Introducing a new soccer ghosting Algorithm

Hugo Georgenthum

February 29, 2024

Introduction

In contemporary soccer analysis, understanding player movement patterns and positional effectiveness is crucial for accurate player valuation and team performance optimization. A key concept in this realm is "ghosting" which defines the predictive modeling of future player positions based on historical data and contextual information. By leveraging mathematical techniques and spatio-temporal data analysis, ghosting offers valuable insights into player behavior, aiding in recruitment decisions, tactical adjustments, and overall team strategy.

In this paper, we explore a new algorithm of ghosting in soccer analytics, by using different techniques of deep learning. Whereas time dependency is a commonly used technique for ghosting algorithm as in paper [2], we present here a new approach in particular concerning the game representation. Indeed, the challenge of ghosting for soccer game lies in the complexity in the game phases.

This paper is organised as follows. Section 1 introduces the representation of the game data, which represent the input of our model. Section 2 provides our assumptions and motivation behind the model. Finally, Section 3 concludes with a discussion on the results and future work.

1 Game representation

We consider the task of modelling player movement during a soccer game. Our first objective is to know how to describe the game state at each time t. We first give some background on player positions. There are different possibilities when capturing player positioning. First, there is the absolute position which is defined as the player coordinates in the pitch. Also, one can capture the relative position, which is defined as the horizontal and vertical ordering of the player among the other players. The strategy we are using here is to capture both the absolute and relative positions of the players and the ball, combined with their velocities and accelerations, that will be defined later in this section. To construct our datasets, we collect different features of each player, that will be stored using a collection of matrices.

1.1 Indexes

We want to be able to represent the relative positions of the 22 players and the ball. To this aim, we will see the game as a 3-teams game, with the ball being the only player in its own team. We will note $P_1, \ldots, P_{11}$ the players of the Home team, $P_{12}, \ldots, P_{22}$ the players of the Away team and P_{23} for the ball. A player/ball is said to have indexes $\text{Index}(P_i) = (ver_i, hor_i)$, with ver_i and hor_i being respectively the vertical and horizontal index of the ordering of player i respectively among the vertical and horizontal axes. We have

$$\text{Index}(P_i) \in \{1, ..., 23\}^2 \quad \forall i \in \{1, ..., 23\}$$

We now define a class of matrices $\mathcal{M}$ such that for every matrix $M \in \mathcal{M}$:
$\cdot M \in \mathcal{R}^{23\times 23}$
$\cdot$ Each row and column has exactly one non-empty cell
Each matrix in this class will differ on the localisation and the value of the non empty cells. Let $M \in \mathcal{M}$, $1 \leq k, l \leq 23$:

$$M_{kl} = \begin{cases} f_i & \text{if Index}(P_i) := (ver_i, hor_i) = (k, l) \\ & \text{otherwise} \end{cases}$$

with f_i representing a a coordinate of a feature of player i.

1.2 Features

There is 5 different features we want to store for the players:

- Team Label: $f_i \in \{1, -1, B\}$ for a player being respectively in the Away Team, Home Team or the Ball. For implementation purposes we would have to consider B has a constant. We set $B = 23$.

- Role: the role of the player among the team's composition is crucial to understand his behaviour. This feature is a 2 dimensional vector summarizing the average relative position of the player among his own team. In other words, we consider the ordering of his position along the horizontal and vertical axis among his teammates. For a player i we get that $r_i \in \{1, ..., 10\}^2$ with

  $$r_i = (a, b),$$

  where a and b represent the coordinates of the player i at the horizontal and vertical axis respectfully, relative to the positions of the other players on its team.
  As mentioned earlier, this feature is taken as the average position of the player in the match, i.e. This feature is constant for the states of the same match. We also note that this definition doesn't concern for the ball and goalkeeper since these players have a unique role. For implementation purposes, we choose the the horizontal axis changes in function of the team considered, such that in general the goalkeeper has horizontal axis = 1.

- Position: A Player i has a position in 2 dimensional polar coordinates

$$p_i := (x_i,\ y_i),$$

representing the distance and the angle of a player to the center of the field.

