is effective when transferring either features or actions, but not both; 3) Assisted explorations (`HRL2`) offer no significant improvement to the mean rewards over the PPO baseline, however higher reward distributions are observed.

Analysis

(a) Jump over a tall pipe is the most challenging state for the level



(b) `HRL_1` Policies Entropy Loss During Training



(c) `HRL_1` Policies Rewards During Training

Figure 3. Understanding the performance of the policies

All policies failed to complete the level, indicating challenges in the intrinsic game mechanics and HRL algorithms.

- **Game Challenge** Level 1-1 has a tall pipe early on that traps Mario. Continuous actions are required to jump over it, which is difficult for the agent to learn. Temporal structure frames help, but action history is also crucial. BC's randomized supervised learning discards such information (see Figure 3(a)).
- **PPO-Weights Pre-Training** Loading either the feature extractor (`HRL1-FEAT`) or action head (`HRL1-MLP`) provides a prior for state and action distributions, aiding exploration. However, loading both leads (`HRL1-ALL`) to low entropy, causing PPO to stop exploring. Issues with covariate shift and limited demo coverage arise, as only 5 human plays with "good" states are recorded, providing little information for bad states (see Figure 3(b)(c)).
- **Assisted Explorations** Figure 2 shows that assisted explorations do not improve mean rewards significantly. Mario often terminates at the tall pipe (Figure 3(a)). However, once Mario passes this point, higher reward distributions are observed, indicating the benefit of practicing later parts of the level without rediscovering them via random exploration.

Conclusion

Behavior cloning accelerates early training, and selective transfer of model components provides bootstrapping of PPO learning. Future work should enhance state representations to address game challenges, and better demo data and algorithms are needed to mitigate covariate shift and limited demo coverage.

Reference

[1] Tim Salimans and Richard Chen.
Learning montezuma's revenge from a single demonstration, 2018.

dhyan@stanford.edu