Predicting Player Trajectories in Shot Situations in Soccer

Per Lindström[1,2], Ludwig Jacobsson[2], Niklas Carlsson[1], and Patrick Lambrix[1]

[1] Linköping University, Sweden, {niklas.carlsson,patrick.lambrix}@liu.se
[2] Signality, https://www.signality.com/

Abstract. Player behaviors can have a significant impact on the outcome of individual events, as well as the game itself. The increased availability of high quality resolution spatio-temporal data has enabled analysis of player behavior and game strategy. In this paper, we present the implementation and evaluation of an imitation learning method using recurrent neural networks, which allows us to learn individual player behaviors and perform rollouts of player movements on previously unseen play sequences. The method is evaluated using a 2019 dataset from the top-tier soccer league in Sweden (Allsvenskan). Our evaluation provides insights how to best apply the method on movement traces in soccer, the relative accuracy of the method, and how well policies of one player role capture the relative behaviors of a different player role, for example.

1 Introduction

In recent years the availability of tracking data has grown considerably, through the use of wearable GPS trackers and advances in optical object tracking. This has made it possible for analysts to provide deeper insights than before. For example, in soccer, work has been done on such issues as player and team analysis [3, 2], player ghosting [26], and predicting ball position [23].

An important factor that can be learned from this data is player movement. Once we can learn and predict player movement, interesting questions can be asked and answered. For instance, one may obtain insights on a particular player's movement patterns in a particular situation and thereby have a better chance to find strategies to counter that player. It may also be possible to find players that have similar patterns and may therefore be seen as possible substitutes for one another. By substituting the patterns of two players it may also be possible to see how different players act in and solve different game situations. Comparing a player to a league average player or to a league average player with a certain role, may give insights on a player's creativity, but could also be used in training to show a player when good or bad decisions are made. Further, defensive and offensive effectiveness in different situations may be investigated.

In this paper we present a method to predict player movement based on imitation learning (section 4). Through experiments (section 5) we give technical insights as well as validation and player comparisons. The method is evaluated using a 2019 dataset from the top-tier soccer league in Sweden (Allsvenskan).

Our results highlight how to best apply the method on movement traces from soccer so to achieve good accuracy, validate the accuracy of the models (e.g., as function of time since rollout starts and player role), and provide insights regarding how well different player-specific policies (learned using our model) capture the relative behaviors of different player roles (e.g., similarities and differences between different player roles) and illustrate how different player behaviors can be compared with the use of rollout using different player profiles.

2 Related Work

In many sports (but for the sake of brevity we focus on soccer), work has started on game play. For instance, in soccer, there is work on rating actions [9, 36], pass prediction [21, 5, 7, 39, 15], shot prediction [29], expectation for goals given a game state [8] or a possession [11], or more general game strategies [28, 17, 12, 14, 4, 16, 1]. Game play information can be used to predict the outcome of a game by estimating the probability of scoring for the individual goal scoring opportunities [10], by relating games to the players in the teams [30], or by using team rating systems [24]. It can also be used to measure player performance by investigating in metrics [37], skills [22, 40, 31, 41], performance under pressure [6] and on the correlation with market value [18]. Tracking data is used for player and team analysis [3, 2], game analysis [13], and predicting ball position [23]. An interactive system allowing for querying and analyzing tracking data is described in [38]. The work closest related to ours is [27, 26] where imitation learning is used to learn the general behavior of inferred roles or in a multi-agent setting. In our work we use imitation learning to learn the behavior of specific players.

