Machine Learning
https://doi.org/10.1007/s10994-024-06611-1

In-game soccer outcome prediction with offline reinforcement learning

Pegah Rahimian[1] · **Balazs Mark Mihalyi**[1] · **Laszlo Toka**[1,2]

Received: 5 October 2023 / Revised: 30 April 2024 / Accepted: 4 August 2024

Abstract
Predicting outcomes in soccer is crucial for various stakeholders, including teams, leagues, bettors, the betting industry, media, and fans. With advancements in computer vision, player tracking data has become abundant, leading to the development of sophisticated soccer analytics models. However, existing models often rely solely on spatiotemporal features derived from player tracking data, which may not fully capture the complexities of in-game dynamics. In this paper, we present an end-to-end system that leverages raw event and tracking data to predict both offensive and defensive actions, along with the optimal decision for each game scenario, based solely on historical game data. Our model incorporates the effectiveness of these actions to accurately predict win probabilities at every minute of the game. Experimental results demonstrate the effectiveness of our approach, achieving an accuracy of 87% in predicting offensive and defensive actions. Furthermore, our in-game outcome prediction model exhibits an error rate of 0.1, outperforming counterpart models and bookmakers' odds.

Keywords Temporal graph neural networks · Soccer · In-game win probability · Reinforcement learning

Editors: Philippe Lopes, Werner Dubitzky, Daniel Berrar, Jesse Davis.

✉ Pegah Rahimian
rahimian.pegah@gmail.com

Balazs Mark Mihalyi
balazsmark.mihalyi@edu.bme.hu

Laszlo Toka
toka.laszlo@vik.bme.hu

[1] Budapest University of Technology and Economics, Budapest, Hungary

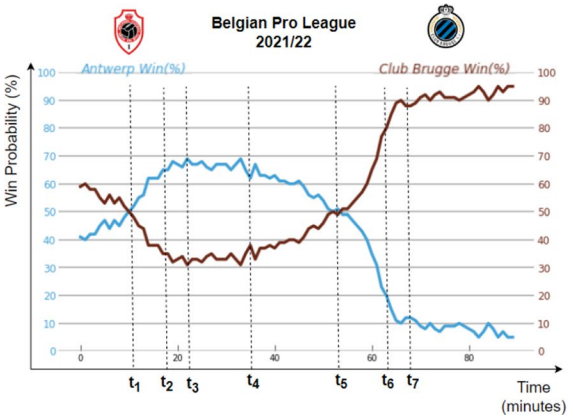
[2] ELKH-BME Cloud Applications Research Group, Budapest, Hungary

Published online: 06 September 2024

1 Introduction

Soccer outcome prediction has gained paramount attention among both industry professionals and researchers. Zion Market Research suggests that the sports betting market was worth over 104 billion dollars in 2017, and will reach 155 billion dollars by 2024.[1] Most of the works on outcome predictions to date have focused on win probabilities of each team prior to the start of the game considering historical goals scored in previous games (e.g., Maher, 1982; Dixon & Coles, 1997; Crowder et al., 2002; Havard & Salvesen, 1997; Dimitris, 2003; Owen, 2011). The other category of works propose solutions for in-game outcome prediction considering features such as score, time-remaining, etc. Simple machine learning models are trained for predicting in-game outcomes in eSport (Yang et al. 2016), and Bayesian approach has been proposed for real soccer match outcome prediction (Robberechts et al. 2021). Our work belongs to the second category of predicting in-game win probabilities at any moment of watching a soccer game. Figure 1 outlines the final output of our deployed system by predicting the win probability of each of the teams Royal Antwerp and Club Brugge in an important game of the Belgian Pro League 2021/22, leading Club Brugge to secure the win title. The table on the right hand side lists game highlights proving that our approach could correctly capture the special moments of the game. The evolution of probabilities illustrates how each team started the game and how they progressed throughout the match with poor performance of Brugge in the first half, then improving in the second half.

In order to estimate such an accurate prediction for the in-game outcomes, we had to deal with several challenges. First, among all sports, soccer is a relatively long game with many players and various decisions are challenging to measure and evaluate due to its low-scoring (i.e., sparse reward), complex and highly dynamic nature. Most decisions have little immediate impact but may positively contribute to the team winning on the long run. For instance, a simple short pass in the midfield may open up valuable space elsewhere on the pitch for a teammate. Like Johan Cruyff once said: "Sometimes something's got to



Time	Event	Score
t1	Antwerp missed goal	0-0
t2	Antwerp scored goal	1-0
t3	Antwerp missed goal	1-0
t4	Brugge missed goal	1-0
t5	Brugge scored goal	1-1
t6	Brugge scored goal	1-2
t7	Brugge scored goal	1-3

Fig. 1 In-game win probability in a game between Royal Antwerp and Club Brugge (end result: 3-1) in 2021/22 season of Belgian Pro League

[1] https://mercurius.io/en/learn/predicting-forecasting-football.

happen before something is going to happen". Furthermore, soccer tracking data is highly unstructured and the way in which the state of a game is represented is central to aforementioned soccer analytics efforts. Dealing with soccer tracking data requires algorithms that are able to handle sequential decision making, permutation invariance, sparse rewarding, interactions between the players and the ball. To address all challenges, we employ the Temporal Graph Network (TGN) as the underlying neural architecture. The TGN comprises an encoder that captures the spatial and temporal characteristics of event and tracking data. Specifically, it encodes the complex interactions between players and the ball over time. We utilize a decoder component to predict offensive and defensive actions based on the encoded representations. To optimize the decision-making process, we apply Conservative Q-Learning (CQL) to the TGN. This enables us to learn an optimal policy for extracting the most effective offensive and defensive actions at each time-step during soccer matches. To do so, we design a Markov Decision Process (MDP) environment from soccer data and train the Soccer Network with a version of offline Reinforcement Learning (RL) algorithm to infer the optimal actions from the observations in the soccer dataset, and utilize the difference between the Q-value of actual and optimal actions as a crucial component of outcome prediction. We show how our novel approach of differentiation between actual and optimal actions improves the accuracy of win prediction in real-time. To the best of our knowledge, our work is the first application of offline RL to measure the difference between actual and optimal decisions solely from historical dataset and to utilize it for improving in-game outcome prediction.

This paper is organized as follows: Sect. 2 provides a comprehensive review of recent advancements in graph neural networks, action prediction, RL, and outcome prediction in the context of soccer. In Sect. 3, we detail our methodology for representing tracking data in a format conducive to modeling TGNs. We elucidate how this enables us to predict the most probable offensive and defensive actions based on historical behavioral data. Section 4 delves into our novel application of offline RL, which allows us to extract strategic insights solely from historical data. We demonstrate how this technique enhances our ability to assess teams' performance in terms of offensive and defensive actions while maximizing expected goals. Building on this, Sect. 5 elucidates how we leverage the acquired optimal strategies to accurately predict in-game outcomes in soccer. Section 6 presents the experimental results and evaluation of our approach, focusing on action prediction and in-game outcome prediction, providing insights into its efficacy and performance. Finally, Sect. 7 concludes our work.

2 Related work

Forecasting soccer match results has long captivated the interest of sports scientists, researchers, soccer enthusiasts, and bettors alike. Numerous statistical and machine learning methodologies have been proposed by researchers to enhance the accuracy of predictions. However, soccer is a sport characterized by its complexity, and predicting its outcome cannot rely solely on analyzing historical game results. This paper advocates for a novel approach that focuses on analyzing in-game actions. By assessing the deviation of actual team actions from the optimal actions that could have been taken to maximize the probability of winning, we aim to develop a highly accurate model capable of estimating the probability of victory or defeat at any given moment during a soccer match. To achieve this goal, our modeling framework incorporates the following key components: Graph

Neural Network for modeling the temporal aspect and predicting the subsequent action, RL for evaluating actions and quantifying their deviation from the optimal course of action, and Match Outcome Prediction. In this section, we delve into the state-of-the-art methods employed for each of these components.