- The velocity is computed as being the difference of 2 positions separated by a given time step. we get

$$v_i := (\dot{x_i},\ \dot{y_i}) = \left[\frac{x_i^t - x_i^{t-\Delta_t}}{\Delta t},\ \frac{y_i^t - y_i^{t-\Delta t}}{\Delta t}\right].$$

- Acceleration

$$a_i := \dot{v_i} = (\ddot{x_i},\ \ddot{y_i}) = \left[\frac{\dot{x_i}^t - \dot{x_i}^{t-\Delta t}}{\Delta t},\ \frac{\dot{y_i}^t - \dot{y_i}^{t-\Delta t}}{\Delta t}\right].$$

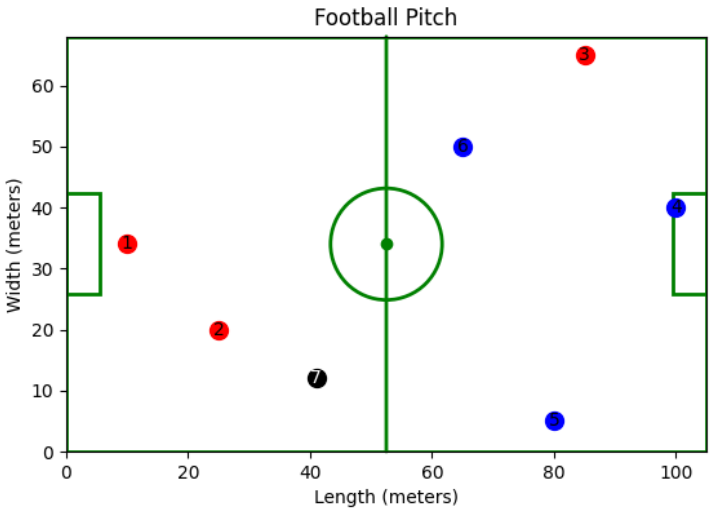
Combining these features, we obtain a vector of 9 dimensions.

States

Let $M^t,\ M^{r,1},\ M^{r,2},\ M^{p,1},\ M^{p,2},\ M^{v,1},\ M^{v,2},\ M^{a,1},\ M^{a,2} \in \mathcal{M}$. Each one of these matrices are storing one value of the 9-dimensional features with M^t storing the Team Labels, $M^{pr,1},\ M^{r,2}$ are storing the roles coordinates of the players,$M^{p,1},\ M^{p,2}$ storing the 2 positions coordinates, $M^{v,1},\ M^{v,2}$ for the velocities and $M^{a,1},\ M^{a,2}$ for the acceleration coordinates. The main reason we choose such a representation of the features is to be able to capture the relative positions of all the players on the pitch. We finally get the following set of game states $\mathcal{S}$ such that $\forall s \in \mathcal{S}$:

$$s = \left\{M^t,\ M^{r,1},\ M^{r,2},\ M^{p,1},\ M^{p,2},\ M^{v,1},\ M^{v,2},\ M^{a,1},\ M^{a,2}\right\}.$$

We now give a small example: For illustration purposes, suppose soccer players become extremely fast and strong and the FIFA has to set the number of players in a team to 3 players. The matrices are then of size 7×7 (2 times 3 players and the ball. Consider the following game situation:



Considering this football pitch, we can compute the polar coordinates and team label and get the following features and matrices $M^t,\ M^{r,2},\ M^{p,1} \in s$.

Player	(x, y)	Team Label	Role
1	$(10, 34)$	1	(1,2)
2	$(25, 20)$	1	(2,1)
3	$(85, 65)$	1	(3,3)
4	$(100, 40)$	-1	(1,2)
5	$(80, 5)$	-1	(2,1)
6	$(65, 50)$	-1	(3,3)
7	$(41, 12)$	23	(3,2)

$$M^t = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 23 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 \end{bmatrix}, \quad M^{r,2} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 \end{bmatrix}$$

$$M^{p,1} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 44.93 & 0 \\ 0 & 0 & 0 & 22.64 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 47.89 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 42.49 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 30.84 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 25.97 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 50.93 & 0 & 0 \end{bmatrix}$$

2 Ghosting Algorithm

A ghosting algorithm is an algorithm of prediction, i.e. we want to be able to determine the possible future states of the game. To do so, we will use the game representation presented before to predict the future positions of the 22 players and the ball.

However, two problems are arising when dealing with such a model. First, the game representation is high-dimensional. Indeed, we have 9 matrices of size 23×23 which results in 4761 entries. Secondly, time dependency is crucial in soccer analysis as it captures the evolving dynamics of matches, enabling the understanding of how game states transition over time. By considering the temporal sequence of events, we can extract valuable insights into player strategies, team behavior, and performance trends, leading to more accurate predictions, informed decision-making, and effective performance evaluation.

2.1 Dimension Reduction

The goal of this section is to create a meaningful representation of the game states into a lower-dimensional latent space. This problem of dimension reduction can be solved using different types of models depending on the task requirements. In our case the dimension reduction is unsupervised, which means that we have to analyze data without labeled responses, seeking to find patterns or groupings on their own. It's useful for tasks like identifying clusters of similar data points or reducing the dimensionality of complex datasets. Furthermore, the definition of the game states as before can be considered as images of size 23×23 with 7 channels. Thus, we want to be sure that our model captures the properties of indexes of our game representation.