3 Background

In this work we use two learning phases each using a version of imitation learning. In imitation learning (e.g., [32]) a *policy* π is trained to imitate the behavior of an expert. The setting for the learning contains *states* representing what an agent perceives at a given time step, and *actions*, that allow us to go from one state to another. The expert π^* generates the observed state-action pairs which the learned policy should imitate. In the first phase we use behavioral cloning (e.g., [35]), where the policy maps states to actions. In the second phase we learn a policy that maps a state and a set of contexts to a *trajectory with horizon* T which consists of a set of T states (and actions). The latter mapping is also called a *rollout*, and is achieved by sequentially executing π given an initial state. Two distributions of states are used to train the policies. A distribution of states visited by the expert is defined as $P^* = P(x|\pi^*)$, while a distribution of states induced by policy π_θ, parameterized by θ, is given by $P(x|\theta)$. The distribution P^* is provided from observations of the expert and the cost is given by $C(\pi^*(x), \pi(x))$. As it may not be possible to know or observe the true cost function for a task, a surrogate loss ℓ, is adopted, which can be minimized instead of C. The learning goal for behavioral cloning (first phase) is then to find the

parameters, which make the policy imitate the expert with minimal surrogate loss. The best policy $\hat{\pi}_\theta$ is then given by $\hat{\pi}_\theta = \text{argmin}_\theta \, E_{x \sim P^*} \ell(\pi^*(x), \pi_\theta(x))$. In general imitation learning (second phase), the distribution of states is instead given by rollouts and the best policy is $\hat{\pi}_\theta = \text{argmin}_\theta \, E_{x \sim P(s|\theta)} \ell(\pi^*(x), \pi_\theta(x))$.

4 Approach

Model. In our setting the expert in the imitation learning is a soccer player that we want to model. A state $x \equiv (s, c)$ is comprised of two parts: the system state s describing the modelled player, and the context c describing all other players and the ball. In our work a state is represented by a state vector. For each entity in a state, i.e., all players on the field and the ball, we store the position on the field as Cartesian coordinates. The entities in each state are sorted so that their order in the state is consistent over the duration of a sequence. This is needed for learning and interpreting changes of an entity over time. The players in each state are grouped per team, and within their team they are sorted according to their lineup index provided by the league. Lineup indices are consistent through games, and thus also consistent through any subsequence within a game. The modelled player p_m, is placed at the beginning of the state vector. This is done in order to explicitly show the model which player that is being modelled, since its position in the lineup may differ between games. p_m is followed by all players in his team, all players in the opponent team and finally the ball. Further, an action in a state results in a change in position.

The policies used in our experiments are implemented in Keras (https://keras.io/) as multi-variable regression models. Given a state x_t where t represents the order in a sequence of states, a policy π predicts an action $\hat{a}_t = \pi(x_t)$ which describes how p_m's position has changed in $t + 1$. When a policy predicts an action $\hat{a}_t$ on state x_t, it receives as input a sequence of state vectors representing states $x_{t-w}, ..., x_t$. This sequence can be thought of as the memory of the model, as information from the states in this sequence can be used for deciding on the prediction of the action. The size w of this sequence is called the *window*. Our model is implemented using a Long Short-Term Memory (LSTM) network with two LSTM layers with 512 nodes in each layer (similar to [26, 27]). The model has a total of 3,253,250 trainable parameters.

Sequences are rolled out by feeding consecutive states into a policy and iteratively making predictions on those while updating the next system state given the current action. The predicted sequence $\hat{X}$ is initiated with the first w state vectors of the observed sequence X: $\hat{X}_{0:w} = X_{0:w}$ The predicted sequence $\hat{X}$ can then be rolled out by predicting an action $\hat{a}_t = \pi(\hat{x}_t)$ using $\{\hat{x}_{t-w}..., \hat{x}_t\}$. The following system state s_{t+1} in $\hat{x}_{t+1}$ is updated with the result of $\hat{a}_t$. This process is applied iteratively on each $\hat{x} \in \hat{X}$, from $t = w$ to $t = T$, where T is the horizon. Each state is updated with the predicted action, which means that any deviation (or error) introduced by the policy is propagated through the sequence. A policy π is evaluated by rolling it out on a set of sequences $\mathcal{S}$ over a horizon of T time steps. During the rollout of each sequence X, the Euclidean

distance d_t between p_m's position in x_t and $\hat{x}_t$ is calculated for each time step t. The distance d_t is referred to as the *error*. The mean error at t for a set of sequences $\mathcal{S}$ can be obtained by averaging the errors for all sequences in $\mathcal{S}$ at time step t. The global error is calculated by averaging the mean errors for all time steps in a rollout.

