DRF: Dense Reward Reinforcement Learning Framework based on Large Language Models

Jin Ziyuan [1] Cui Hanjia[1] Li Bowei[1]

[1]ShanghaiTech University

Abstraction

In Multi-agent reinforcement learning, complex environments like soccer games often struggle with sparse rewards, which slows down learning. We propose using large language models (LLMs) to create dense, structured reward functions from natural language descriptions of these scenarios. Through iterative human-machine collaboration, this approach speeds up training and improves policy quality. We tested this method in a 3v3 soccer environment using Unity ML-Agents, showing that it leads to more efficient training and better policy performance compared to sparse or manually designed rewards.

Introduction

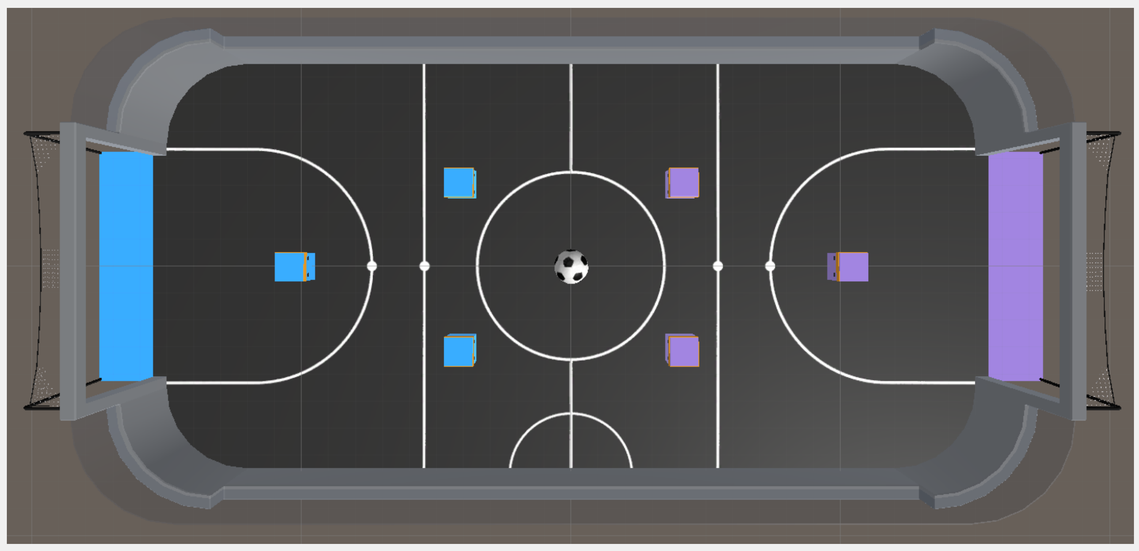
Sparse rewards hinder efficient reinforcement learning, especially in complex multi-agent systems. For instance, in multi-agent soccer, agents receive feedback only when a goal is scored or conceded, making it difficult to learn coordinated behaviors. Manually designing effective dense reward functions is time-consuming, requires deep domain and RL expertise, and risks introducing bias or unintended incentives if done poorly. To address this, we explore using large language models to automate dense reward function design.

Motivation

- Sparse rewards cause slow learning.
- Agents struggle with credit assignment and coordination.
- Manual reward design is time-consuming and suboptimal.

Environment

A prominent testbed is the Unity ML-Agents 3v3 Soccer environment, where two teams of three agents play a simulated soccer match.

