# Exploring Reinforcement Learning with Soccer Simulations

##### **Anthony Shenouda**

Computing & Software 736, McMaster University

January 3rd, 2023

https://github.com/AnthonyShenoud/Exploring-RL-within-Soccer

***Abstract*-** The realm of soccer is heavily invested in player research, development, scouting, and various other facets, contributing billions annually. With technological advancements, teams increasingly turn to dynamic programming techniques to model gameplay, strategies, and tactics. This study shares a pipeline for simulating soccer matches, offering outcomes and insights into individual player contributions. Leveraging Reinforcement Learning (RL) techniques, the investigation aims to discern the attributes pivotal in enhancing overall team performance. Although in its beginning stages, this pipeline establishes a groundwork for more intricate techniques and methodologies.

T

1. Introduction

his project begins by refining and adapting a simulation environment initially developed by a French programmer to suit the scope of this study. The primary objective is to establish an accessible pipeline that facilitates the application and testing of Reinforcement Learning (RL) techniques within the domain of soccer. This specialized environment contains two teams, each composed of 11 players. Its customizable features include alterations to team tactics, simulating the impact of training on player statistics, player substitutions, and real-time assessment of their individual and collective team performance.

The goal of the model developed for this study is to discern optimal strategies for improving the team, and ultimately secure three consecutive wins against the opposing team. This derives from the common format in many soccer competitions where two-legged matches occur, with victory determined by the team with more wins or goal differentials. Achieving success in three consecutive simulations indicates consistency and a heightened probability of real-world success.

Validation and reliability assessments have been conducted through various simulations to test the environment. This includes random checks such as playing without a goalie to test the impact of conceding goals. Additionally, statistically worse teams should correspond with expected outcomes, ensuring the credibility of the simulation environment. The RL framework actions involve maintaining the existing team or enhancing either offence or defence then observing the subsequent outcomes. The implementation incorporates a Q-learning agent and subsequently trains the Q-learning agent.

This problem space is complex, requiring the simplification of the environment into manageable components to conduct RL. The significance of this work lies in its potential to offer insights into player selection strategies, identify deficiencies in team skills (be it offensive or defensive), and serve as an environment for assessing team performance against various opponents using diverse tactics.

1. Related Works
   1. Introduction of Original Simulation Environment:

A recent French programmer (who has opted not to share their name) laid the groundwork for simulating soccer matches [Mycode-Developpement]. Their work is a deterministic model, where predefined rules govern player actions with some randomness. However, this proved useful for the case of this study as a level of consistency is important to determine the accuracy of results.

* 1. Limitations

Building upon the programmer’s foundation, this research addresses several key limitations to enhance the usability of the simulation environment [Mycode-Developpement]. Due to the lack of developers and a professional workplace, their code was susceptible to various bugs, lacked usability (French naming of variables), and needed several modifications to suit the needs of this study. Therefore, the work in this study included bug fixes, the ability to alter player and team attributes, and running thousands of simulations while keeping track of player and team performance.

* 1. Divergent goals

The original simulation environment was designed to give the user real-time results of a single game for entertainment purposes. Although visually appealing, this was both taxing and needed to be more efficient to perform thousands of simulations in seconds. Time wasn’t the only refactoring required. The model needed to be able to communicate with the MATLAB workflow that would send various modifications to the teams and must be able to feed the impacts of the modifications back. This begins the work of reinforcement learning.

* 1. Other studies

Thoughts from the paper, *Deep Soccer Analytics: Learning an action-value function for evaluating soccer players*, are used throughout this study. As opposed to the work mentioned above, the work found in this paper is superior in complexity, application, and research. Using various Deep Reinforcement Learning techniques in soccer backed by mathematical proofs, their work is often cited and used in both design and implementation decisions.

1. Proposed Solution
   1. The objective

The goal of this solution is to take actions to aid a weaker team, Team 1 achieves three consecutive wins against Team 2. To start, when running 10,000 simulations, Team 1 achieved victory 25% of the time. The three actions that can be taken are, to do nothing, improve the team’s offensive rating, or improve the team’s defensive rating. For the sake of simplicity, that is the experiment discussed in this paper. However, this pipeline can handle making far more detailed actions such as improving a specific attribute for a specific player and learning the outcome. The result is 12 states where there are various combinations of maintaining the current team or improving it.