Algorithm 1 Sequence training

Input: Training sequences $\mathcal{S}_{tr}$, Validation sequences $\mathcal{S}_v$,
Input: Pre-trained policy π_0,
Input: Training horizon T_h, Epochs N
Output: Policy π

```
1: D ← ∅
2: S_0 ← S_tr
3: for j = 1 to T_h do
4:     S_j ← S_{j−1}
5:     for X = {x_0, ..., x_T} in S_{j−1} do
6:         for t = 0 to T_h − 1 do
7:             Predict â_t = π_0(x_t)
8:             Calculate x̂_{t+1} = calculate_state(â_t, x_t, x_{t+1})
9:             Add (x̂_{t+1}, a_{t+1}) to D
10:            Replace x_{t+1} in X with the generated state x̂_{t+1}
11: for i = 1 to N do
12:     π_i ← π_{i−1}
13:     Train π_i on D
14:     Validate π_i on S_v
15: return Best π_i on sequence validation loss
```

Learning. The training process is divided into two phases. During both phases, mean squared error is adopted to calculate loss. In the first phase the policy is pre-trained on state-action pairs from training sequences. This is classical supervised learning and teaches the policy the mapping from observed states x_t to expert actions a_t. The policy is trained during N epochs and the best policy is returned, based on validation loss. The validation data $\mathcal{D}_v$ is made up of all state-action pairs from the validation sequences. In the second phase the policy is trained on (partially) rolled out sequences as described in Algorithm 1. The approach is based on DAgger [33] and the imitation learning algorithms used in [26, 27]. In the first part of the algorithm (lines 3-10) the set of training data from the first phase is augmented by (partially) rolling out those sequences. This is done by using the pre-trained policy π_0 to gradually make one-step predictions on each time step t in each sequence X of the training sequences $\mathcal{S}_{tr}$, adding all generated state-action pairs to a new dataset $\mathcal{D}$, and then iterating over the sequences again with the states generated in the last round. This process is performed T_h times as specified by the training horizon parameter. The training horizon should be lower or equal than the sequence horizon; $T_h \leq T$. The *calculate_state* function on line 8 calculates a predicted state $\hat{x}_{t+1}$ where the

Table 1. Summary of dataset.

Season	Period	Games	Players	Events	Goals	Other shots
2019	March 31 - June 2	79	276	1,668	193	1,475

system state of $\hat{x}_{t+1}$ is based on the current system state s_t and action $\hat{a}_t$, and the context of $\hat{x}_{t+1}$ is the context c_{t+1} of x_{t+1}.
In the second part (lines 11-14) the policy is trained on this augmented data set and validated by rolling out on full sequences in a validation set using their global errors as validation loss. Finally, the best policy is returned based on sequence validation loss, which is the global error given by rolling out on the validation sequences. Although many single-agent imitation learning applications use dynamic oracles [33, 27] to generate one-step corrections a_t^* for each sampled state $\hat{x}_t$, dynamic oracles are expensive and require complex training schemas, and there has been research on circumventing the need for dynamic oracles when learning to roll out sequences [25, 19]. Algorithm 1 follows their example by using the observed expert demonstrations to teach the policy how to behave in the sampled states.

