Reinforcement Learning: Football with Manchester

Florina Asani, Pascal Bakker, Amisha Jindal, Oscar Garcia Fernandez

Goal: Create an A.I. that consistently wins a majority of football games and compare the ability of four different RL Models to perform in such a complex setting. Using the Google environment, the agent will control one player at a time, and switch between football players to successfully defend and score points.

Introduction

The objective of RL is to prepare brilliant agents that can cooperate with their current circumstance and unravel complex errands, with true applications towards advanced mechanics.

Football requires the team to simultaneously strategize how to score goals and how to defend from the opponent. A single agent must solve multiple situations and leverage teamwork to win.

Figure 1. Football Engine

Google Research Football Environment

A. Google Research Football Environment

A material science based 3D football reproduction where agents control possibly one or all football players in their group, figure out how to pass among them, and work out how to defeat their rival's guard to score objectives.

B. Football Engine

An imitation of a football match including objectives, fouls, corner and extra shots, and offsides. Written in profoundly advanced C++ code, it can run on off-the-rack machines, both with GPU and without GPU-based delivering empowered.

C. Football Benchmarks

A bunch of benchmark issues are proposed for RL research dependent on the Football Engine. Three versions are given: *the Football Easy Benchmark,* the Football Medium Benchmark, and the Football Hard Benchmark, which just vary in the strength of the adversary.

Environment Specifications

A. State and Observation Space

The raw observation gives us the state definitions which are as follows: *ball position and possession, player positions, the current scoreline, the game state (yellow cards, red cards, score etc.), current active player.*

B. Action Space

Top	Bottom	Left	Right
Top-Left	Top-Right	Bottom-Left	Bottom-Right
Short Pass	High Pass	Long Pass	Shot
Do Nothing	Sliding	Dribble	Stop-Dribble
Spring	Stop Moving	Stop Spring	

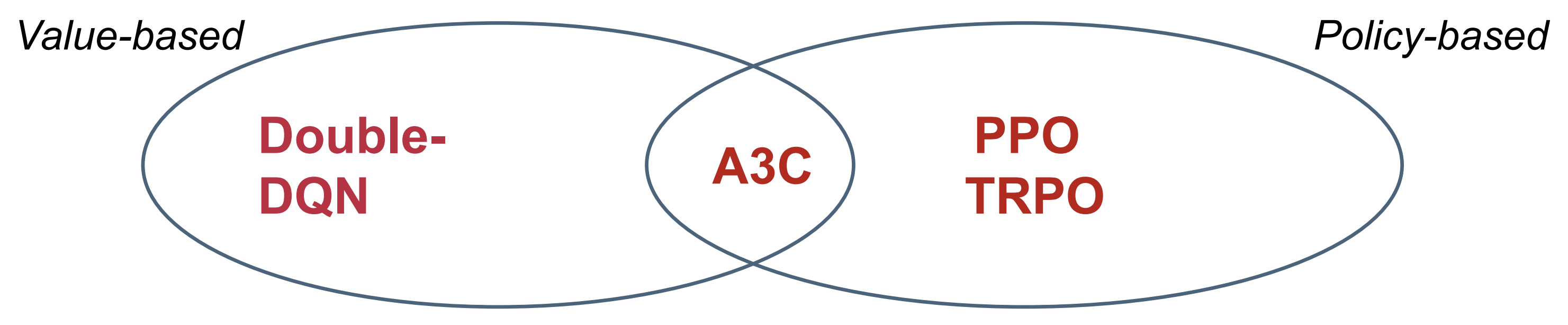
Table 1. Available action set in the Football Environment

C. Rewards

The engine includes two reward functions:

- *Scoring*: +1 for scoring a goal and 01 for conceding a goal.
- *Checkpoint:* is a reward by encoding the domain knowledge that scoring is aided by advancing across the pitch.

