DS251: Project Report

Project Title : MARL Penalty Shot Challenge
Group No.: 8
Team Members:
1. Abhishek Kumar
2. Arnav Gautam
3. Dhruv Gupta
4. Mitul Vardhan

1. Introduction

This project is based on Reinforcement Learning. We have built a multi-agent Penalty Shootout Game using reinforcement learning. This game creates a competitive environment involving multiple agents with intricate interactions and continuous state and action spaces. This setup is designed to emulate real-world social behaviors in a dynamic manner.

In the game, there will be two agents - a puck and a bar. Puck's objective is to reach the finish line without making contact with the bar, while the bar's goal is to capture the puck before it successfully crosses the finish line. Various cutting-edge deep reinforcement learning techniques, including TD3, DDPG, and PPO, for the puck and bar agents have been used, engaging them in competitive training scenarios. Through iterative experimentation, we explore diverse reward functions and fine-tune hyperparameters to determine the most effective solution for both agents.

2. Problem Statement

The challenge involves two entities: the puck and the bar. The objective for the puck is to traverse the finish line without coming into contact with the bar, which, in turn, aims to capture the puck before it reaches the finish line. All positional coordinates are normalized within the range [-1,1]. The initial positions of the puck and the bar are (-0.75,0) and (0.75,0), respectively. The finish line is situated at x = 0.77. Both agents are governed by joysticks providing input but within the range [-1,1].