5 Experiments

5.1 Data and data preparation

Data. The dataset used in this paper was provided by Signality, a startup company that delivers sports analytics for customers such as the top-tier soccer league in Sweden. The dataset captures all 79 games played before the summer break of the 2019 season, and includes tracking data, game metadata, and manually annotated event data. The tracking was created by applying Signality's automated object detection and recognition technology on game video recordings. The tracking data has a sample rate of 25 Hz and contained trajectories of all players and the ball. Each trajectory contains position, speed and a trajectory identifier at every time step. All positions are represented by Cartesian coordinates (x,y) $\in$ [0,1] $\times$ [0,1]. Each trajectory is assigned a jersey number as soon as it is recognized by the system. The tracking data also contains basic statistics such as each player's accumulated distance, passes and ball possession. The game metadata includes information such as arena information, pitch size, and team lineup, as well as information about the players and teams in the game. This data is used to create a mapping from trajectory identifiers to player identities, roles and lineup. The manually annotated data contains game events such as shots, yellow cards, medical treatment, substitutions, and goals. These events are used to, e.g., find sequences of interest to train and evaluate policies. We preprocessed the data by padding missing values or entities with the value -1, by scaling the coordinates to the size of an average Swedish soccer pitch (105×64 m^2), and downsampling the sample rate to 5 Hz, or one frame per 200 ms.
Training and validation data. In this paper we focus on situations related to goals and shots. The training and validation data are therefore sequences related

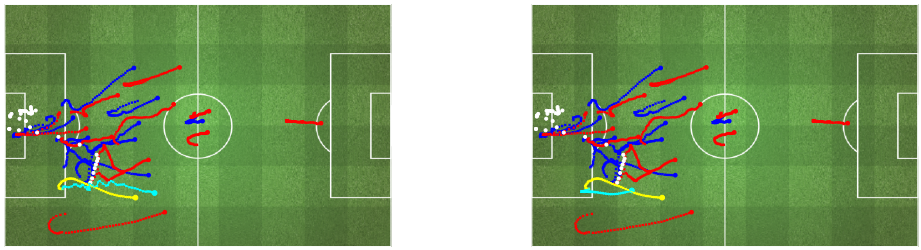
to such situations. We extracted sequences starting 10s before and ending 10s after an annotated shot, and for which each state in the sequence contains the observed coordinates for p_m. Each sequence contains up to 100 observed states. The modelled team can either be the defending team or the attacking team in the extracted sequences. Table 1 summarizes the key characteristics associated with the dataset. For the analysis presented here, we focus on the 150 players (out of 276) with the most games (in this case 12-24 games per player). From the 1,668 annotated events, play sequences for the 150 players were added randomly to both the training and validation dataset. For the training dataset, we extracted randomly two sequences from each game half (when at least three annotated sequences existed) and assigned the rest of the sequences to the training set. (When only two sequences existed, one was assigned for the evaluation, and when only one existed it was only used for training.) In total, 5,188 random sequences were used for validation and 21,284 sequences were used for training. All arenas and lineups are represented during both training and validation.

Visualization. To reason regarding the behavior of a policy, rollouts are visualized as plots on a soccer pitch. Each dot is one observed data point. The spacing between each dot in a trajectory is 200 ms. Yellow dots are the ground truth, or the observed player trajectory. Turquoise dots are the rolled out policy, with the first dot coinciding with the player's actual initial state. The big dot of a trajectory shows the final position of that player. Red and blue dots represent positions of the other players, and white dots observed ball positions.

5.2 Technical evaluation insights

The calculations are resource intensive and take time. For this reason, we use limited player samples to first determining a promising learning methodology. Here, we highlight the insights obtained from three such steps.