Autoencoder

A variational autoencoder learns to encode input data into a low-dimensional latent space, capturing its essential features, before decoding it back into its original form. It does this by simultaneously training an encoder network to generate the latent space representation and a decoder network to reconstruct the input data from that representation. Let $\mathcal{L}$ be the Latent space with $dim(\mathcal{L}) = 10$. We define the Encoder and Decoder networks as

$$E : \mathcal{S} \mapsto \mathcal{L}, \quad D : \mathcal{L} \mapsto \mathcal{S}.$$

D and E are Neural Networks, which depend respectively on their parameters θ_D and θ_E. In particular, since our goal is to be able to encode the state space $\mathcal{S}$ in $\mathcal{L}$, we want to train our networks such that $\forall s \in \mathcal{S}$, $s = D(E(z))$. Thus, we are looking for the parameters minimizing the difference between s and $D(E(z))$:

$$\theta_D^\star, \theta_E^\star = \text{argmin}_{\theta_D,\ \theta_E} ||s - D_{\theta_D}(E_{\theta_E}(s))||^2$$

With for two states in $\mathcal{S}$ $s_1,\ s_2$ we have $||s_1 - s_2|| = \sum_{i=1}^{9} ||M_1^i - M_2^i||$.

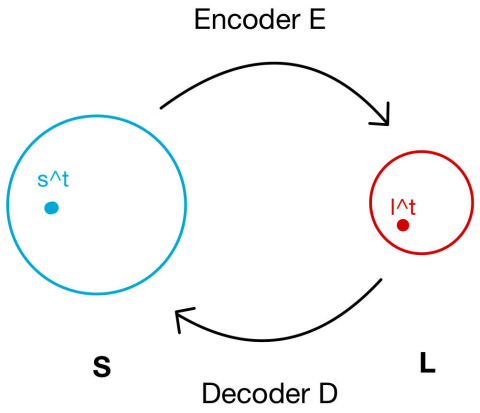


Figure 1: Scheme

Convolutional Neural Network

Once we know the global strategy to perform an unsupervised dimension reduction, we need to define and train both encoder and decoder neural networks.

The solution to the first problem is to use a dimension reduction model. Indeed, the reason defined a game state as 9 matrices instead of one matrix with 9-dimensional entries is to consider the state as an image with 9 channels. We choose to bring the states into a latent space of dimension 10 using a combination of intermediate layers.

Encoder:

Step 1: Input Layer	Input Dim: $9 \times 23 \times 23$
Step 2: Convolutional Layer	Output Dim: $32 \times 23 \times 23$
Step 3: Convolutional Layer	Output Dim: $64 \times 23 \times 23$
Step 4: Max Pooling Layer	Output Dim: $64 \times 11 \times 11$
Step 5: Flattening	Output Dim: $1 \times 6 * 64 * 128$
Step 6: Linear Layer	Output Dim: 1×128
Step 7: Linear Layer	Output Dim: 1×10

Decoder:

Step 1: Input Layer	Input Dim: 1×10
Step 2: Linear Layer	Output Dim: 1×128
Step 3: Linear Layer	Output Dim: $1 \times 6 * 64 * 128$
Step 4: Un-flattening	Output Dim: $64 \times 11 \times 11$
Step 5: Transposed Convolutional Layer	Output Dim: $32 \times 23 \times 23$
Step 6: Transposed Convolutional Layer	Output Dim: $9 \times 23 \times 23$

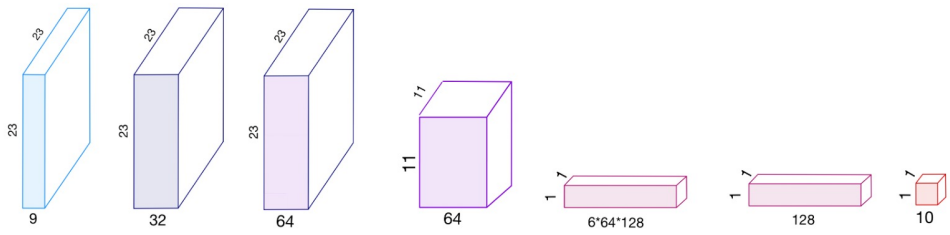


Figure 2: Scheme of The Dimension Evolution of the Convolutional Network Encoder

Let $\mathcal{S}$ be the total state space sampled from data. We can now construct our Convolutional Autoencoder Algorithm to create our Latent space $\mathcal{L}$.