A diagram of a diagram

Description automatically generated with medium confidence

Figure 1 State Transition

* 1. The Dataset, Parsing, and data cleaning

Work the author, Anthony Shenouda, completed before and is used is outlined in this section:

The dataset (Figure 2) utilized for this study serves as a comprehensive repository sourced directly from the official FIFA website. It encompasses a vast array of over 50 player attributes ranging from traits such as shot power, passing, and dribbling and their team that are fed into the environment. Collecting data from the most recent 2023-2024 season across the top five European leagues, this dataset encapsulates the profiles of nearly 3000 players, offering a rich pool of talent from the world’s most elite soccer competitions.

A screenshot of a computer

Description automatically generated

Figure 2 FIFA Dataset

The procedure for acquiring, parsing, and processing the dataset starts with direct extraction from the official FIFA website, leveraging Python's Beautiful Soup web-scraping library. Post-extraction, that data is structured and organized in an Excel file. The combination of web-scraping techniques and Excel storage ensured no values were lost. However, issues with duplication, currency conversions, and improperly formatted values were next to tackle.

The data cleaning phase aims at refining the acquired dataset to ensure its accuracy and suitability for analysis. The process begins by employing Pandas' duplicate function which detects and eliminates any redundant entries. The next issue was with the attribute values. Throughout the season, players would improve or underperform, and their characteristics would be updated accordingly. This was represented by the current weighting followed by a ‘+’ or ‘-’ number representing the change (i.e., 84+1 or 76-3). Another function was developed to rectify this issue.

The final player data and their teams are extracted and used in the simulation environment. This allows for the most accurate game expectations using the most recent data.

* 1. Setting up a Simulation environment for reinforcement learning

Using the simulation as the foundation for the environment, modifications mentioned in Related Work, Divergent Goals are implemented in this step. Using modern coding practices, as well as removing graphics and delays are done to improve efficiency. Time wasn’t the only refactoring completed. Next modifications to the environment are made take in inputs from MATLAB and output the results required results back in real time.

* 1. reward design choice

In [Deep Soccer Analytics], a player is given a reward every time they score. As this study focuses on team performance as opposed to individual performance, a reward of 1 is given every time the team wins, and -1 every time the team loses. The reason for the negative reward is to punish teams for missing out on a potential win to simulate real-world behaviour.

* 1. Setting up and configuring Q-Learning agent

Creating a Q-learning agent involves several key steps to ensure effective learning and decision-making within the simulated environment. [CreateMDP. MATLAB]

obsInfo = getObservationInfo(env);

actInfo = getActionInfo(env);

qTable = rlTable(obsInfo, actInfo);

qFunction = rlQValueFunction(qTable, obsInfo, actInfo);

qOptions = rlOptimizerOptions(LearnRate=1);

Here, *obsInfo* and *actInfo* retrieve observation and action specifications from the environment. The rlTable function constructs a Q-table based on these specifications, and rlQValueFunction creates a Q-value function using this table.

agentOpts = rlQAgentOptions;

agentOpts.DiscountFactor = 1;

agentOpts.EpsilonGreedyExploration.Epsilon = 0.9;

agentOpts.EpsilonGreedyExploration.EpsilonDecay = 0.01;

agentOpts.CriticOptimizerOptions = qOptions;

qAgent = rlQAgent(qFunction, agentOpts);

The rlQAgentOptions function initializes options for the Q-learning agent. The DiscountFactor determines the importance of future rewards, while EpsilonGreedyExploration manages the balance between exploration and exploitation. Setting CriticOptimizerOptions to qOptions ensures the Q-table's learning rate is utilized.

* 1. Train q-learning agent

Training the Q-learning agent involves setting specific training parameters and criteria to ensure effective learning and convergence toward an optimal policy. [CreateMDP. MATLAB]

trainOpts = rlTrainingOptions;

trainOpts.MaxStepsPerEpisode = 50;

trainOpts.MaxEpisodes = 100;

trainOpts.StopTrainingCriteria = "AverageReward";

trainOpts.StopTrainingValue = 5;

trainOpts.ScoreAveragingWindowLength = 20;

MaxStepsPerEpisode: Specifies the maximum number of time steps allowed per episode. This determines how long the agent interacts with the environment before the episode ends.

MaxEpisodes: Defines the maximum number of episodes for which the agent will be trained. Each episode consists of a series of interactions with the environment.

StopTrainingCriteria and StopTrainingValue: These parameters set the criteria for stopping the training process. In this case, the training will stop when the average cumulative reward across 20 consecutive episodes exceeds 5. Adjusting this value can influence how quickly or thoroughly the agent learns the optimal policy.