Background



Model-Free control

A. TRPO

A scalable algorithm that optimizes policies by gradient descent. It updates policies using the largest step possible while considering a specific constraint that discusses how close the new and old policies can be.

B. A3C

A deviation of Q-Learning, utilizes a policy gradient with a functional approximating in the form of a neural network. This network uses an actor-critic model, with the benefit of asynchronous.

C. Double DQN

A model free, off policy, algorithm that utilises Double Q-learning to reduce overestimation by decomposing the max operation in the target into action selection and action evaluation.

D. PPO

A type of policy gradient training that alternates between sampling data through environmental interaction and optimizing a clipped surrogate objective function using stochastic gradient descent.

Experiments and Results

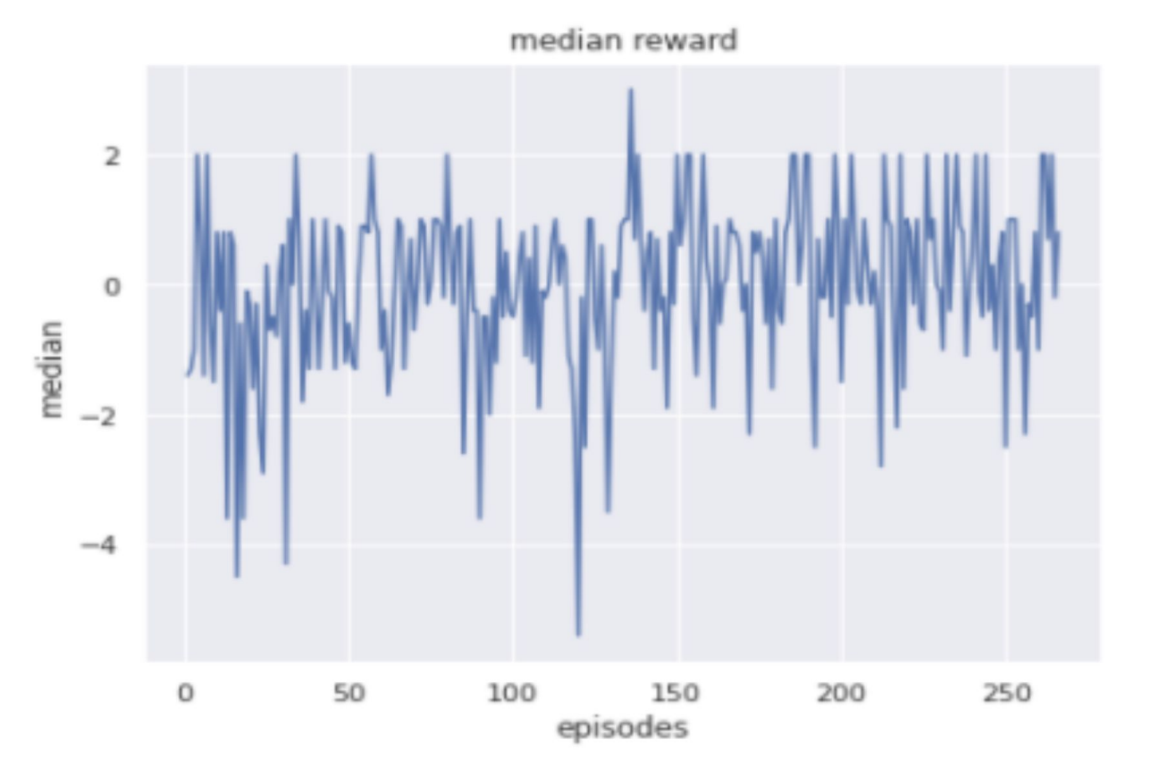


Figure 2. Median rewards for the Double DQN trained model

The DQN model:

trained with an epsilon greedy strategy for 800000 steps

running an evaluation game every 3000 training steps

The PPO model:

trained for 450 episodes, 1350000 steps

running validation game every 10 episodes.

