Reinforcement Learning Agent for Google Research Football with Manchester City F.C. Competition

J. D. Escallón Guzmán*, J. D. Lozano Luna*, J. E. Muñoz Gómez*
*Department of Computer Engineering
Universidad Distrital Francisco José de Caldas
Emails: {jdescallong, judlozanol, jemunozg}@udistrital.edu.co

***Abstract*—In this article, we present the design and implementation of a reinforcement learning agent for the Kaggle competition ”Google Research Football with Manchester City F.C.” [1] A systemic analysis approach was applied as a method to better understand the complex dynamics of the simulation environment. The proposed solution includes a modular architecture, which is composed of input/output adapters, an action selection module based on the PPO algorithm, a contextual reward system, and a fallback mechanism to manage performance degradation during the agent's training. Due to the high sensitivity and unpredictability of the environment's interactions, it presents a high level of chaos, which was addressed through a reward strategy designed to adapt to different game contexts. When structuring the training process, an approach was taken that rewards desired behaviors and penalizes harmful ones, it was achieved to guide the agent toward strategic and adaptable decision-making. This study evidences the integration of systemic thinking in the design of RL agents allows to develop robust solutions capable of performing in chaotic environments.**

I. Introduction

The rapid advance of IA and ML techniques has favored the appearance of agents capable of performing in dynamic and complex tasks. One of the platforms that stands out to explore this is kaggle, the competitions of this platform challenge its participants to develop IA agents to compete. Among them is the competition ”Google Research Football with Manchester City F.C.” gives an opportunity to design an agent capable of playing football in a simulated environment.

In this competition, the participants must present their IA agent capable of controlling a soccer team in a turn-based simulation. Each agent can control only one player at a time, generally the one who has the ball or the one closest to it. The environment developed by Google Research [3], is based on real soccer rules [2] with some exceptions, such as the fixed duration of matches of 3,000 steps, the absence of side changes, substitutions, extra time or penalty shootouts.

This project took a systemic analysis approach to break down the competitive environment into its simplest and most fundamental parts, such as its inputs, processes and outputs with the goal of understanding the chaotic nature of the game. This methodology allows us to identify critical information flows, as well as important requirements to build an optimal IA agent. From this analysis, a modular solution was designed with components capable of interpreting observations, selecting better actions and adapting to the changing contexts of the environment.

The core of the solution focuses on the integration of a PPO reinforcement learning algorithm with a customized reward system, Promotes realistic and effective soccer strategies. With the purpose of mitigating potential instabilities during training a backup system was implemented to preserve the most stable versions of the model. This article documents the entire design process, strategy, implementation and results obtained from the agent, highlighting benefits of applying systemic thinking in the development of IA-based solutions.

II. Problem Context and Systemic Analysis

The competition ”Google Research Football with Manchester City F.C.” [1], organized by Google proposes a simulation environment where IA agents must control players in simulated matches. Each agent controls a single player per team at each instant of the game, prioritizes the player who possesses the ball or the closest one to it if the team doesn't have ball possession. The game rules resemble real soccer, with some simplifications like the absence of substitutions, side changes or extra times. The matches have a fixed duration of 3,000 steps, defined by each observation sent and action taken.

From a systems analysis perspective, this environment can be interpreted as a complex system, is composed by inputs (observations), processes (agent's decision-making) and outputs (environment actions). this structure allows a modular presentation of the system, where two subsystems can be identified, The ranking system and the match system.

The environment presents a highly chaotic and sensitive behavior to small variations. situations like random bounces, defensive mistakes or changes in offensive strategies can easily alter a match's outcome. This sensitivity level makes difficult the evaluation of strategies adopted by the agents, since small differences in decisions can produce distinct results.

The most critical information flows are the observations received by the agent. These contain essential information about positions, roles, game states and ball possession. A proper interpretation of this data is key for effective decision-making. Having stated this, a solution is required that can adapt to the dynamic nature of the environment, correctly interpret the information and execute appropriate actions to maximize team performance.

III. Design of the Modular Solution

Based on the systematic analysis carried out, a modular solution was designed capable of facing the complexities of the simulated environment and the competition requirements. This architecture is composed of several modules that work jointly.

A. Input Adapter

The environment allows using different presentations for the observations like raw, simple115_v2 and pixels [3]. To guarantee the compatibility and stability of the system, an input adapter was developed to standardize these observations into a more common structure, facilitating its use by the agent.

B. Action Selection

The core makes decisions based on the PPO algorithm, which allows the agent to learn effective action policies, together with the experience accumulated during the simulations. In each step the agent evaluates the game state and decides among 19 available actions, these actions are then transformed by the output adapter and executed in the simulator.

C. Reward System

Given the chaotic and sensitive nature of the simulation environment, a detailed reward system was designed. This system assigns positive or negative values to the agent's actions depending on multiple factors, such as the player's role, their position on the field, and ball possession. This way, behavior consistent with the game is promoted, such as rewarding successful passes or shots on goal, and penalizing ball losses or shots from ineffective distances.

D. Model Backup System

During the training process there exists a risk that the agent may learn suboptimal behaviors or fall into cycles of undesired decisions. To mitigate this, a backup system was incorporated, this saves stable versions of the model in case a regression in its performance is detected, it will be possible to restore a previous version and continue training from a point preserving the progress achieved.

IV. Reward System Strategy

The reward system is established as the central component to guide the agent's learning. it was developed as a reward strategy that not only responds to specific events like scoring a goal or losing the ball, but also takes into account the game context, player position and game phase.