2.1 Graph neural networks in soccer

Graph representations offer a natural approach to modeling player and ball interactions in sports. Typically, players and the ball are depicted as nodes within a fully connected graph, where edge weights capture their interactions and are refined during the training phase. For instance, Yeh et al. (2019) advocate for utilizing graph neural networks (GNNs), which inherently excel at capturing coordinated behaviors due to their ability to remain invariant to input permutations. In their work, they introduce a graph variational recurrent neural network tailored for predicting the future positions of players in soccer and basketball. Similarly, Yedid (2017) and Kipf et al. (2018) introduce attention mechanisms within a graph framework to learn player trajectories. Graph neural networks have found widespread applications in modeling structured or relational data, as highlighted in Battaglia et al. (2018). In scenarios where the data exhibits sequential characteristics, graph recurrent neural networks (GRNNs) have emerged as a popular choice. For example, Sanchez-Gonzalez et al. (2018) propose a fusion of graph representations with recurrent layers, employing techniques like gated recurrent units (GRUs) as demonstrated by Cho et al. (2018). Annother innovative action rate model has been introduced by Dick and Brefeld (2023) where they represent players and ball by a fully connected graph and resort to graph recurrent neural networks. They focus on predicting passes and when, in time, the pass will be played. At the same time, their model estimates the probability that the player loses possession of the ball before she can perform the action. A similar application of graph neural networks simialr to our paper has been introduced by Brandt and Brefeld (2015). They present a simple pass-based representation that is subsequently used together with the PageRank algorithm to identify the importance of the players. They then Aggregate player scores to team values for predicting the wining chance of the teams. A semi-supervised approach of graph neural networks has been utilized by Anzer et al. (2022) to detect tacticaal pattern in soccer. This approach works well on applications where require a reduced labeling effort, poses a huge benefit for practical applications. With regards to measuring defensive performance in soccer, Stöckl et al. (2021) resort to a convolutional graph neural networks trained on tracking data to measure viable metrics such as expected pass success, expected threat, expected receiver, and measuring defensive performance. A similar usage of graphs to our model in predicting outcomes in sports has been introduced by Xenopoulos and Silva (2021) where they show how their approach can be used to answer ”what if” questions. Lastly, the most famous application of graph neural networks in soccer these days called ”TacticAI” has been introduced by Google DeepMind and Liverpool FC to offer an effective corner kick retrieval system. This AI football tactics assistant meticulously developed and rigorously evaluated in collaboration with domain experts from Liverpool FC, stands out as a groundbreaking innovation in the field. The study showcases that TacticAI's model suggestions not only closely mirror real-world tactics but also outshine existing strategies in preference, garnering a remarkable 90% favorability rate. Furthermore, the trajectories of players can be predicted with a graph variational recurrent neural network that can accurately model the relationship between players and predict the long-term trajectory, as proposed by Teranishi et al. (2022). Additionally, TacticAI introduces an efficient corner

kick retrieval system, further underlining its practical utility and effectiveness on the field (Wang et al. 2023).

This paper builds upon our earlier research, which explored the application of TGN in predicting pass outcomes (Rahimian et al. 2023). The TGN component utilized here closely resembles our previous work, leveraging its temporal and permutation invariant properties to accurately identify the most likely and intended recipients of passes, as well as predict pass success. However, in this study, we extend beyond pass prediction to encompass a broader scope of offensive and defensive actions throughout the game. Furthermore, we employ RL techniques on this graph structure to derive optimal actions, which are subsequently employed for match outcome prediction. This extension marks a significant advancement in our exploration of TGN's capabilities within the context of soccer analytics.

2.2 Transformer-based action prediction models in soccer

In soccer analytics, transformer-based action prediction models have emerged as a cutting-edge approach to forecasting player actions and interactions on the field. Inspired by their success in natural language processing tasks, transformers excel at capturing long-range dependencies and contextual information, making them well-suited for sequential data analysis in soccer matches. These models leverage self-attention mechanisms to weigh the importance of different spatial and temporal aspects of the game, allowing them to effectively predict player movements, passes, shots, and defensive actions. By processing the entire sequence of past events in a match, transformer-based models can anticipate future actions with remarkable accuracy, providing invaluable insights for coaches, analysts, and decision-makers in the soccer industry. In this regards, the Seq2Event model has been proposed by Simpson et al. (2022) which they predict the next match event given the past match events and context. They also propose a metric creation using a general purpose context-aware model as a deployable practical application, and demonstrate development of the poss-util metric using a Seq2Event model. Utilizing the wealth of football match event data now available, analysts and researchers seek to develop new performance metrics and gain insights into key performance indicators. However, traditional approaches to modeling sequential sports events and evaluating performance metrics may fall short when dealing with large-scale spatiotemporal data, particularly in capturing temporal processes. To address this challenge, Yeung et al. (2023) introduce a Transformer-based neural model tailored for football event data. This model demonstrates superior predictive performance compared to baseline models in our experiments. Additionally, they propose a holistic metric, the possession utilization score (HPUS), for comprehensive analysis of football possessions. They validate its effectiveness by examining its relationship with team performance indicators over a season, such as final rankings, average goals scored, and expected goals (xG). Their findings underscore the significant correlations observed with average HPUS, illustrating its utility in analyzing possessions, matches, and inter-match dynamics.

2.3 Reinforcement learning in soccer

In the realm of soccer analytics, RL finds diverse applications encompassing action valuation, action optimization, and the development of RL agents capable of navigating the field autonomously. Traditionally, RL has been employed to assess the value of on-ball actions through the estimation of Q-functions or policy functions (Liu & Schulte 2018; Liu et al.

2020; Routley & Schulte 2015; Dick & Brefeld 2019). However, existing studies often treat teams as single agents and neglect the valuation of off-ball players across all time steps, particularly in the absence of events. In the domain of inverse RL, research has focused on the estimation of reward functions (Rahimian & Toka 2021; Luo et al. 2020; Muelling et al. 2013). Meanwhile, efforts to estimate policy functions have involved trajectory prediction via imitation learning (Teranishi et al. 2020) and behavioral modeling, aiming to replicate policies using neural networks rather than optimizing them. An innovative application of RL lies in action optimization, treating RL as a control problem to determine optimal actions aimed at maximizing scoring opportunities and securing victories. However, training such models in soccer analytics presents challenges due to the fixed nature of historical data, limiting the ability to alter players' actions in past games. Nevertheless, recent advancements in offline RL offer promising solutions by extracting optimal strategies from static historical game data. Various models of offline RL, such as Off-Policy Policy Gradient, Marginalized Importance Sampling, Conservative Q-Learning, and Pessimistic Value Functions, have been proposed to address distributional shifts encountered in historical game data (Levine et al. 2020). Specifically, Rahimian et al. (2021, 2022); Rahimian and Toka (2023); Rahimian et al. (2023) pioneered the application of off-policy Policy Gradient methods in soccer to optimize offensive and defensive actions, aiming to maximize scoring opportunities and secure victories. In these endeavors, the choice of neural network architecture for predicting behavioral actions, as well as the design of reward functions, play pivotal roles in achieving effective action optimization. To expedite the training process and streamline reward function design, Conservative Q-Learning -an offline RL approach- has emerged as a viable alternative to off-policy policy gradient methods.

2.4 Match outcome prediction in soccer

Forecasting soccer match results has always been one of the most fascinating activities among sports scientists, researchers, soccer fans, and bettors. In the sports analytics area, researchers are continuously trying to develop novel and highly precise statistical predictive models. The goal of these models is to be able to forecast the outcome of a shot and the final result of a match, which outperforms the prediction of bookmakers. In the literature, there are two types of models to forecast the outcome of soccer matches. The first category is modeling the number of goals scored by a given team as a function of explanatory variables (e.g., own and opponent's strength ranking, home/away team, etc.). Most of these models consider Poisson-type assumptions for the number of goals to be scored, either independent Poissons for the home and the away team or a bivariate Poisson. Some instances of this category are proposed by the following studies: Maher (1982); Dixon and Coles (1997); Crowder et al. (2002); Havard and Salvesen (1997); Dimitris (2003); Owen (2011). The second category focuses right on the outcome, i.e., Win, Draw, Loss. Their final results are limited to the decision of who the winner is, mostly following the ordered probit or ordered logit models, e.g., Goddard and Asimakopoulos (2004); Koning (2001); Forrest and Simmons (2000). Each category has its own pros and cons. A comprehensive evaluation of those has been published in Goddard (2005). They showed that the best performance can be achieved using a hybrid method, which adds the covariates in goal-based models, and predicts the outcome of the match (win/draw/loss) from them. In parallel to the studies that focus only on the historical number of goals scored (e.g., (Koopman & Lit, 2019) with dynamic prediction and Tugbay (2020) with static prediction), more recent works try to predict the outcome probabilities in-game. Several methods

have been proposed to estimate in-game win probability in baseball (Lindsey 1961), basketball (Beuoy 2015; Ganguly & Frank 2018), and soccer (Robberchts et al. 2021). On the industrial side, Opta provided win probabilities during live broadcasting in 2019 (Hopkins 2019), but unfortunately they do not disclose their prediction method. However, none of the aforementioned approaches compute optimal actions solely from the dataset and utilize them as a crucial feature for outcome prediction. Our proposed approach accurately computes optimal decisions and action effectiveness to be used for the result prediction task.

3 Modelling with temporal graph networks

Soccer tracking data consists of unstructured locations of 22 players and the ball on the field at a preset frequency. Analyzing such a dataset is a cumbersome task due to its nature of being spatial, temporal, and permutation invariant. Several methods have been proposed by researchers to deal with tracking data, define structure, and make it suitable for analysis. Some works treat each time step of the data as an image and apply convolutional neural networks (CNNs) to analyze it (e.g., (Fernandez & Born, 2020; Rahimian et al., 2022, 2023)), and others order players using the permutation invariant sorting scheme (e.g., Rahimian et al. 2022; Mehrasa et al. 2018; Rahimian et al. 2021)), in which the ball holder is selected as the anchor in the first position, and the rest of the players are numbered according to their distances to the anchor. A better approach to model interaction between players and the ball is to treat the data as a graph. Dick and Brefeld represent players and ball by a fully connected graph and resort to graph recurrent neural networks (GRNN) (Dick & Brefeld 2023). However, the continuous time difference between actions and player substitutes in different matches are not properly modelled in that work. In order to mitigate these problems, our work proposes using a continuous time dynamic graph, which is a timed list of events that may include node addition or deletion, and node or edge feature transformation. We employ TGN, a generic, efficient framework for deep learning on dynamic graphs represented as sequences of timed events (Rossi et al. 2020). TGN is a neural model for dynamic graphs that can be regarded as an encoder-decoder pair, where an encoder is a function that maps from a dynamic graph to node embeddings, and a decoder takes as input one or more node embeddings and makes a task-specific prediction, e.g., node classification or edge prediction. The contribution of our work is the modification of the decoder part of TGN to make it applicable for soccer action prediction instead of node classification or edge prediction. In this section, we formalize the problem of action prediction using TGN.