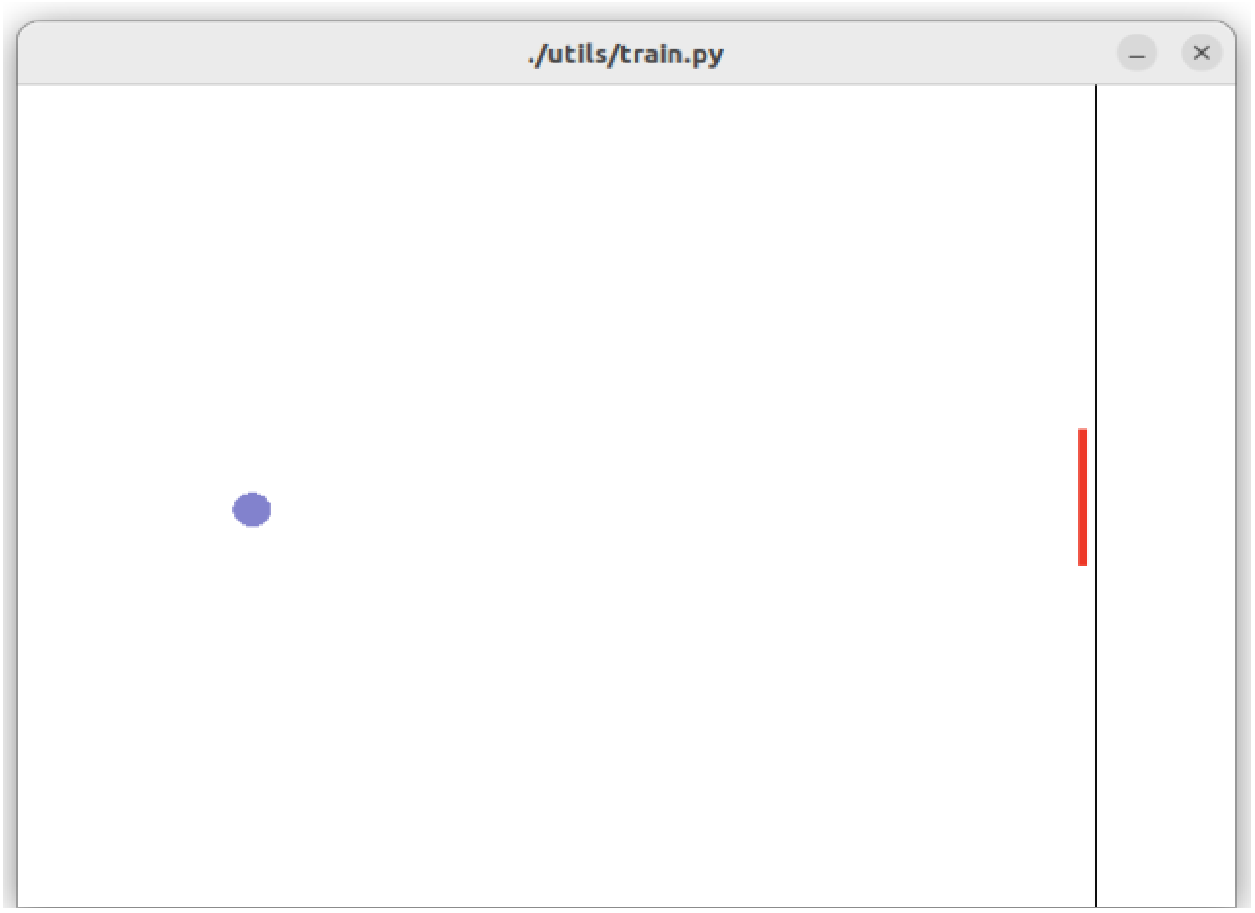

Figure 1: Rendered Environment : From Local Machine

Both agents are constrained to stay within the screen boundaries. If the bar successfully catches the puck, it receives a reward of +1, while the puck receives a reward of -1. Conversely, if the puck manages to elude the bar and reach the finish line, it earns a reward of +1, and the bar incurs a penalty of -1. Above image shows the interface of the game.

3. Environment Details and Implementation

3.1 Environment Details

The problem is presented as an RL challenge with two competing agents. Due to potential policy changes in the opponent's strategy during training, non-stationarity is introduced for both agents. Rewards are assigned at the end of each episode, with +1 granted to the victorious player and -1 to the defeated one.

For the bar agent, the state space includes the bar's position, the puck's angular position, θ, and an indicator variable v_ind. v_ind ranges from -3 to 3 (only integer values), where +1, +2, +3 signify near-maximal positive actions taken by the agent in the last 1, 2, 3 time steps, respectively. The same interpretation applies to negative v_ind values.

We have incorporated v_ind in the state space because capturing the count of recent time steps where the agent executed actions close to the maximum, can influence its future decisions.

3.2 Implementation

The codebase is structured into three primary packages:

3.2.1 Environment

This component manages the normalized coordinates of the puck and the bar, including parameters such as θ and the acceleration indicator variable, v_ind. Each time step in the gym environment package for this multi-agent problem involves receiving action pairs from the puck and bar agents. Subsequently, it updates state variables, providing the new state and a rewards tuple for each agent based on transition mechanics and termination criteria.

Additionally, the environment is visually rendered at each step using an internal rendering library for the puck, bar, and goal line.

3.2.2 Library

Tools for model training and testing

We utilize the Tianshou library for constructing custom networks and executing model training and testing. To optimize computational speed and efficiency, we have seamlessly integrated it with our gym environment using wrappers. Both the environment and agents come with configuration files for specifying parameters. A universal script is employed to parse arguments from the shell and pass them onto the script during both training and testing.