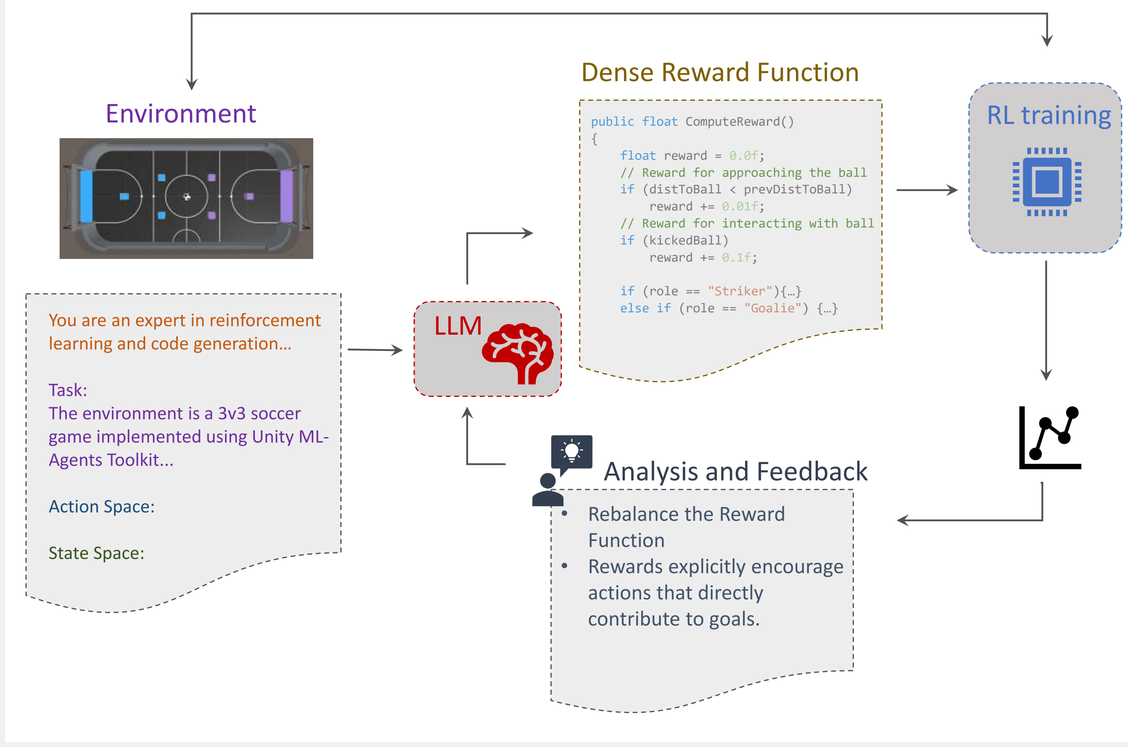


- **Goal:** Get the ball into the opponent's goal while preventing the ball from entering own goal.
- **Observation Space:** Each agent have a set of ray sensors detecting nearby objects.
- **Actions:** 3 discrete branched actions corresponding to forward, backward, sideways movement, as well as rotation.
- **Reward function (initial)**: 1, minus time penalty, when ball enter opponent's goal; -1 when ball enters team's goal.
- **Evaluation**: ELO scoring mechanism.

Setup

- Proximal Policy Optimization (PPO) algorithm is selected as the basic algorithm for agent model training and learning.
- We do not use any pre-trained model, instead training via self-play.

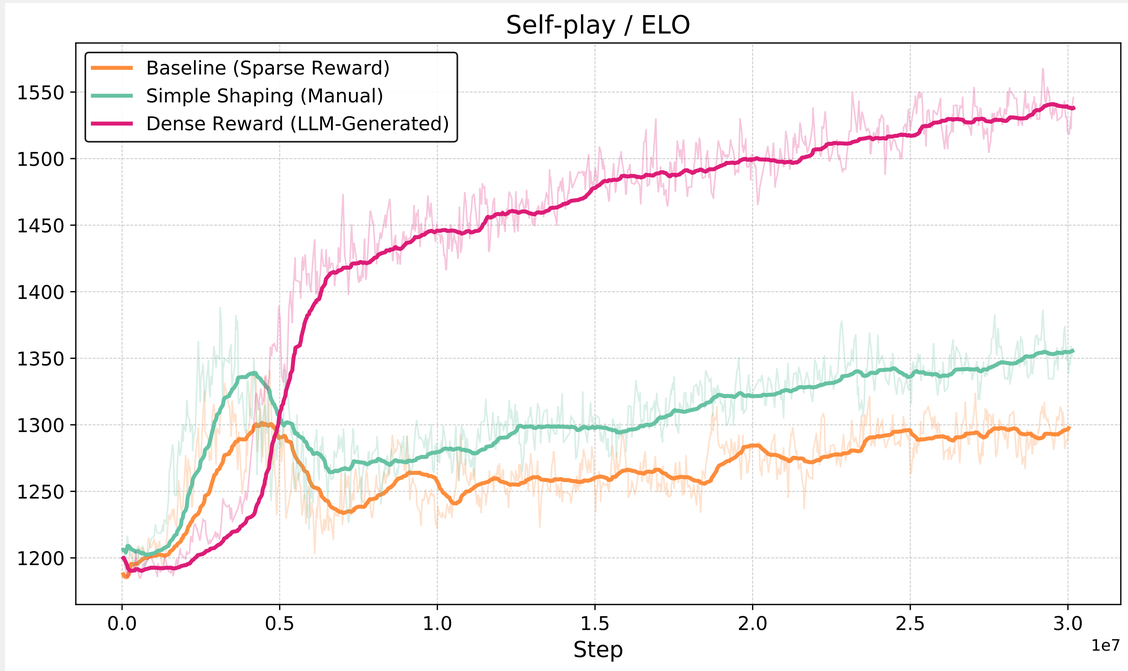
Methods



- We use a large language model to generate a custom reward function in code, informed by a detailed description of the environment and a high-level description of desirable behaviors in soccer.
- We review the LLM's generated code and test it in the environment with a few rollout simulations.
- We leverages the LLM's ability to incorporate human instructions to refine the reward design.

Experiments

Our results show that agents trained with the LLM-derived dense rewards learn faster and achieve higher performance.



Takeaways

- **LLM-generated dense rewards** effectively overcome sparse reward limitations, yielding more efficient training and better performance.
- **Human-in-the-loop refinement** is critical for ensuring reward function alignment.

Future Directions

- Replicate on smaller scale model
- Incorporate a memory storage module and a reflection mechanism