3.1 Model definition

Soccer network is a modified version of TGN that is an essential component for predicting future actions given the current game situation. In this study, we define the soccer network as a continuous-time dynamic graph represented as a sequence of time-stamped events and producing the probability of performing each of the offensive (pass, dribble, shot) and defensive (tackle, interception, clearance) actions given a particular game state. Soccer network is modeled as a sequence of events $G = \{x(t1), x(t2), ...\}$, representing addition or change of a node or interaction between a pair of nodes. An event $x(t)$ belongs to one of the following two types: 1) a node-wise event $(v_i(t))$ where i denotes the index of the node and v is the temporal feature vector associated with the event, 2)

an interaction event ($e_{ij}(t)$) between nodes i and j is represented by a temporal edge. The temporal set of nodes and edges are shown by $V(T) = \{i : \exists\, vi(t) \in G, t \in T\}$ and $E(T) = \{(i,j) : \exists\, e_{ij}(t) \in G, t \in T\}$, respectively. A snapshot of the temporal graph G at time t is the multi-graph $G(t) = (V[0,t], E[0,t])$ with $n(t)$ nodes.

The dataset consists of high-resolution spatiotemporal tracking and event data. The tracking data includes the (x, y) coordinates of all 22 players and the ball on the pitch at 25 observations per second. The event data includes on-ball action types such as passes, shots, dribbles, etc., annotated with additional features such as contestants, game period, ball possessor player ID, start and end locations of the ball. We then merge tracking with event data; each record of our merged dataset includes all players and the ball coordinates with their corresponding features for each snapshot, i.e., every 0.04 s.

Nodes are the 11 offensive players (OP), the 11 defensive players (DP), the ball (B), all goal lines (G) as a single node, and all touchlines (T) as a single node; these are the elements of the soccer network derived from the event and tracking data.

Actions are represented between two nodes as a link of the soccer network, which corresponds to each of the offensive and defensive actions as follows:

- Offensive actions:
  - Pass (OP-OP)
  - Dribble (OP-B)
  - Shot (OP-G)
- Defensive actions:
  - Tackle (DP-OP)
  - Interception (DP-B)
  - Out (DP-T)
  - Clearance (T-B)

Possession p is a link sequence $p = [a_1, a_2, ..., a_m]$, where m is the number of links (i.e., actions) in that possession.

The task of soccer network is to predict the future action given the current game state.

3.2 Temporal feature generation

We build temporal features on top of event and tracking data for each of the nodes and edges for each time-step of a performed action. We construct the following features for each node in the data: (x, y) locations, players and ball velocity, distances and angles of each node to goal, distance to the ball carrier, and a flag that indicates whether the player is the ball carrier. Our constructed edge features are: a flag defining the relationship between the two nodes (teammates 2, or opponents 1), the distance between the two interacting nodes.

3.3 Soccer network architecture

Our proposed soccer network serves the goal of predicting the offensive and defensive actions given a game situation. It consists of an encoder-decoder pair, where an encoder is a function that maps from a dynamic interaction of the nodes (i.e., link sequences) to node embeddings, and a decoder takes as input the node embeddings and performs link

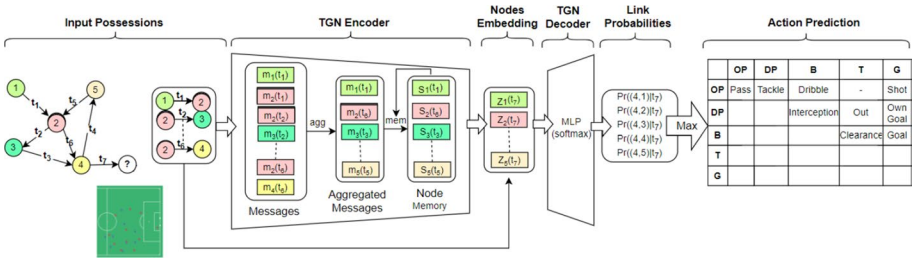


Fig. 2 Soccer network architecture

prediction of the future time-step. We then put an action label to the link according to the predicted source and destination nodes. For instance, if the source and destination nodes are both offensive players (denoted as OP-OP), the predicted action is labeled as a pass, and if the source node is an offensive player (OP) and the destination node is a defensive player (DP), the action is labeled as a tackle. Figure 2 illustrates the architecture of our network. Following the methodology of TGN Rossi et al. (2020), it consists of the following five modules:

1. Memory: Memory is the state of each node at time-step t, which is shown by vector $S_i(t)$ for each node i the model has seen so far. We update the memory $S_i(t)$ after an event $x(t)$ has occurred, which can be any of the node-wise ($v_i(t)$) (e.g., changing the location or velocity of the players on the field), or interaction-wise ($e_{ij}(t)$) (e.g., passing between two players) types. This module helps memorizing the long-term dependencies between the nodes of the graph (e.g., the actions happened in the past such as passing or tackle between two players, or a shot). When a new player is sent onto the field and a new node is created, the network initializes a zero vector for it, and then updates the memory after each event the player is involved in. This node addition or deletion can be applied even after training. Therefore, we use it in the real-time while deployment of the system.
2. Message function: For each event $x(t)$ involving node i, we compute a message to update its memory. If a node-wise event happens $v_i(t)$ (e.g., a node feature changes), the following single message will be computed for node i involved in the event at time-step t:

$$m_i(t) = msg(s_i(t-1), t, v_i(t))$$

and in case of an interaction $e_{ij}(t)$, we compute two messages for the source node i and destination node j at time t as follows:

$$m_i(t) = msg(s_i(t-1), s_j(t-1), \Delta t, e_{ij}(t))$$

where the msg functions are learnable message functions. In this work, we experiment with setting it as either a Multilayer Perceptron (MLP) or identity (id) that is the concatenation of the inputs.
3. Message aggregator: Since each node i might be involved in multiple events, including node-wise or interaction, we must aggregate all those messages until time t_b the node has been involved in. We show the aggregated message for node i as $\bar{m}_i$ and compute it as follows:

$$\bar{m}_i(t) = agg(m_i(t_1), ..., m_i(t_b))$$

where the *agg* is the aggregation function. In this work, we experiment with setting the *agg* as either to the *most recent message* (i.e., keep only most recent message for a given node) denoted as *(last)*, and to the *mean message* (i.e., average all messages for a given nod) denoted as *(mean)*.

4. Memory updater: A memory s_i of node i is updated upon each event involving the node:

$$s_i(t) = mem(\bar{m}_i(t), s_i(t-1))$$

where *mem* is a learnable memory update function. In this work, we experiment with setting *mem* as either an Long Short-Term Memory (LSTM) or a Gated Recurrent Unit (GRU).

5. Embedding: This module generates embedding $z_i(t)$ for node i at time t. It helps avoiding memory staleness problems: in TGN, the memory of a node is updated only if the node is involved in an event. This is problematic in our soccer task since it might happen that a player is inactive for a long time and he does not perform any actions, neither changes his location. In this case, his memory becomes stale and we need to aggregate his neighbours' memory and compute an up-to-date embedding for him. The embedding is calculated as follows:

$$z_i(t) = emb(i,t) = \sum_{j \in \eta} h(s_i(t), s_j(t), e_{ij}, v_i(t), v_j(t))$$

where η represent the neighbors of node i (we use all the rest of 21 players, ball, touchline, and goal nodes as the neighbours in this work) and h is a learnable function. In this work, we experiment with learning h as one the following cases:

- *Identity (id):* $emb(i,t) = s_i(t)$, which uses the memory directly as the node embedding.
- *Temporal Graph Attention (attn):* A series of L graph attention layers compute i's embedding by aggregating information from its $L-hop$ temporal neighborhood. The temporal graph attention is able to select the neighbours that are the most important according to their features and timing information.

4 Measuring team performance with reinforcement learning

This section describes our approach to define a Markov Decision Process (MDP) for soccer games and computing Q-values to evaluate actions of players under different game contexts. At first, we show how we model MDP from soccer data to evaluate the actual actions of the players, and then infer the optimal action which would lead to a higher chance of wining in the same situation. We then show how the difference between the values of the actual and optimal actions can be utilized for measuring team performance.

4.1 Learning actual Q-values

We aim to use a RL model to learn an action-value Q-function. Unlike most previous works on active RL, which aim to calculate optimal strategies for complex continuous-flow games, we solve a prediction (not control) problem with passive learning Barto and Sutton (1998). We use RL as a behavior analytics tool for the real players performing actions on the pitch, not by controlling artificial agents (Liu & Schulte 2018; Liu et al.