A. Game Phases

For dynamic adaptation, the reward system was divided into five phases determined by ball possession and field position:

- **Attack:** the team will have possession of the ball in the opponent's field
- **Defense:** the opponent will have the ball near our goal
- **Build-up:** the team will have possession of the ball in our own field
- **Transition:** these will be intermediate situations
- **High press:** the opponent will have the ball in their own field

Each phase influences the type and magnitude of rewards associated with each action, rewarding appropriate behaviors according to the data flow.

B. Rewards per Action

Specific rewards and penalties were established associated with actions/events such as:

- Shots on goal have a positive reward,
- Ball losses from failed passes or out-of-bounds deliveries are penalized,
- Recoveries through interceptions or opponent errors give a reward,
- Successful passes especially forward passes give a reward,
- Excessively retaining the ball without passing generates a penalty

C. Player Role Consideration

The system also takes into account the player's position, since responsibilities and expected behavior vary according to the agent's role. For example, defenders are penalized for leaving their zone or allowing goals, while forwards are rewarded for creating scoring opportunities.

D. Emergent Behaviors and Adjustments

During training, some unwanted emergent behaviors were identified, resulting from imbalances in the reward system.

The agent learned to stay in specific areas of the field to maximize rewards while ignoring the ball. It also detected that long backward passes weren't penalized, so it exploited this logic with inefficient consequences between the goalkeeper and forwards. In simplified scenarios like 1v0, the agent developed effective behaviors that later proved inefficient in 11v11 matches - such as shooting immediately upon receiving the ball.

These observations enabled adjustments to the reward system, incorporating necessary penalties such as unnecessary long-range shots and balancing passing and possession incentives.

V. Results and Experiments

To validate the effectiveness of the proposed solution, iterative trainings were carried out in which rewards were actively adjusted, the emergent behaviors of the agent were analyzed. This section presents the main findings during the experimental development.

A. Training Process

The agent was trained using the Google Research Football environment [3] with the full 11vs11 scenario. The PPO algorithm was employed, implemented through the TF-Agents library of TensorFlow [7]. The training was conducted in several sessions within a Docker environment, ensuring replicability and control over system dependencies.

During training, backup mechanisms were used to store stable versions of the model. This allowed experimenting with different reward system configurations, avoiding compromising the overall learning progress. When unwanted emergent behaviors, deviations or deteriorations in the agent's behavior were detected, the last stable version of the model was restored.

B. Evolution of the Reward System

The initial design of the reward system generated several efficient behaviors in simplified scenarios, but did not generalize adequately to the complete environment. Throughout the training numerous adjustments were made, among them:

- Penalization for shots from long distances
- Differentiated rewards by pass type
- Punishment for prolonged possession

These adjustments allowed the agent to develop balanced strategies and coherent with the game.

C. Agent Behavior Observation

Upon completing the training process, agent evaluations were conducted under conditions similar to the competition environment. Positive behaviors were observed such as:

- Decision-making based on ball location and assigned role
- Ability to press defensively and recover the ball in opponent territory
- Generation of effective passing sequences and shots in favorable conditions

However, limitations were still detected in high-pressure situations or when match context changed abruptly.

D. Final Evaluation

To assess overall performance, the agent was matched against previous versions of itself and against baseline agents from the environment [3]. The results demonstrated consistent improvement in win rates and more stable, realistic behavior.

VI. Conclusions

The development of an IA agent capable of playing soccer in a simulated environment, requires not only technical knowledge in RL but also a deep understanding of the system in which it is inserted. In this project it was demonstrated how the use of systemic thinking allows decomposing a chaotic environment into fundamental elements, thus facilitating the design of a modular and effective solution.

The use of the Proximal Policy Optimization (PPO) algorithm, combined with a well-contextualized reward system, allowed the agent to learn behaviors coherent with the game. through multiple cycles of training evaluation and adjustments, the system managed to overcome common problems, unwanted emergent behaviors and the over-optimization of specific actions. The integration of a backup mechanism proved key for preserving progress and free experimentation on the agent.

The designed solution demonstrated functional performance, with adaptive behaviors and aligned with the objectives. However the environment still presents quite a few challenges, especially regarding the agent's ability to generalize against abrupt changes in the match. This opens future lines of work focused on multi-agent learning, long-term tactical planning and incorporation of more sophisticated exploration mechanisms.

References

[1] Google, "Google Research Football with Manchester City F.C. – Kaggle Competition," Kaggle, 2020. [Online]. Available: https://www.kaggle.com/competitions/google-football/overview

[2] R. of Sport, "Football rules," Rules of Sport. [Online]. Available: https://www.rulesofsport.com/sports/football.html

[3] Google Research, "Football Environment Documentation," GitHub - google-research/football. [Online]. Available: https://github.com/google-research/football/blob/master/gfootball/doc/observation.md

[4] ——, "Google Football Scenarios," GitHub - google-research/football. [Online]. Available: https://github.com/google-research/football/tree/master/gfootball/scenarios

[5] Y. Song et al., "An Empirical Study on Google Research Football Multi-Agent Scenarios," Machine Intelligence Research, vol. 21, no. 3, pp. 549–570, 2024.

[6] Kaggle, "Google Research Football with Manchester City F.C. – 2nd Place Discussion," Kaggle Discussions, 2021. [Online]. Available: https://www.kaggle.com/competitions/google-football/discussion/202977

[7] TensorFlow, "TF-Agents: A reliable, scalable, and easy-to-use TensorFlow library for reinforcement learning," TensorFlow. [Online]. Available: https://www.tensorflow.org/agents

[8] J. D. Escallón Guzmán, J. D. Lozano Luna, and J. E. Muñoz Gómez, "Systems Analysis – Workshop Reports," GitHub Repository, 2025. [Online]. Available: https://github.com/judlozanol/Systems-Analysis