LSTM

Now that we have successfully reduced the dimension of our state space while retaining the essential features, our objective is to build a predictive model for player positions. This model will consider both the representation of the

Algorithm 1: CAE

initialize the models E and D
for *each epoch* **do**
  for *each state* $s \in \mathcal{S}$ **do**
    $\cdot\ l = E(s)$
    $\cdot\ \tilde{s} = D(l)$
    $\cdot$ Compute Loss $||s - \tilde{s}||^2$
    $\cdot$ Update parameters $\theta_E,\ \theta_D$ using stochastic gradient descent

game and the temporal dependencies of past states. By leveraging the lower-dimensional space, which encapsulates meaningful information about the game state, we aim to develop a robust framework for forecasting player positions. This approach allows us to capture the dynamics of the game effectively, enabling more accurate predictions of player movements over time.

Let $s^t \in \mathcal{S}$ the game state at time t. Our goal is to construct an algorithm that given the previous T states can predict the future states.

$$\left[s^{t-T\Delta t}, ..., s^{t-\Delta t}\right] \mapsto s^{t+\Delta t}$$

The model we use to capture this time dependency is a Recurrent neural Network, In particular Long Short Term Memory Neural Networks (LSTM).

Player

Soccer is a highly complex and dynamic game influenced by numerous factors such as player positions, team strategies, opponent tactics, and environmental conditions. To accurately predict the future position of a player on the field, it is essential to consider their specific role within the team structure. For instance, a forward's movements and positioning differ significantly from those of a midfielder or defender. Indeed, constructing distinct predictive models tailored to each player's role can enhance the precision and effectiveness of the predictions. By incorporating role-specific features and strategies, such as attacking patterns for forwards or defensive positioning for defenders, these models can better capture the nuances of player behavior, leading to more insightful and reliable predictions.

Therefore, we define a total of 27 different models, each one of them to capture the specificity of each position with 25 of them corresponding to the possible in-field positions, 1 for the goalkeeper and 1 for the ball. Indeed, the 10×10 average position role matrix can be seen as a 5×5 representing each player average position. As an example, let Average_Compo_Home$\in \mathcal{R}^{10\times10}$, and Average_Compo_Home$[i][j] = player_k$. Then, the average position of player k is $(\lfloor i/2 \rfloor, \lfloor j/2 \rfloor)$.

Algorithm

Each of the 27 LSTM models integrates both the latent space representation and the features of the target player as inputs. This results in a 17-dimensional input layer derived from the combination of the player features (team label, position, velocity, acceleration) and the state representation, while producing a 2-dimensional output layer. The reason we only output the position instead of the whole features is that the velocity and the acceleration can be fully computed considering the predicted position and the past features.

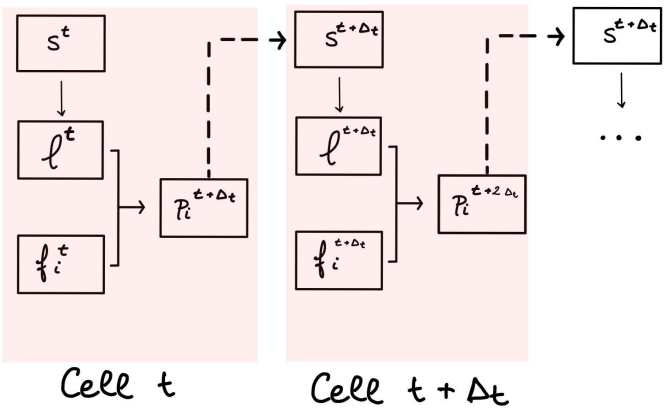


Figure 3: LSTM cells for prediction

For implementation purposes, we consider the data as sequences of T Δt-separated frames, each sequence is noted Seq_T^t starting at time t. Each model is trained using the behaviour of players at the corresponding position.