2020). Learning Q-values in this context matches the work by Nakahara et al. (2023) that proposes evaluating possible actions for on- and off-ball soccer players based on multi-agent deep RL. They analyzed the relationship with conventional indicators, season goals, and game ratings by experts. In our work, we aim to directly derive the optimal action maximizing the scoring chances at the end of possession instead of evaluating the possible actions on the field. Our approach of modelling soccer with an MDP requires a number of well-defined elements: a tuple of (S, A, R, π), where S represents the set of states, A represents the set of actions, R represents the reward function formulating the reward the agent receives for any given state/action pair, and π represents the policy that is interpreted as the probability that the player takes any given action based on the current state of the environment. Now we can define each of these components in our soccer possession environment:

- Agents: offensive and defensive players from both teams which are in position of performing any action.
- Action A: movements performed by players. Our model applies a discrete action vector using a one-hot representation. The offensive actions are: pass, dribble, and shot. The defensive actions are: tackle, interception, out, and clearance.
- State S: sequence of actions and their features (defined in previous sections including players and the ball locations, velocities, distance and angle to goal, etc.) in a possession of each team.
- Reward R: we assign hand-crafted reward functions according to the performed actions:

$$r(s,a) = \begin{cases} xG & \text{if A is a shot;} \\ 0 & \text{if A is pass or dribble;} \\ -0.1 & \text{if A is defensive action (possession loss),} \end{cases} \tag{1}$$

  Our proposed reward function computes the immediate reward by the arbitrary action that each player performed. Choosing the shot, the player receives the expected goal value (i.e., the probability of scoring a goal from a shot) which we denote as xG. If he performs any action other than a shot (e.g., dribble or pass), we assign zero as the reward. On the other side, if a defensive players stops the possession, we assign -0.1 as the reward. In this work, -0.1 is the parameter of our approach and has proved to be the best reward of possession loss to confirm the convergence of the policy network, and conform to the average xG in dataset. Moreover, the sum of $r(s, a)$ at each time-step throughout the whole episode is the indicator of the expected goal for the team. Thus, the control objective is to maximize the expected goal of the team. The most effective RL algorithms in this context are those which could learn optimal strategy and converge with a simplified and less handcrafted reward functions. Unlike the complex rewards proposed in the previous works by Rahimian et al. (2022, 2022), Rahimian and Toka (2023), Rahimian et al. (2023) to train the optimal policy using policy gradient algorithm, the applied optimization algorithm in this work could be optimally trained and produce results with the above simplified reward function.
- Episode τ: starts at the beginning of the game, or after a goal, and ends with a goal or at the end of the game.

In order to calculate the Q-values for each action, we follow the methodology of Liu and Schulte (2018), Liu et al. (2020) which estimates the probability that the ball possessor team scores a goal at the end of the current episode. To do so, we train our proposed soccer network described in the previous section with the Temporal Difference (TD) prediction method Sarsa. We aim to learn a function that estimates $Q(S_t, A_{actual})$ for any actual action observed in our dataset. The TD loss function for computing Q-values is as follows:

$$L(\theta) = \sum_{t \in T} \mathbb{E}[(r_{t+1} + Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t))^2]$$

We used mini-batch gradient descent with backpropagation to find weights of our soccer network that minimize this loss function above.

4.2 Optimal decision making with offline reinforcement learning

So far, our proposed soccer network is able to predict the future action given a game situation, and the Q-value of any action observed in the dataset. But the estimated actions are not comprehensive enough to assist the players and coaches in optimal decision making: the predicted actions are estimated according to the general policy of the historical games, and there is no evidence of optimality of the decisions and policies made by teams and players in prior matches. The current analytical methods Fernandez et al. (2019); Peralta Alguacil et al. (2020) propose estimating value surfaces by training the neural network to predict the probability of goal scoring within the next 10 actions or at the end of the actual possession. In this section, we elaborate on our proposed optimization algorithm that can directly estimate the optimal action that helps team winning on the long run rather than learning values for each of the discrete actions that occurred in prior games.

Considering the sequential decision making and the opportunity of modelling our soccer dataset as a Markov Decision Process (MDP), the most suitable optimization algorithm to infer the optimal action for each situation is the Q-learning algorithm. Q-learning is an off-policy algorithm (Barto & Sutton 1998), meaning that the target can be computed without consideration of how the experience was generated. In principle, off-policy RL algorithms are able to learn from data collected by any behavioral policy (Fujimoto et al. 2019). However, we would require an algorithm that would be able to extract policies with the maximum possible utility out of the available data, without interaction with the environment. These types of algorithms fall into the family of offline RL, which employ a dataset D collected by behavior policy (e.g., the output of action prediction performed by our soccer network in previous sections). The dataset is collected once, and is not altered during training, which makes it feasible to use large previous collected datasets. The training process does not interact with the MDP at all, and the policy is only deployed after being fully trained (Levine et al. 2020).

4.3 Conservative Q-learning

Offline RL algorithms typically suffer from overestimation of the values and distributional shift, as the policy learned and used to sample actions $a\prime$ differs from the policy used to generate the dataset. The actions generated by the learning policy may be out-of-distribution from what is present in the dataset, leading to potential overestimation of Q-values. While approaches like the evolutionary multi-objective reinforcement learning (EMORL) algorithm (Fuhong et al. 2023) and the deep contrastive

representation learning with self-distillation (DCRLS) method (Xiao et al. 2024) offer innovative solutions in their respective domains, they may not directly address the specific challenge of overestimation in offline RL. In contrast, our use of Conservative Q-Learning (CQL) for offline optimization in soccer analytics directly targets this issue. CQL is specifically designed to mitigate overestimation bias and distributional shift, making it well-suited for learning from historical soccer data where policy distributions may differ from the learning policy. Leveraging CQL, we can effectively optimize decision-making in soccer matches and improve the accuracy of win prediction in real-time.

In order to mitigate the overestimation problem, Kumar et al. introduced Conservative Q-Learning (CQL) (Kumar et al. 2020), an algorithm which first modifies the updates the rule of the Q-function to also minimize Q-values under current policy, while minimizing the Bellman error, and then maximizing values under the data distribution for the underestimation issue. The CQL algorithm can then be used to perform Q-Learning or train Actor-Critic policies, like SAC. In Conservative Q-Learning, we explore an alternative method to traditional approaches in RL, particularly in actor-critic frameworks, where constraints are typically imposed on the policy. Instead of restricting the policy directly, we focus on regularizing the value function or Q-function to address potential issues of overestimation for actions occurring outside the expected distribution. This method offers several advantages, including its applicability to various RL methods like actor-critic and Q-learning, even when a policy is not explicitly represented, and its ability to avoid the need for modeling the behavior policy.

To implement this approach, they introduce a conservative penalty term into the objective function. This penalty term, referred to as C_{CQL_0}, minimizes Q-values for actions chosen according to a specific distribution, thereby reducing the impact of potentially erroneous high Q-values for out-of-distribution actions. By incorporating this penalty, we ensure that the Q-values align more closely with the expected distribution of actions, without explicitly estimating uncertainty. Additionally, they consider a variant of this approach, denoted as C_{CQL_1}, which includes both a minimization term under the chosen distribution and a maximization term for state-action pairs in the batch. This modification aims to strike a balance between minimizing Q-values for out-of-distribution actions and maximizing values for actions within the expected distribution. While this variant does not guarantee a pointwise lower bound on the true Q-function, it offers appealing conservatism guarantees under the current policy while significantly reducing underestimation in practice. (See the work by Kumar et al. (2020) for more explanation on CQL algorithm.)

In this work, we use CQL to compute pessimistic Q-values of the optimal actions. We then calculate these Q-values for the optimal actions for each game state and denote it as $Q(S_t, A_{optimal})$. The loss function of descrete version of CQL is as follows:

$$L(\theta) = \alpha \mathbb{E}_{S_t}[log \sum_{A} expQ_\theta(S, A) - E_A[Q_\theta(S, A)]] + L_{DoubleDQN}(\theta)$$

where α is an automatically adjustable value via Lagrangian dual gradient descent.

We integrated our soccer network into the d3rlpy package d3rlpy (2019) and trained CQL algorithm accordingly. (See Appendix 8.2 for implementation details and hyperparameters of this algorithm).