Performs better than DQN, though it does not appear to learn strategy with only a million steps, and just blindly pushes to the opponent zone.

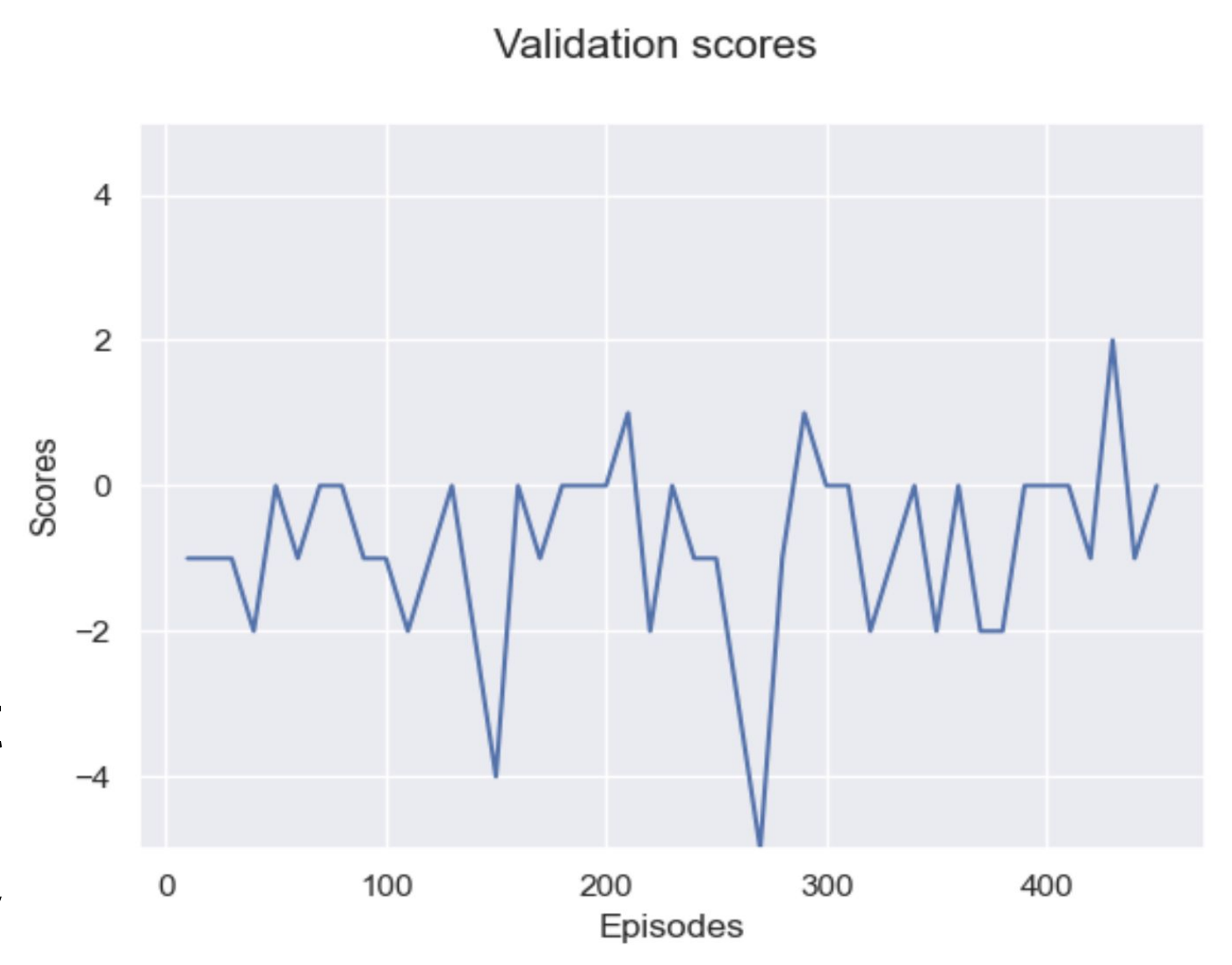


Figure 3. Validation Scores for the PPO trained model

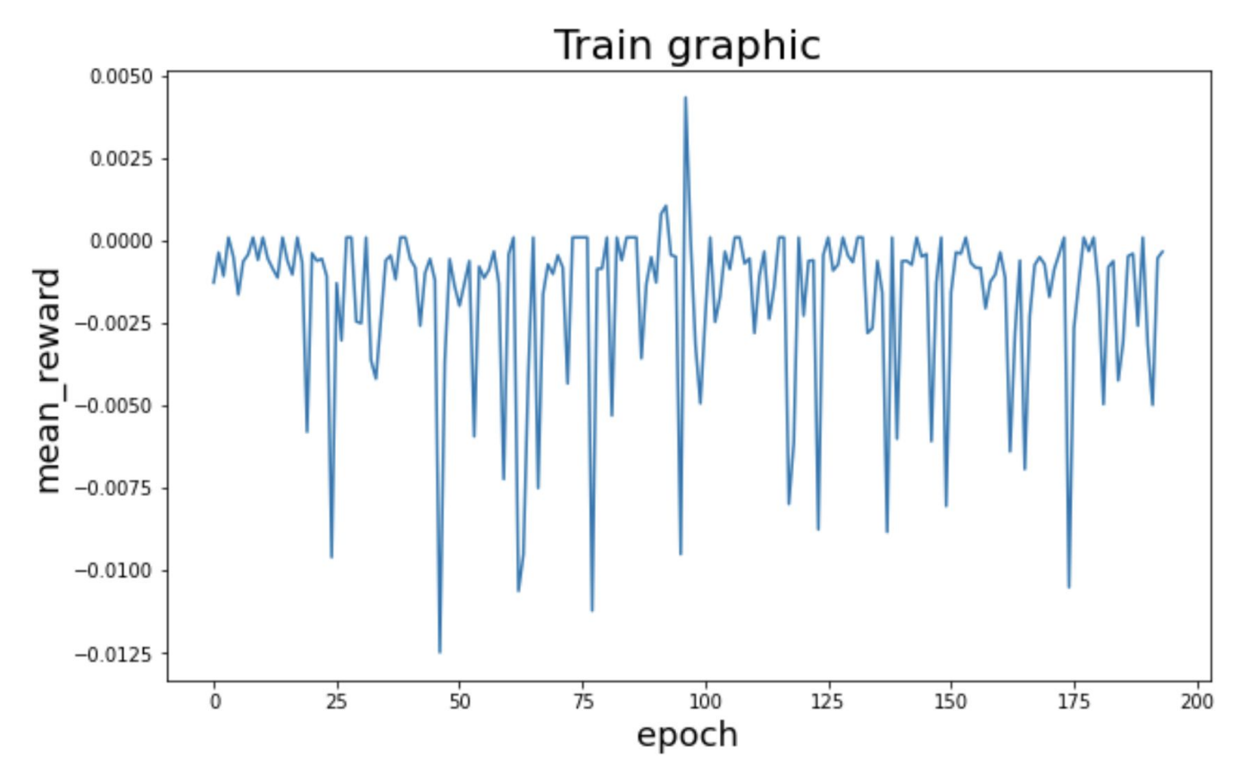


Figure 4. Mean Rewards for the A3C trained model

The A3C model:

trained for 64000 timesteps but was unable to fully learn from the football environment.

The TRPO model:

trained for multiple scenarios, 415 episodes or 250000 steps

Compared to other agents, it took more time to converge.