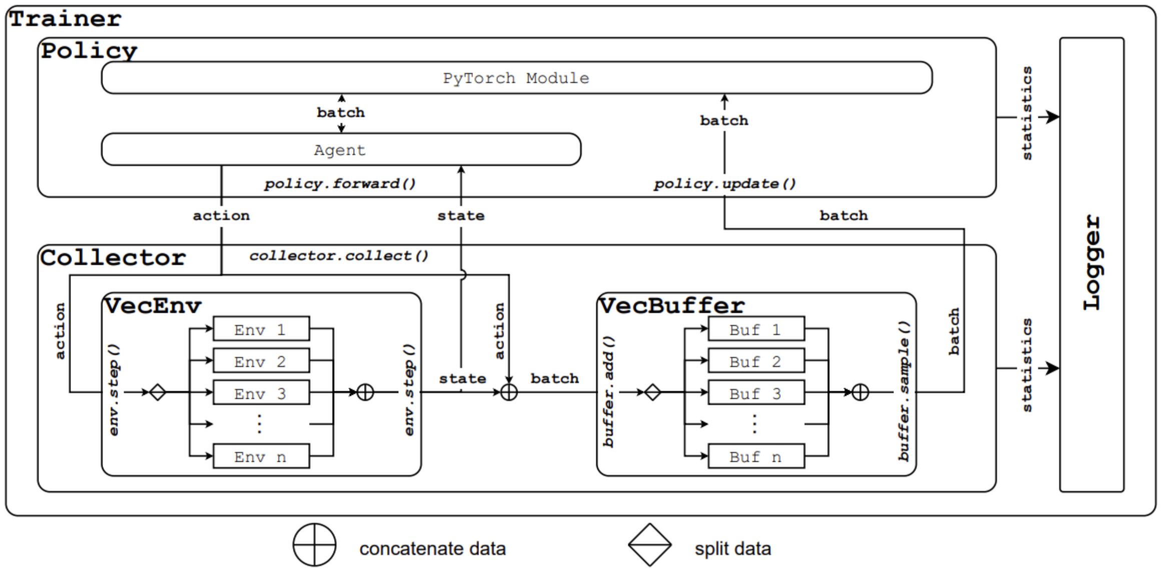

Figure 2: A high-level depiction of Tianshou's training process

3.2.3 Two-Agent Policy Wrapper

To support training within a two-agent environment, we devised a Two-Agent Policy Wrapper. This wrapper takes distinct policies for the puck and bar agents as input, functioning with the environment as a consolidated policy. It transmits the experience tuple from the environment to the agents and then processes actions from both agents, combining them before forwarding the input to the environment.

3.2.4 Environment Wrapper

The original environment wasn't conducive to advanced algorithmic training due to its complex observations and straightforward rewards. Thus, we created a wrapper atop the original environment, incorporating the following features:

1. Customized reward function for improved training.
2. Flattened observation with one-hot encoding for the discrete environment, providing a simplified observation structure better suited as input for neural networks.
3. Enabling discretization of action space with a parameter describing the size of new action space.

4. Solution Approach

We compare DRL algorithms, including TD3, DDPG, and PPO, by employing them as either the puck or the bar, aiming to optimize the most effective algorithm for each role. The original environment provides a reward function characterized by a rectangular pulse, sharply decreasing when the puck misses the bar at the edge. To expedite and smooth the learning experience, we modify the reward function explicitly to convey the learning objective. However, during the evaluation phase, we revert to using the actual reward function to assess the agent's performance in the authentic environment.

5. Contributors

Name	**Roll Number**
Abhishek Kumar	12140040
Dhruv Gupta	12140580
Arnav Gautam	12140280
Mitul Vardhan	12141070

References

- He He, Jordan L. Boyd-Graber, Kevin Kwok, and Hal Daum´e III. Opponent modeling in deep reinforcement learning. CoRR, abs/1609.05559, 2016. URL http://arxiv.org/abs/1609.05559.
- Kelsey R McDonald, John M Pearson, and Scott A Huettel. Dorsolateral and dorsomedial prefrontal cortex track distinct properties of dynamic social behavior. Social Cognitive and Affective Neuroscience, 15(4):383–393, 05 2020. ISSN 1749-5016. doi: 10.1093/scan/nsaa053. URL https://doi.org/10.1093/scan/nsaa053.
- Yuan Pu, Shaochen Wang, Xin Yao, and Bin Li. Context-based soft actor critic for environments with non-stationary dynamics. CoRR, abs/2105.03310, 2021. URL https://arxiv.org/abs/2105.03310.
- Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. Nature, 518(7540):529–533, Feb 2015. ISSN 1476-4687. doi: 10. 1038/nature14236. URL https://doi.org/10.1038/nature14236.Ben Moews, Rafael S de Souza, Emille EO Ishida, Alex I Malz, Caroline Heneka, Ricardo Vilalta, Joe Zuntz, COIN Collaboration, et al. Stress testing the dark energy equation of state imprint on supernova data. Physical Review D, 99(12):123529, 2019.
- Ben Moews, Rafael S de Souza, Emille EO Ishida, Alex I Malz, Caroline Heneka, Ricardo Vilalta, Joe Zuntz, COIN Collaboration, et al. Stress testing the dark energy equation of state imprint on supernova data. Physical Review D, 99(12):123529, 2019.
- Jiayi Weng, Huayu Chen, Dong Yan, Kaichao You, Alexis Duburcq, Minghao Zhang, Hang Su, and Jun Zhu. Tianshou: A highly modularized deep reinforcement learning library. arXiv preprint arXiv:2107.14171, 2021.