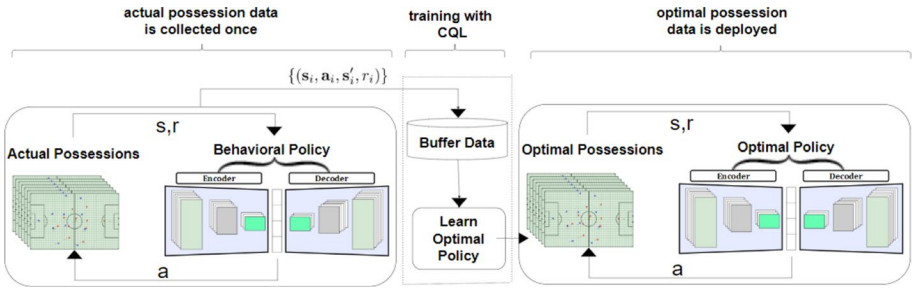


Fig. 3 Offline CQL workflow for producing optimal actions inside a possession from actual possession data. The state, action, and reward are shown with s, a, and r respectively. The undelying network is the encoder and decoder components of TGN that is trained with actual possession data to exploit behavioral policy, and its parameters are tuned with offline CQL algorithm to derive optimal policy. The middle training part is the offline CQL which employs a dataset collected by the behavioral policy (i.e., predicted actions from soccer network). The dataset is collected once, and is not altered during training, which makes it feasible to use large, previously collected datasets. The training process does not interact with the MDP at all, and the policy is only deployed after being fully trained

4.4 Measuring teams' effectiveness

Measuring teams' effectiveness is one of the most prominent use-cases of our approach. Since we have both actual actions observed in the dataset, and the inferred optimal action in the same game state, we are able to measure how close teams and players are to optimal policy. To do so, we compute the difference between our estimated $Q(S_t, A_{actual})$ for the actual action and $Q(S_t, A_{optimal})$ for the optimal one at the same game state S_t and sum it over all actions performed in the dataset. We call this metric as Optimal Action Divergence and calculate it as follows:

$$\text{Optimal Action Divergence}(S_t) = Q(S_t, A_{optimal}) - Q(S_t, A_{actual}); \tag{2}$$

which shows the closeness of the actual action to the optimal one for each game state S_t, and we compute teams' effectiveness as:

$$\text{Teams' Effectiveness} = 1 - \frac{1}{T}\sum_{t=1}^{T} \text{Optimal Action Divergence}(S_t) \tag{3}$$

for a game with T events. As a result, the larger effectiveness of a team shows their better performance in selecting optimal actions in any of the offensive and defensive situations. The workflow of inferring optimal policy with offline CQL and measuring teams' effectiveness is illustrated in Fig. 3.

5 Match outcome prediction

In this section, we describe our proposed approach for constructing an in-game win probability model for soccer.

5.1 Modelling match outcome and win probabilities

The task of an in-game win probability model is to predict the probability distribution over the possible match outcomes: $Pr(Y|S_t)$ given the game state S at time t, in which the outcome Y can be any of the win, draw, loss for the team. In this section, we define the game state as the previous five actions a of the current time-step t and its corresponding features x and denote them as $S_t = [(a_{t-4}, x_{t-4}), ..., (a_{t-4}, x_{t-4}), (a_t, x_t)]$. Setting the game state with a fixed number of previous actions offers several advantages. First, most machine learning algorithms require fixed number of features. Second, considering a small window focuses attention on the most relevant aspects of the current context. The number five of previous actions is a parameter of the approach, and was empirically found to work well.

We formulate the problem as a multi-class classification to predict the probability of each outcome given a game state. Our task can then be defined as:

Given: game state $S_t = [(a_{t-4}, x_{t-4}), ..., (a_{t-4}, x_{t-4}), (a_t, x_t)]$;

Estimate: the probability of win, draw, loss outcomes for the ball possessor team.

To compute the probabilities, we experiment with the following classification algorithms: CatBoost, Logistic Regression, Random Forest.

5.2 Game state features

We utilize the following categories of features for describing the game state and predict outcomes:

1. Base features:
   - Time remaining from action occurrence until the end of match half;
   - Goal difference between the goals the teams have scored so far.
2. Teams' effectiveness feature:
   - Optimal Action Divergence(S_t): the divergence of the actual action from the optimal one for each game state S_t performed by the players. This feature gives an indication of teams' effectiveness in scoring in the rest of the game. The larger teams' effectiveness and smaller Optimal Action Divergence outline higher chance of winning.
3. Contextual features:
   - Team goals: The number of goals each team scored so far. This feature gives an indication of the teams' strength in goal scoring in the past and the likelihood of scoring again).
   - Red cards: The number of red cards each team has received so far. A double yellow card is counted as a red card.
   - Yellow card: The number of yellow cards each team has received so far. A second yellow card is not counted, as it is included in the red cards. We assume that a weaker team that is defending is more likely to commit a foul and receive more yellow cards.
   - Duel winning: the percentage of duels won in the previous five time-steps.
   - Penalty calls: The number of penalty calls and the percentage resulted in a goal.
   - Amount of injury time and stoppage time.

- xG: Number of shots including blocked shots and situations where a player has been in a good position to score, divided by total number of opportunities.

5.3 Evaluation method

An in-game win probability predictor model should be able to provide calibrated probability estimates that reflect what is the most likely to happen in reality. Since we formulate the problem as a multi-class classification task with three possible outcomes, we evaluate the accuracy of prediction with Ranked Probability Score (RPS) (Constantinou & Fenton 2012) at match time t to quantify how close the predictions are to the actual outcome. We compute the RPS metric as follows:

$$RPS_t = \frac{1}{2}\sum_{i=1}^{2}(\sum_{j=1}^{i} p_{t,j} - \sum_{j=1}^{i} e_j)^2 \tag{4}$$

where vector $p_t = [Pr(Y = win|S_t), Pr(Y = draw|S_t), Pr(Y = loss|S_t)]$ is the estimated probabilities at a time t. e_j denotes the final outcome of the game, in which the win ($j = 1$), draw ($j = 2$), and loss ($j = 3$) are denoted as [1,1,1], [0,1,1], and [0,0,1], respectively.

To simplify the understanding of RPS for coaches and players, it can be likened to a measure of how well the predictions match with what actually happens. To calculate this metric, we assign specific values to the outcomes of win, draw, and loss in order to calculate the RPS metric accurately. The notation [1,1,1] represents the outcome of a win, indicating that the predicted probability for winning the game is fully assigned to the correct outcome. Similarly, [0,1,1] signifies a draw, where the predicted probability is divided equally between the draw and loss outcomes. Finally, [0,0,1] denotes a loss, indicating that the predicted probability is fully assigned to the loss outcome. This notation scheme ensures that the RPS metric appropriately penalizes deviations between predicted probabilities and actual outcomes, thereby providing a comprehensive evaluation of the model's performance in predicting in-game win probabilities. A lower RPS score implies more accurate predictions, while a higher score suggests that the predicted probabilities deviate from the observed outcomes. In essence, RPS provides a quantifiable way to assess the reliability of probabilistic forecasts, helping coaches and players gauge the trustworthiness of predictive information in their decision-making processes. See the paper by Costa Constantinou and Fenton (2012) for detailed explanation of this metric.

6 Experiment

In this section, we showcase the practical application of our deployed system by first describing the dataset, then by providing an evaluation of our soccer network and match outcome prediction, finally describing the teams' effectiveness in terms of selecting the optimal action. We trained the soccer network for action prediction purpose using the code provided in our GitHub repository.[2] Then we integrate the network to the d3rlpy offline RL package[3] to train CQL algorithm. The rest of the machine learning algorithms has been

[2] https://github.com/hsnlab/sports_analitica.

[3] https://d3rlpy.readthedocs.io/en/v1.1.0/index.html.

trained using scikit-learn Python package.[4] All models are trained on a server enriched with GV100GL [Tesla V100 PCIe 16GB] GPU.

6.1 Dataset

The dataset we used for deployment of our proposed system consists of high-resolution spatiotemporal tracking and event data covering all 330 games of the 2020-21 season, and 100 games of 2021-22 season of the Belgian Pro League collected by Stats Perform.[5] The tracking data includes the (x,y) coordinates of all 22 players and the ball on the pitch at 25 observations per second. The event data includes on-ball action types such as passes, shots, dribbles, etc., annotated with additional features such as contestants, game period ID, ball possessor player ID, start and end locations of the ball. We merge tracking with event data; each record of our merged dataset includes all players and the ball coordinates with their corresponding features for each time-step of an action performed by offensive or defensive players.

6.2 Teams' effectiveness

In order to provide an interpretable summary of the results for the soccer players and coaches, our framework could be adjusted to analyze the team-specific propensities to perform any of the action types within any of the phases of ball possessions. Moreover, we compute Teams' Effectiveness metric (formula of (3)) for each team participating in the 2020/21 season of Belgian Pro League. Table 1 presents the calculated effectiveness of the teams. The teams are sorted according to the league table at the end of season 2020-21. For instance, the mean effectiveness over all possessions of Club Brugge in the 2021-22 season is calculated as 0.45. Another observation from Table 1 is that the teams at the top of the table have larger effectiveness in comparison to the teams at the bottom of the table. That is because a team like Club Brugge at top of the table quite often selects the optimal actions (i.e., their behavioral policy is nearly the same as their optimal policy), whereas the behavioral policy of the teams at the bottom of the table is far from their respective optimal policy.