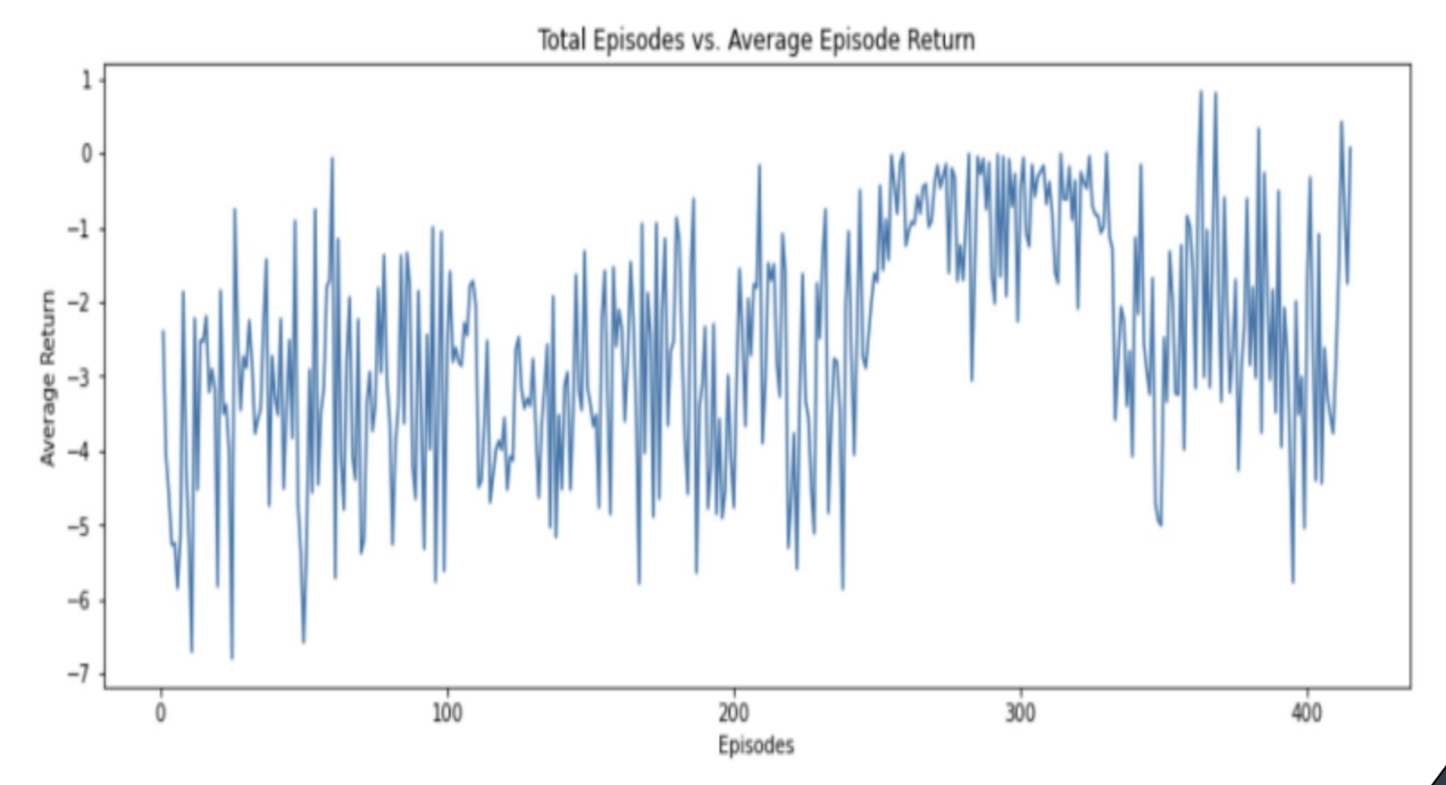


Figure 5. Average episode rewards for the TRPO model

Discussion and Future Approach

- Using goals as rewards creates sparse rewards
  - Possible solution: using "academy scenarios".
- Define new rewards - e.x giving positive rewards every time it steals the ball.
- Train against a more skilled agent
- More effective training only when agents are trained for over 50 million timestamps. Computational constraints create issues.