Absolute vs. relative actions. Actions can be represented using absolute actions, i.e, using the coordinates of consecutive player positions (e.g., [26]), or relative actions, i.e, using the difference between the consecutive positions (e.g., [34]). We investigate the influence of the representation on the learned policy. For these experiments, we used five random players, and each policy was trained for each action type and player. For both training and validation we used a window size of 20 time steps. Over the 9,500 samples associated with these players the absolute method had an average error of 9.01m with a standard deviation of 7.22m. The corresponding error using the relative method was 6.89m (σ=5.84m). We note these values (both averages and standard variations) are smaller, that the 95% confidence intervals (i.e., [8.87,9.16] and [6.77,7.01]) are non-overlapping, and that the null hypothesis that the average errors are the same can be rejected with a t-score of 22.25 (>1.96). In general, we have found that the relative actions also result in smoother rollout behavior from the start. This is illustrated in Figure 1, which shows example rollouts using the two methods. Motivated by the smaller errors and smoother rollouts, in the following experiments, we use only the relative actions.



(a) Absolute (b) Relative

Fig. 1. Example rollouts.

We note that [26] has shown that it is possible to get smooth results with absolute actions, and we may be able to obtain this with other parameters, an extended state description, or more data. Further, a drawback of using relative actions without a dynamic oracle [27] is that the policy is taught to move in the same direction that the expert did in a certain time step even if it during the rollout in sequence training has deviated significantly from the original trajectory.

Baseline comparisons. We note that our methodology differs from most prior work on imitation learning in that we circumvent the need for dynamic oracles when learning to roll out sequences. While this still allows us to imitate the general behavior of players, this naturally results in the absolute positional error increasing over time. To put these errors in perspective, we compared the learned policies with policies generated using a random walk algorithm (providing a kind of upper bound) and the results obtained when evaluating the policies on the training data itself (providing a kind of rough lower bound). For this analysis we used the same set of five players as above. For the random walk, random steps were sampled from a normal distribution specified by the mean and standard deviation of movement observed in the dataset. With these settings, we obtained a global error of 10.17m (random walk) and 4.75m (training, relative actions), respectively. Despite a relatively unfair comparison, these results suggest that our relative action approach shows promising results in that it has errors closer to what the policy achieves on the training data itself than a random walk. Trajectories that are confined to a small area naturally exhibit lower errors for the random walk, while the opposite is true for longer trajectories. Furthermore, although the difference in global error for the learned policy and the random walk policy is not that high, a qualitative assessment of the rollouts makes it clear that for the random walk the rollouts are random, and the policy does not follow the general movement of the sequence at all, whereas the method presented here does (e.g., Figure 1).

Window size. To investigate the influence of the window size, multiple policies were trained with variations on this parameter. The window size limits the policy's memory and gives an upper bound on the longest time frame for retained information. Further, from a performance perspective, longer window sizes re-

Table 2. Impact of window size.

Window	Mean	Stdev	Conf interval
10	7.60	6.23	[7.47, 7.73]
20	7.14	5.70	[7.02, 7.26]
30	7.42	6.05	[7.30, 7.55]
40	7.72	6.04	[7.60, 7.85]
50	7.23	6.30	[7.10, 7.36]

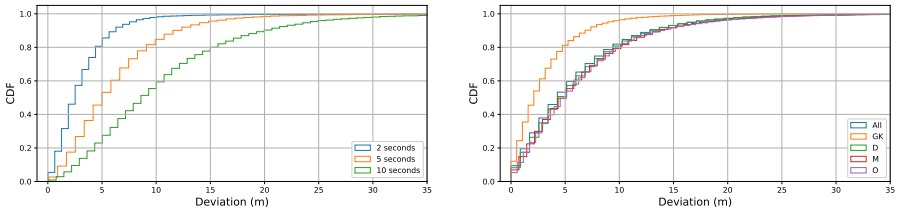
quire more computation during training and prediction [20]. Table 2 shows example results for five different window sizes. Interestingly, we have observed the highest accuracy with an intermediate window size of 20 time steps (our default window). While the relative errors for different window sizes are relatively similar, the window size of 20 time steps results in somewhat smaller averages than the other window sizes (statistically significant at the 95% confidence level compared to all windows except 50). It is also interesting to note that both the averages and the standard deviations are smallest for these intermediate window sizes. These results suggest that the policies are not able to fully utilize the additional information captured by the larger window sizes. In the following, we use a window size of 20 time steps.