6.3 Evaluation results

6.3.1 Soccer network evaluation results

We aim to evaluate the prediction performance of the undelying network for the prediction phase, and use it consequently for the optimization task. To do so, we inspect the performance of the teams' offensive and defensive behavior prediction (i.e., the probability of interrupting a possession by each of the offensive and defensive actions). In order to handle the spatiotemporal nature of our dataset, we needed a sophisticated model and rich feature set, which could optimize the prediction performance. Thus, model selection was the core task of this study. We first created appropriate state dimensions suitable for

[4] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[5] http://www.statsperform.com/

Table 1 The effect of approaching the optimal policy on the team-level

Teams	Teams' effectiveness
Club Brugge	0.45
Antwerp FC	0.43
Anderlecht	0.40
Genk	0.42
Oostende	0.38
Standard	0.40
AA Gent	0.35
Mechelen	030
Beerschot	0.22
Waregem	0.28
OH Leuven	0.30
Eupen	0.21
Charelori	0.18
Kortrijk	0.15
Sint-Truiden	0.12
Cercle Brugge	0.14
W-Beveren	0.11
Excel Mouscron	0.09

Teams are sorted according to their final ranking in the 2020/21 season of Belgian Pro League

Table 2 Prediction performance of different design choices for spatiotemporal analysis

Spatio temporal model	Accuracy (%)	Loss	Inference time (s)	Parameters
3D-CNN	73	0.63	0.31	61,211
LSTM	71	0.63	0.11	50,804
Autoencoder-LSTM	79	0.59	5.02	92,022
CNN-LSTM	81	0.56	0.51	56,036
TGN	87	0.52	0.63	72,300

Inference times are the average running times over 20 iterations of training

each model by reshaping the state inputs, then fed our reshaped arrays to the following networks: 3D-CNN, LSTM, Autoencoder-LSTM, CNN-LSTM, and TGN, to compare their classification performance (c.f. Table 2). We chronologically split the possessions into 80% of train, 10% of validation for hyperparameter tuning and model selection, and the remaining 10% as hold-out data for testing. Chronological splitting in soccer data analytics offers several advantages that contribute to the realism and robustness of predictive models. By organizing the dataset based on the temporal order of events, this approach mimics the chronological flow of soccer matches, aligning with real-world scenarios where data is collected over time. This temporal realism enables models to adapt to evolving patterns, handle changes in player performance and strategies, and capture seasonal trends inherent

in soccer seasons. The technique helps prevent data leakage, as future information is not used during model training, ensuring the model's evaluation integrity. Additionally, models developed through chronological splitting showcase their generalization capabilities to future scenarios, making them more reliable for real-time decision-making during matches. Overall, this approach fosters the development of robust models that are less sensitive to short-term fluctuations, providing a comprehensive and realistic assessment of the model's performance in dynamic soccer analytics environments. Random splitting, where the dataset is randomly divided into training, validation, and test sets, has its limitations in the context of soccer data analytics. One significant problem is that it may not adequately account for the temporal nature of the data. Soccer events, strategies, and player performances can evolve over time, and random splitting does not ensure a chronological representation of these changes. This can result in models that may not generalize well to future scenarios or may be overly sensitive to short-term fluctuations. In soccer, where seasonality, player transfers, and changes in team dynamics are common, models developed without considering the temporal order of events may lack the adaptability needed to make accurate predictions. Random splitting also poses a risk of data leakage, as information from the validation or test sets could unintentionally influence the model during training, leading to an overestimation of its performance. Categorical cross entropy loss on train and validation datasets is used to evaluate the model. All layers of the networks are carefully calibrated. Dropout is used to reduce interdependent learning among units, and early stopping is used to avoid overfitting. As Table 2 suggests, TGN outperforms other models in terms of accuracy and loss in the test set. In Table 2, the number of parameters depends on the size of game state and architecture of the respective neural network (e.g., number of layers and neurons of each layer). This table also presents loss to evaluate the goodness of the fit for each model trained on different type of states. Loss is a measure of how well the model is performing with respect to its objective. It quantifies the difference between the predicted values and the actual values (or labels) in the training data. The goal during training is to minimize this loss. Common loss functions include mean squared error for regression tasks and cross-entropy loss for classification tasks. Accuracy, on the other hand, is a measure of how many predictions the model gets correct compared to the total number of predictions. It is a percentage value indicating the proportion of correctly classified samples. While accuracy is an essential metric, it does not take into account the confidence or certainty of predictions. The theoretical connection between accuracy and loss lies in the fact that reducing the loss often leads to an improvement in accuracy. During training, the optimization algorithm adjusts the model's parameters to minimize the loss function. As the loss decreases, the model tends to make better predictions, which, in turn, improves accuracy. However, it's crucial to note that accuracy alone may not provide a complete picture of the model's performance, especially in situations with imbalanced classes or when the uncertainty of predictions is a significant concern. In summary, the theoretical connection between accuracy and loss is rooted in the shared objective of improving the model's predictive capabilities. Reducing the loss generally corresponds to an increase in accuracy, but both metrics are used in tandem to comprehensively assess and fine-tune the performance of a neural network during training. Further information about hyperparameters and implementation details of TGN are provided in Appendix 8.1.

Table 3 Multiple variants of TGN used in our ablation studies. msg, agg, mem, and emb stand for message function, aggregation function, memory updater, and embedding, respectively

	msg	agg	mem	emb
TGN-id	id	last	GRU	id
TGN-mean	id	mean	GRU	attn
TGN-last	id	last	GRU	attn
TGN-att	id	last	LSTM	attn

Table 4 Area under curve (AUC) and average precision (AP) for future edge prediction tasks in transductive and inductive settings of different versions of TGN

	Transductive		Inductive	
	AUC	AP	AUC	AP
TGN-id	0.76	0.73	0.63	0.64
TGN-mean	0.78	0.74	0.66	0.68
TGN-last	0.80	0.78	0.68	0.68
TGN-att	0.88	0.85	0.80	0.79

All the results are averaged over 10 runs

6.3.2 Ablation study

Our soccer network consists of TGN which has learned the future edge probabilities in a self-supervised manner (i.e., corresponding methods, for processing unlabelled data to obtain useful representations for downstream learning tasks). We evaluate its performance through both transductive and inductive settings. In the transductive setting, we predict future links of the nodes observed during training, whereas in the inductive setting we predict future links of nodes never observed before. For all tasks, we used a split of 80%-10%-10% for the train-validation-test sets respecting the chronological order of the games to make sure actions from the same matches do not end up in both train and test sets and to avoid temporal information loss. Furthermore, we perform an ablation study to design the best combination of different components of TGN modules as described in Table 3. The final results of Average Precision and AUC for different versions of TGN are illustrated in Table 4. According to this table, TGN-att outperforms the counterparts (by achieving 88% of AUC in the transductive setting) and we use its prediction results for the rest of our experiments for our deployed system.

We conducted an ablation study to investigate the performance of different variants of the TGN utilized in our framework. Table 3 outlines the various configurations, each denoted by TGN followed by a specific identifier indicating the message function (msg), aggregation function (agg), memory updater (mem), and embedding (emb). We evaluated these variants based on their ability to predict future edge probabilities in both transductive and inductive settings. The results, presented in Table 4, highlight the superior performance of TGN-att, which employs attention mechanisms for memory updating, achieving the highest Area Under Curve (AUC) and Average Precision (AP) scores across both settings.

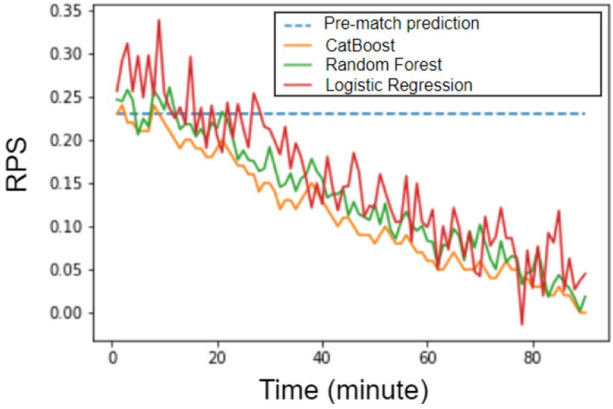


Fig. 4 Comparison of RPS computed for every minute of the game trained with three different classification algorithms, and the pre-match outcome prediction

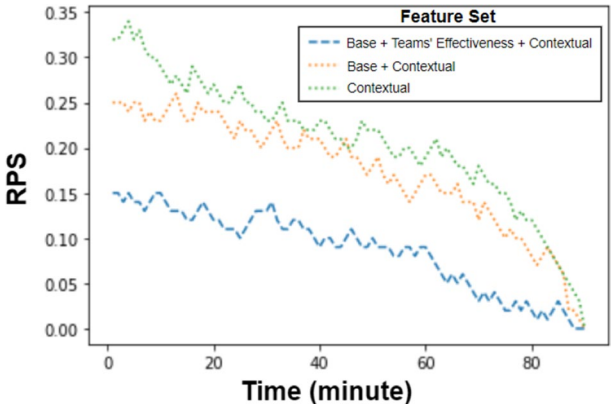


Fig. 5 RPS of outcome prediction for every minute of the game trained with three feature sets. Adding the teams' effectiveness feature (blue dashed line) significantly improves the prediction

6.3.3 Match outcome evaluation results

Since our proposed system is designed for real-time match result prediction, we need to calculate RPS for each time-step of the game which has been scaled to 90 min. The RPS of all in-game outcome prediction models improves when the game progresses (Figs. 4, 5), as they gain more information about the final outcome. In Fig. 4, we compare RPS computed for every minute of the game, in which the outcome has been predicted with three classification algorithms. According to the figure, the CatBoost result (orange line) has the lowest RPS in most of the times with an average of 0.11 throughout the game. Furthermore, we study the effect of adding different category of features in Fig. 5. We observe that the combination of three categories of features (the blue dashed line) is outperforming the other feature sets with an average of 0.08 RPS throughout the game. Furthermore, adding the Teams' Effectiveness feature significantly improves the prediction accuracy and proves that our optimization algorithm has learned optimal decisions pretty well and accurately computed the Action Effectiveness metric.