ScoreAveragingWindowLength: Specifies the window length used to compute the average reward. It determines how many episodes are considered when calculating the average reward used for the stopping criteria.

1. Experiments & results
   1. Effects of epsilon and epsilon decay

As illustrated below, as epsilon becomes smaller there becomes less exploration. This makes sense as this is an environment with few actions and harsh punishment for wrong decisions. In a real-world context, teams must be strategic with how they handle risks as taking the incorrect actions will be costly.

A screen shot of a graph

Description automatically generated

Figure 3a Effects of Epsilon = 0.9 & Decay = 0.01

A screen shot of a graph

Description automatically generated

Figure 3b Effects of Epsilon = 0.9 & Decay 0.1

* 1. increasing reward when the team performs better

This model encourages not only winning but also winning by larger margins. This is done by increasing the reward for more decisive wins.

A screen shot of a graph

Description automatically generated

Figure 4 Increasing the Reward for Better Team Performance

1. Conclusions

In summary, this study has ventured into the realm of soccer simulation using reinforcement learning techniques, aiming to enhance team performance and strategic decision-making. Through the development of a pipeline and extensive experimentation, several key findings and learnings have emerged.

Firstly, the successful integration of RL techniques into the simulation environment has showcased the potential for leveraging AI in sports analytics. The ability to simulate games, alter team attributes, and observe resulting performances provides a valuable tool for understanding the impact of different strategies. Throughout this research, several crucial lessons have been learned. The importance of data quality and model efficiency, the tunning of various Q-Learning Parameters and aligning simulation outputs with real-world scenarios has been explored. This study begins the discussion and future work of integrating complex sports environments with RL frameworks.

1. Future work
   * 1. Integration of Advanced RL Techniques:

Future work includes integrating other advanced statistical techniques and machine learning algorithms. Deep reinforcement learning or actor-critic methods or exploring methodologies like neural networks or ensemble learning would enhance the accuracy and allow for increased environmental complexity.

* + 1. increasing state and action pairs:

This paper discussed a simplified version of the model using minimal states and actions. Increasing the number of states by increasing the number of actions taken would provide more real-world application.

References

1. "Chatgpt - Openai." ChatGPT, chat.openai.com/auth/login. Accessed 7 Dec. 2023.
2. FIFPlay. "FIFA 22 Attributes Explained." FIFPlay, www.fifplay.com/fifa-22-player-attributes/.

Accessed 15 Dec. 2023.Liu, G., Luo, Y., Schulte, O., & Kharrat, T. (2020, July 21). Deep Soccer Analytics: Learning an action-value function for evaluating soccer players - data mining and knowledge discovery. SpringerLink. https://link.springer.com/article/10.1007/s10618-020-00705-9

1. "Players."SoFIFA,sofifa.com/players?type=all&lg%5B0%5D=13&1g∞5B1%5D=16&1g%5B2%5D=19&1g%5B3%5D=31 &lg95B4%5D=53&col=vl&sort=desc&showCol%5B%5D=phy. Accessed 11 Nov. 2023.
2. Liu, G., Luo, Y., Schulte, O., & Kharrat, T. (2020, July 21). Deep Soccer Analytics: Learning an action-value function for evaluating soccer players - data mining and knowledge discovery. SpringerLink. https://link.springer.com/article/10.1007/s10618-020-00705-9
3. CreateMDP. MATLAB & Simulink. (n.d.). https://www.mathworks.com/help/reinforcement-learning/ug/train-reinforcement-learning-agent-in-mdp-environment.html
4. Liu, G., Luo, Y., Schulte, O., & Kharrat, T. (2020, July 21). Deep Soccer Analytics: Learning an action-value function for evaluating soccer players - data mining and knowledge discovery. SpringerLink. https://link.springer.com/article/10.1007/s10618-020-00705-9
5. Mycode-Developpement. (n.d.). Mycode-developpement/football-game-simulator: Relive the world’s greatest soccer posters or those from your favorite teams. this is a football match simulator where 11 players on each team play against each other. the simulator is currently in beta version! GitHub. https://github.com/Mycode-Developpement/football-game-simulator
6. Shenouda,Anthonyshenoud/Marketvaluesoccerplayers.GitHub. https://github.com/AnthonyShenoud/MarketValueSoccerPlayers

Authors

**First Author** – Anthony Shenouda, MASc., Software Engineering, McMaster University, shenoa2@mcmaster.ca.

**Correspondence Author** – Dr. Douglas Down, Professor, McMaster University, downd@mcmaster.ca,