Multi-player-based pre-training. We have observed significant value using multi-player based pre-training. To illustrate this, here we present evaluation results showing how pre-training on other players' observations compares to only pre-training on player p_m's observations, in terms of rollout error on validation sequences from p_m. For these experiments, we used five (new) random players and trained the modeled players using only data for the player itself (as in the prior experiments presented in this section) or using data for all players, respectively. To avoid influence of goalkeepers (who have much different movement patterns), we only used data for non-goalkeepers here. Again, we use a window size of 20 time steps and evaluate over 50 time steps, and use the relative actions. When only pre-training using the player itself we obtain an error of 7.12m (σ=5.96m), and when using pre-training using multiple players we obtain an error of 6.73m (σ=5.59m). Over 7,800 samples, this results in non-overlapping confidence intervals (i.e., [6.99,7.25] vs. [6.61,6.85]) and a t-value of 4.22 (>1.96).

5.3 Validation and player comparisons

Motivated by the above findings, in the remainder we use the relative actions, a 20 time step window, and multi-player pre-training for our learning and evaluation. For this section, we present results over all 150 players. Overall, over the 259,400 samples considered we observe an average error of 6.27m (σ=5.77m).

Error accumulation. Figure 2(a) shows the Cumulative Distribution Function (CDF) distributions of the error in the absolute player position after 2, 5, or 10 seconds have elapsed. It should be noted that the shots typically happen 6-8 seconds into the sequences, and can cause significant changes in players' actions



(a) Time (2, 5, 10s) (b) Position (G, D, M, O)

Fig. 2. CDFs of the relative errors.

Table 3. Error for different player roles.

Position	Mean (Stdev)	Players	Samples
Goalkeepers (G)	3.30 (3.30)	13	25,700
Defensive (D)	6.49 (5.69)	63	108,300
Midfielder (M)	6.69 (6.07)	45	76,350
Offensive (O)	6.71 (6.05)	29	49,050

on the pitch. For this reason, it is perhaps not surprising that the tail of the distribution (with larger errors) increases significantly between the 5 second and 10 second time stamps.

Player role breakdown. In general, we have observed similar prediction errors for all categories except goalkeepers. This is illustrated by the tight clustering of the other categories (D,M,O) CDFs in Figure 2(b), but can also be seen from looking at the average errors of the different categories (Table 3). Here, the defender (D) category includes all "backs", (M) includes "midfielders", and the offensive (O) category includes "forwards" or "wings" as annotated by Allsvenskan.

Direction errors. Despite the model targeting general behavior, rather then definite prediction, we have found that it often does a reasonable job predicting the direction of the player movement. Here, we present prediction results for the general direction, as defined by in which out of four directions the player moves the furthest: forward (i.e., towards the opposition's end), backwards, inwards (i.e., towards the middle of the field), or outwards. These general directions are split by four lines separated by 90°. Table 4 presents the confusion matrices for the movements after 2s, 5s, and 10s. For all cases, the highest values are along the diagonal, and the method has stable F1-scores of 0.53, 0,52, and 0.53.

Cross-evaluation. To investigate whether the policies have learned the behavior of the player it was trained on, policies trained on different players were cross-evaluated on each others' validation sequences. Table 5 shows the errors when using the model of one player to predict the movements of a different player. We note that the highest values are shown along the diagonal (or the players of the same player role). These results suggest that the best policies (in terms of global error) are achieved when using the policies for the specific player. We also noted that policies from players with similar roles often exhibit similar

Table 4. Confusion matrix for directional prediction errors over 2s, 5s, and 10s.