Next, we present the results of our evaluation, comparing the performance of our prediction framework with that of four baseline models in Table 5. To assess the quality of our predictions, we again employ the Rank RPS metric, where lower RPS values indicate

Table 5 Comparison of RPS scores from different in-game outcome prediction models

Model	Average RPS
Our prediction model	0.10
Bayesian approach Robberchts et al. (2021)	0.25
Betting portal (Unibet)	0.39
Betting portal (bwin)	0.43
Betting portal (Curebet)	0.53

better prediction results. Our prediction model, which directly emphasizes optimal decisions and leverages them for in-game result prediction, outperforms the following existing models, as expected. First, we employed a Bayesian approach Robberchts et al. (2021) to predict in-game outcomes using our Belgian Pro League dataset, resulting in an average RPS of 0.25. Additionally, we utilized a betting portal Odds portal (xxxx) to scrape historical odds of Belgian matches, converted them into win-draw-loss probabilities, and calculated the average RPS using odds from three prominent bookmakers: bwin[6] (RPS = 0.43), Unibet[7] (RPS = 0.39), and Curebet[8] (RPS = 0.53). In contrast, our approach yielded an impressive average RPS of 0.1 across all matches, establishing its superiority over existing reproducible baselines.

6.3.4 Computational complexity analysis

The computational complexity of our proposed framework encompasses various aspects, including model training, inference, and real-time prediction. The primary computational overhead lies in training the deep learning models, particularly the TGN and the RL algorithm. The training complexity of these models depends on factors such as the size of the dataset, the number of parameters in the model architecture, and the complexity of the optimization process. For TGN, the computational complexity mainly arises from processing the high-resolution spatiotemporal tracking and event data. The complexity of TGN training scales with the number of nodes (players and ball) and the length of the sequences (time steps) in the input data. Additionally, the use of attention mechanisms and recurrent neural networks (RNNs) in TGN may contribute to increased computational demands during training. Incorporating the Conservative Q-Learning algorithm into our framework adds another layer of complexity, particularly during the offline RL process. While CQL offers advantages such as improved sample efficiency and robustness, it may require extensive computational resources for training, especially when dealing with large-scale datasets and complex action spaces. During inference and real-time prediction, the computational complexity is primarily determined by the efficiency of the deployed models and algorithms. Our framework leverages optimized inference techniques and parallel processing to minimize latency and maximize throughput, enabling efficient real-time analysis and prediction of match outcomes and player actions. Generally, while our framework introduces computational challenges, such as training deep learning models and offline RL algorithms, we employ optimization strategies and parallel computing techniques to

[6] https://casino.bwin.com/en/games.

[7] https://www.unibet.com/

[8] https://www.curebet2023.com/en.

mitigate these challenges and ensure efficient performance in real-world applications. Further analysis and optimization of computational resources will be conducted to enhance the scalability and efficiency of our framework for soccer analytics.

7 Conclusion

In this paper, we have presented an innovative approach for in-game outcome prediction in soccer using a combination of TGNs and offline RL. Our methodology leverages the rich spatiotemporal tracking data available in soccer matches to predict offensive and defensive actions and optimize decision-making processes. Through the integration of TGNs, we effectively model the complex interactions between players and the ball over time, enabling accurate action prediction. Additionally, our application of offline RL allows us to extract optimal strategies solely from historical data, providing valuable insights into teams' performance and enhancing our ability to predict in-game outcomes. Our experimental results demonstrate the effectiveness of our approach, with high accuracy achieved in both action prediction and in-game outcome prediction tasks. Specifically, we achieve an accuracy of 87% in predicting offensive and defensive actions and an error rate of 0.1 in in-game outcome prediction, outperforming counterpart models and bookmakers' odds. These results highlight the potential of our methodology to significantly impact various stakeholders in the soccer ecosystem, including teams, leagues, betting industries, media, and fans.

Moving forward, we believe that our approach has the potential for further refinement and application in real-world scenarios. Future research directions could include exploring additional features and data sources, refining the modeling architecture, and incorporating dynamic adjustments based on real-time game developments. By continuing to innovate in the field of soccer analytics, we aim to provide valuable insights and enhance decision-making processes for all stakeholders involved.

Hyperparameters and implementation details

Temporal graph network

- *Model architecture:* We employed the TGN architecture for learning the temporal dynamics of soccer events. The TGN consists of message passing and aggregation mechanisms to capture the temporal dependencies between actions in the soccer game.
- *Hyperparameters: Message function (msg):* We experimented with different message functions including identity (id), mean, and last, denoted as TGN-id, TGN-mean, and TGN-last respectively. *Aggregation function (agg):* The aggregation function aggregates the messages received by each node. We utilized aggregation functions such as last and mean. *Memory updater (mem):* We employed GRU and LSTM units for updating the memory state in the TGN modules. *Embedding function (emb):* We utilized attention-based embeddings (attn) for learning node representations.

- *Implementation details:* The TGN architecture was implemented using the PyTorch library. We trained the TGN models using Adam optimizer with a learning rate of 0.001. We utilized a batch size of 64 and trained the models for 100 epochs.

Conservative Q-Learning

- *Algorithm:* We employed Conservative Q-Learning (CQL) for optimizing player decisions in soccer. CQL aims to learn an optimal policy by regularizing the Q-function to avoid overestimation of out-of-distribution actions.
- *Hyperparameters: Penalty term* (α): We experimented with different penalty weights to balance between the conservative penalty term and the standard Bellman error term. *Behavior policy* (μ): We employed a regularized adversarial objective for computing the behavior policy, aiming to minimize the Q-values for out-of-distribution actions.
- *Implementation details:* The CQL algorithm was implemented using the d3rlpy offline RL package. We trained the CQL models using Adam optimizer with a learning rate of 0.0001. We utilized a batch size of 128 and trained the models for 5000 iterations.

Acknowledgements Project no. 2021−1.2.4-TÉT-2021-00053 has been implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the 2021−1.2.4-TÉT funding scheme.

Author contributions all authors have contributed.

Funding Open access funding provided by Budapest University of Technology and Economics.

Declarations

Conflict of interest Not applicable.

Ethical approval Not applicable.

References

Anzer, G., Bauer, P., Brefeld, U., Fassmeyer, D. (2022). Detection of tactical patterns using semi-supervised graph neural networks. In *16th MIT sloan sports analytics conference*.

Barto, A., & Sutton, R. (1998). *Reinforcement learning: An introduction*. MIT press Cambridge.

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V. F., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gülçehre, Ç., Song, H. F., Ballard, A. J., Gilmer, J., Dahl, G. E., Vaswani, A., Allen, K. R., Nash, C., Langston, V., ... Pascanu, R. (2018). Relational inductive biases, deep learning, and graph networks. CoRR abs/1806.01261, http://arxiv.org/abs/1806.01261

Beuoy, M. (2015). Updated NBA win probability calculator. https://inpredictable.com/2015/02/updated-nba-win-probabilitycalculator.html

Brandt, M., Brefeld, U. (2015). Graph-based approaches for analyzing team interaction on the example of soccer. In *8th workshop on machine learning and data mining for sports analytics*. https://api.semanticscholar.org/CorpusID:9760981

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2018). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*.

Constantinou, A. C., & Fenton, N. E. (2012). Solving the problem of inadequate scoring rules for assessing probabilistic football forecast models. *Journal of Quantitative Analysis in Sports*, *8*(1). https://doi.org/10.1515/1559-0410.1418

Crowder, M., Mark Dixon, C., Dixon, M., Ledford, A., & Robinson, M. (2002). Dynamic modelling and prediction of english football league matches for betting. *Journal of the Royal Statistical Society. Series D (The Statistician), 51*(2), 157–168.

d3rlpy: Discrete version of conservative q-learning algorithm. (2019), https://d3rlpy.readthedocs.io/en/v1.1.0/references/generated/d3rlpy.algos.DiscreteCQL.html#d3rlpy.algos.DiscreteCQL.

Dick, U., & Brefeld, U. (2019). Learning to rate player positioning in soccer. *Big Data, 7*(1), 71–82.

Dick, U., & Brefeld, U. (2023). Action rate models for predicting actions in soccer. *AStA Advances in Statistical Analysis, 107*, 29–49.

Dimitris, K. (2003). Analysis of sports data by using bivariate poisson models. *The Statistician, 52*, 381–393.

Dixon, M., & Coles, S. (1997). Modelling association football scores and inefficiencies in the football betting market. *Applied Statistics, 46*(2), 265–280.

Fernandez, J., & Born, L. (2020). Soccermap: A deep learning architecture for visually-interpretable analysis in soccer. In *The European conference on machine learning and principles and practice of knowledge discovery in databases (ECML PKDD)*.

Fernandez, J., Bornn, L., & Cervone, D. (2019). Decomposing the immeasurable sport: A deep learning expected possession value framework for soccer. In *13th MIT sloan sports analytics conference*.

Forrest, D., & Simmons, R. (2000). Making up the results: The work of the football pools panel, 1963–1997. *Journal of the Royal Statistical Society: Series D (The Statistician), 49*(2), 253–260.