Algorithm 2: LSTM Training

initialize the 27 models;
for *each* Seq_T^t **do**
  for *each player i=1,...,23* **do**
    · assign the player i (in-field players, goalkeepers, ball) to the corresponding model
    for *t=1,...,L* **do**
      for *j=1,...,k-1* **do**
        · Compute state $s^{t+j\Delta t}$
        · $l^{t+j\Delta t} = E(s^{t+j\Delta t})$
        · Get features $f_i^{t+j\Delta} = [p_i,\ v_i,\ a_i]$ of player i
        · Flatten the vector $[f_i^{t+j\Delta},\ l^{t+j\Delta}]$
        · Compute $p_i^{t+j\Delta}$
      · Compute loss $\sum_{j=0}^{k} ||\tilde{p}_i^{t+j\Delta t} - p_i^{t+j\Delta t}||^2$ (3)
      · update parameters of the corresponding model (4)

Algorithm 3: LSTM Training

initialize the 27 models;
for *each player i=1,...,23* **do**
  · assign the player i (in-field players, goalkeepers, ball) to the corresponding model
  for *each epoch* **do**
    for *each* Seq_T^t **do**
      · Compute $\tilde{p}_i^{t+j\Delta}$
      · Compute Loss $MeansquareError(\tilde{p}_i^{t+j\Delta t} - p_i^{t+j\Delta t})$

Remark 2.1 *In the process of defining the states and constructing the model, our initial approach involved a fixed time interval L=2T, where we predicted the next T states based on the preceding T states. Nevertheless, the integration of LSTM introduces a valuable flexibility – the ability to use input sequences of varying lengths. This newfound capability empowers us to consider more contextually meaningful sequences, specifically those aligning with distinct game phases. Such sequences would start at notable events such as a kick-off, a foul, or a corner kick.*

To summarize, we should consider our prediction model as a combination of several model, each one used for a specific position. This is strategy used to compute the possible outcomes of of a game sequence. Here is an algorithm to predict the future T states:

Algorithm 4: Prediction

```
· Assign each player to a position
for k=1,...,T do
    · Compute states s^{t−TΔt}, ..., s^{t+(k−1)Δt}
    for j=1,...,23 do
        · Output position p_j^{t+kΔt} using corresponding model
    · Use all predicted positions to compute state s_j^{t+kΔt}
```

Conclusion

In conclusion, our exploration of ghosting in soccer analytics has provided theoretical insights into player movement patterns and positional effectiveness, enhancing our understanding of soccer dynamics. Additionally, our ghosting algorithm allows for the simulation of alternative realities, offering opportunities for scenario analysis and strategic planning. Overall, continued innovation in methodologies will further enhance player valuation and strategic decision-making in soccer analytics.

While our algorithm offers a meaningful representation of soccer games, there's still room for improvement. The implementation of this model has been made using the tracking data from [1]. To ensure our model's reliability and effectiveness, it's essential to conduct efficiency tests and validation procedures across diverse datasets. While our current results show promise based on a single match, the model's performances are limited by memory required to deal with such datasets. However, extending running times such as the number of epoch would improve the model's ability to generalize and capture a wider range of patterns and trends in real-world scenarios. It would be valuable to compare the efficiency of our current model to one that doesn't incorporate the latent space representation. By doing so, we can evaluate the effectiveness of including this

representation in our model architecture. This comparison will help us understand the impact of the latent state representation on the model's performance and determine whether it contributes significantly to the overall effectiveness of our approach.

References

[1] https://github.com/metrica-sports/sample-data/blob/master/readme.md.

[2] Yisong Yue Patric Lucey Le Hoang, Peter Carr. Data-driven ghosting using deep imitation learning. 2016.

[3] Derrick Yam. Attacking contributions: Markov models for football. 2019.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every written paper or thesis authored during the course of studies. In consultation with the supervisor, one of the following three options must be selected:

- (●) I confirm that I authored the work in question independently and in my own words, i.e. that no one helped me to author it. Suggestions from the supervisor regarding language and content are excepted. I used no generative artificial intelligence technologies[1].
- ( ) I confirm that I authored the work in question independently and in my own words, i.e. that no one helped me to author it. Suggestions from the supervisor regarding language and content are excepted. I used and cited generative artificial intelligence technologies[2].
- ( ) I confirm that I authored the work in question independently and in my own words, i.e. that no one helped me to author it. Suggestions from the supervisor regarding language and content are excepted. I used generative artificial intelligence technologies[3]. In consultation with the supervisor, I did not cite them.

Title of paper or thesis:

Introducing a new soccer Ghosting Algorithm

Authored by:
If the work was compiled in a group, the names of all authors are required.

Last name(s):	**First name(s):**
Georgenthum	Hugo

With my signature I confirm the following:
- I have adhered to the rules set out in the Citation Guide.
- I have documented all methods, data and processes truthfully and fully.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for originality.

Place, date	**Signature(s)**
Zürich, 29/02/24	

If the work was compiled in a group, the names of all authors are required. Through their signatures they vouch jointly for the entire content of the written work.

[1] E.g. ChatGPT, DALL E 2, Google Bard
[2] E.g. ChatGPT, DALL E 2, Google Bard
[3] E.g. ChatGPT, DALL E 2, Google Bard