		Predictions			
		In	Out	Fwd	Bkwd
Truth	In	**0.24**	0.20	0.29	0.27
	Out	0.08	**0.39**	0.26	0.27
	Fwd	0.09	0.16	**0.67**	0.08
	Bkwd	0.09	0.20	0.08	**0.62**

(a) 2s (F1=0.53)

		Predictions			
		In	Out	Fwd	Bkwd
Truth	In	**0.26**	0.18	0.30	0.26
	Out	0.08	**0.42**	0.25	0.25
	Fwd	0.09	0.19	**0.62**	0.10
	Bkwd	0.09	0.21	0.09	**0.62**

(b) 5s (F1=0.52)

		Predictions			
		In	Out	Fwd	Bkwd
Truth	In	**0.20**	0.20	0.30	0.30
	Out	0.07	**0.46**	0.24	0.23
	Fwd	0.09	0.19	**0.64**	0.09
	Bkwd	0.09	0.19	0.10	**0.62**

(c) 10s (F1=0.53)

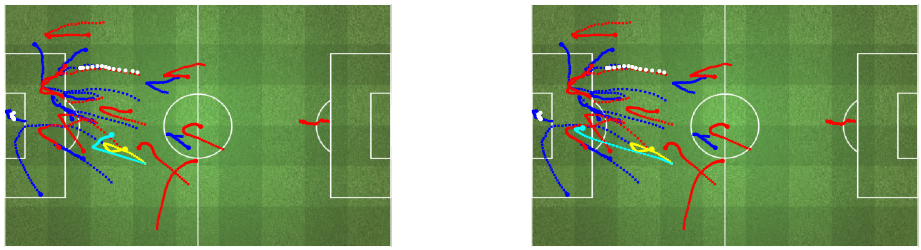
Table 5. Cross-evaluation using ten random players - errors.

		Observed expert player									
		G1	D1	D2	D3	D4	M1	M2	M3	O1	O2
Model player (Policy)	G1	**3.56**	8.33	7.82	10.22	10.96	8.83	11.54	10.01	10.02	7.75
	D1	7.1	**6.86**	6.46	7.96	7.63	7.35	9.51	7.28	7.82	7.22
	D2	6.71	8.05	**5**	7.25	7.77	8.03	10.04	8.01	8.19	8.75
	D3	4.63	7.85	5.63	7.19	7.74	8.13	8.69	7.34	7.24	6.8
	D4	14.17	16.05	10.98	13.82	**6.81**	12.15	13.18	12.74	12.08	10.84
	M1	4.24	8.04	5.94	7.08	7.67	**5.75**	8.48	6.4	7.07	5.82
	M2	5.61	8.69	6.75	7.4	7.26	7.14	**8.17**	6.16	7.46	7.27
	M3	4.98	7.54	5.79	**7.02**	7.22	6.27	**8.17**	**5.56**	6.58	**5.08**
	O1	5.73	8.69	7.23	8.14	7.65	6.76	8.31	6.39	**6.31**	6.33
	O2	4.63	8.22	7.92	8.99	8.62	8.75	9.9	8.06	8.74	5.6

behavior. The latter observations show that the policies in fact do capture different player behaviors. This opens the door for interesting future work in which one player may be exchanged for another player, in a given situation. This is illustrated in Figure 3. Here, we show the predicted (and actual) movement of an offensive player (Figure 3(a)), as well as the predicted movement of a defensive player put into the same situation (Figure 3(b)). In this situation the opposition shoots, and we note that the defensive player would run backwards much further towards the own goal.

6 Conclusion

In this paper we presented a method to predict player movement based on imitation learning, evaluated it using a large-scale 2019 dataset from the top-tier soccer league in Sweden (Allsvenskan). Our evaluation provides insights into how to best apply the method on movement traces in soccer. For example, we show that there is value in using relative actions, that a limited time window often is sufficient to achieve good accuracy, and that there is value in using data also from other players during the pre-training phase. We also validate the accuracy of the models (e.g., with regards to time and player role) and provide insights regarding how well different policies capture the relative behaviors of different player roles (e.g., similarities and differences between different player roles), gleaning some insights into how different player behaviors can be compared with the use



(a) Offensive player (b) Defensive player

Fig. 3. Example rollouts of the same play with different policies.

of rollout using different player profiles. The latter highlight the value of these types of policies and open interesting directions for future work, including investigations on the effect of pre-training only on similar roles as the modelled player. Further, another direction for future research is multi-agent modelling. By cross-updating all player states between each time step the multi-agent approach would model the full dynamics and interactions of the players and not just a single player given a situation.