Fuhong, S., Huanlai, X., Xinhan, W., Shouxi, L., Penglin, D., Zhiwen, X., & Bowen, Z. (2023). Evolutionary multi-objective reinforcement learning based trajectory control and task offloading in uav-assisted mobile edge computing. *IEEE Transactions on Mobile Computing, 22*(1), 7387–7405.

Fujimoto, S., Meger, D., & Precup, D. (2019). Off-policy deep reinforcement learning without exploration. In *36th international conference on machine learning, PMLR*, Vol. 97 (pp. 2052–2062).

Ganguly, S., & Frank, N. (2018). The problem with win probability. In *12th MIT sloan sports analytics conference*.

Goddard, J. (2005). Regression models for forecasting goals and match results in association football. *International Journal of Forecasting, 421*, 331–340.

Goddard, J., & Asimakopoulos, I. (2004). Forecasting football results and the efficiency of fixed-odds betting. *Journal of Forecasting, 23*, 51–66.

Havard, R., & Salvesen, O. (1997). Predicting and retrospective analysis of soccer matches in a league. *The Statistician, 49*, 399–418.

Hopkins, O. (2019). Opta's live win probability model on amazon prime video. https://www.statsperform.com/resource/optas-live-win-probabilitymodel-on-amazon-prime-video/

Kipf, T., Fetaya, E., Wang, K., Welling, M., & Zemel, R. (2018). Neural relational inference for interacting systems. In *International conference on machine learning*.

Koning, R. (2001). Balance in competition in Dutch soccer. *Journal of the Royal Statistical Society: Series D (The Statistician), 49*, 419–431.

Koopman, S. J., & Lit, R. (2019). Forecasting football match results in national league competitions using score-driven time series models. *International Journal of Forecasting, 35*, 797–809.

Kumar, A., Zhou, A., Tucker, G., & Levine, S. (2020). Conservative q-learning for offline reinforcement learning. In *Advances in neural information processing systems*. Curran Associates, Inc.

Levine, S., Kumar, A., Tucker, G., & Fu, J. (2020). Offline reinforcement learning: Tutorial, review, and perspectives on open problems. ArXiv, abs/2005.01643

Lindsey, G. R. (1961). The progress of the score during a baseball game. *Journal of the American Statistical Association, 56*, 703–728.

Liu, G., & Schulte, O. (2018). Deep reinforcement learning in ice hockey for context-aware player evaluation. In *Proceedings of the twenty-seventh international joint conference on artificial intelligence (IJCAI-18)*.

Liu, G., Luo, Y., Schulte, O., & Kharrat, T. (2020). Deep soccer analytics: Learning an action-value function for evaluating soccer players. *Data Mining and Knowledge Discovery, 34*(2), 1531.

Luo, Y., Schulte, O., & Poupart, P. (2020) Inverse reinforcement learning for team sports: Valuing actions and players. In C. Bessiere (Ed.), *Proceedings of the twenty-ninth international joint conference on artificial intelligence, IJCAI-20* (pp. 3356–3363).

Maher, M. (1982). Modelling association football scores. *Statistica Neer- landica, 36*(3), 109–118.

Mehrasa, N., Zhong, Y., Tung, F., Bornn, L., & Mori, G. (2018). Deep learning of player trajectory representations for team activity analysis. In *12th MIT sloan sports analytics conference*.

Muelling, K., Boularias, A., Mohler, B., Schoelkopf, B., & Peters, J. (2013). Inverse reinforcement learning for strategy extraction. In *2013 workshop on machine learning and data mining for sports analytics (MLSA 2013)*.

Nakahara, H., Tsutsui, K., Takeda, K., & Fujii, K. (2023). Action valuation of on- and off-ball soccer players based on multi-agent deep reinforcement learning. *IEEE Access, 11*, 131237–131244. https://doi.org/10.1109/ACCESS.2023.3336425

Odds portal. https://www.oddsportal.com/football/belgium/jupiler-pro-league-2021-2022/results/

Owen, A. (2011). Dynamic bayesian forecasting models of football match outcomes with estimation of the evolution variance parameter. *IMA Journal of Management Mathematics, 2*, 99–113.

Peralta Alguacil, F., Fernandez, J., Piñones Arce, P., & Sumpter, D. (2020). Seeing in to the future: using self-propelled particle models to aid player decision-making in soccer. In *In Proceedings of the 14th MIT sloan sports analytics conference*.

Rahimian, P., da Silva Guerra Gomes, D.G., Berkovics, F., & Toka, L. (2022). Let's penetrate the defense: A machine learning model for prediction and valuation of penetrative passes. In *Machine learning and data mining for sports analytics*.

Rahimian, P., Kim, H., Schmid, M., & Toka, L. (2024). Pass receiver and outcome prediction in soccer using temporal graph networks. In *2023 workshop on machine learning and data mining for sports analytics (MLSA 2023)*.

Rahimian, P., Oroojlooy, A., & Toka, L. (2021) Towards optimized actions in critical situations of soccer games with deep reinforcement learning. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*.

Rahimian, P., & Toka, L. (2021). Inferring the strategy of offensive and defensive play in soccer with inverse reinforcement learning. In *2021 workshop*.

Rahimian, P., Van Haaren, J., Abzhanova, T., & Toka, L. (2022) Beyond action valuation: A deep reinforcement learning framework for optimizing player decisions in soccer. In *16th MIT sloan sports analytics conference*.

Rahimian, P., Van Haaren, J., & Toka, L. (2023). Towards maximizing expected possession outcome in soccer. *International Journal of Sports Science and Coaching, 19*, 230.

Rahimian, P., & Toka, L. (2023). A data-driven approach to assist offensive and defensive players in optimal decision making. *International Journal of Sports Science and Coaching*. https://doi.org/10.1177/17479541221149481

Robberechts, P., Van Haaren, J., & Davis, J. (2021). A bayesian approach to in-game win probability in soccer. In *ACM SIGKDD international conference on knowledge discovery and data mining*.

Rossi, E., Chamberlain, B., Frasca, F., Eynard, D., Monti, F., & Bronstein, M. (2020). Temporal graph networks for deep learning on dynamic graphs. In *International conference on machine learning (ICML) workshop on graph representation learning*.

Routley, K., & Schulte, O. (2015). A markov game model for valuing player actions in ice hockey. In *Proceedings of the thirty-first conference on uncertainty in artificial intelligence*.

Sanchez-Gonzalez, A., Heess, N., Springenberg, J. T., Merel, J., Riedmiller, M., Hadsell, R., & Battaglia, P. (2018). Graph networks as learnable physics engines for inference and control. In *International conference on machine learning (ICML)*.

Simpson, I., Beal, R., Locke, D., & Norman, T. (2022). Seq2event: Learning the language of soccer using transformer-based match event prediction. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*. https://api.semanticscholar.org/CorpusID:251518201

Stöckl, M., Seidl, T., Marley, D., & Power, P. (2021). Making offensive play predictable using a graph convolutional network to understand defensive performance in soccer. In *MIT sloan sports analytics conference*.

Teranishi, M., Fujii, K., & Takeda, K. (2020). Trajectory prediction with imitation learning reflecting defensive evaluation in team sports. In *2020 IEEE 9th global conference on consumer electronics (GCCE)*.

Teranishi, M., Tsutsui, K., Takeda, K., & Fujii, K. (2022). Evaluation of creating scoring opportunities for teammates in soccer via trajectory prediction.

Tugbay, I. (2020). Using poisson model for goal prediction in European football. *Journal of Human Sport and Exercise*, *16*(4). https://doi.org/10.14198/jhse2021.164.16

Wang, Z., Velickovic, P., Hennes, D., Tomaev, N., Prince, L., Kaisers, M., Bachrach, Y., Lie, R., Li, W. K., Piccinini, F., Spearman, W., Graham, I., Connor, J. T., Yang, Y., Recasens, A., Khan, M., Beauguerlange, N., Sprechmann, P., Moreno, P., … Tuyls, K. (2023). Tacticai: An ai assistant for football tactics. *Nature Communications, 15*, 1906.

Xenopoulos, P., & Silva, C. (2021). Graph neural networks to predict sports outcomes. In *2021 IEEE international conference on big data (big data)* (pp. 1757–1763). https://doi.org/10.1109/BigData52589.2021.9671833

Xiao, Z., Xing, H., Zhao, B., Qu, R., Luo, S., Dai, P., Li, K., & Zhu, Z. (2024). Deep contrastive representation learning with self-distillation. *IEEE Transactions on Emerging Topics in Computational Intelligence, 8*(1), 3–15.

Yang, Y., Qin, T., & Lei, Y. H. (2020). Real-time esports match result prediction.

Yedid, H. (2017). Vain: Attentional multi-agent predictive modeling. *Advances in Neural Information Processing Systems, 30*, 2701–2711.

Yeh, R., Schwing, A., Huang, J., & Murphy, K. (2019). Diverse generation for multi-agent sports games. In *IEEE conference on computer vision and pattern recognition*.

Yeung, C. C. K., Sit, T., & Fujii, K. (2023). Transformer-based neural marked spatio temporal point process model for football match events analysis.