References

1. Andrienko, G., Andrienko, N., Budziak, G., Dykes, J., Fuchs, G., von Landesberger, T., Weber, H.: Visual analysis of pressure in football. Data Mining and Knowledge Discovery **31**(6), 17931839 (2017). https://doi.org/10.1007/s10618-017-0513-2
2. Bialkowski, A., Lucey, P., Carr, P., Matthews, I.A., Sridharan, S., Fookes, C.: Discovering team structures in soccer from spatiotemporal data. IEEE Transactions on Knowledge and Data Engineering **28**(10), 2596–2605 (2016). https://doi.org/10.1109/TKDE.2016.2581158
3. Bialkowski, A., Lucey, P., Carr, P., Yue, Y., Sridharan, S., Matthews, I.: Large-Scale Analysis of Soccer Matches Using Spatiotemporal Tracking Data. In: Kumar, R., Toivonen, H., Pei, J., Huang, J.Z., Wu, X. (eds.) Proceedings of the 2014 IEEE International Conference on Data Mining. pp. 725–730 (2014)
4. Bojinov, I., Bornn, L.: The Pressing Game: Optimal Defensive Disruption in Soccer. In: 10th MIT Sloan Sports Analytics Conference (2016)
5. Brandt, M., Brefeld, U.: Graph-based Approaches for Analyzing Team Interaction on the Example of Soccer. In: Davis, J., van Haaren, J., Zimmermann, A. (eds.) Proceedings of the 2nd Workshop on Machine Learning and Data Mining for Sports Analytics. CEUR Workshop Proceedings, vol. 1970, pp. 10–17 (2015)
6. Bransen, L., Robberechts, P., Haaren, J.V., Davis, J.: Choke or shine? quantifying soccer players' abilities to perform under mental pressure. In: 13th MIT Sloan Sports Analytics Conference (2019)
7. Cintia, P., Rinzivillo, S., Pappalardo, L.: Network-based Measures for Predicting the Outcomes of Football Games. In: Davis, J., van Haaren, J., Zimmermann, A. (eds.) Proceedings of the 2nd Workshop on Machine Learning and Data Mining for Sports Analytics. CEUR Workshop Proceedings, vol. 1970, pp. 46–54 (2015)

8. Decroos, T., Dzyuba, V., Van Haaren, J., Davis, J.: Predicting Soccer Highlights from Spatio-Temporal Match Event Streams. In: Proceedings of the 31st AAAI Conference on Artificial Intelligence. pp. 1302–1308 (2017)
9. Decroos, T., Van Haaren, J., Dzyuba, V., Davis, J.: STARSS: A Spatio-Temporal Action Rating System for Soccer. In: Davis, J., Kaytoue, M., Zimmermann, A. (eds.) Proceedings of the 4th Workshop on Machine Learning and Data Mining for Sports Analytics. CEUR Workshop Proceedings, vol. 1971, pp. 11–20 (2017)
10. Eggels, H., van Elk, R., Pechenizkiy, M.: Explaining Soccer Match Outcomes with Goal Scoring Opportunities Predictive Analytics. In: van Haaren, J., Kaytoue, M., Davis, J. (eds.) Proceedings of the 3rd Workshop on Machine Learning and Data Mining for Sports Analytics. CEUR Workshop Proceedings, vol. 1842 (2016)
11. Fernandez, J., Bornn, L., Cervone, D.: Decomposing the immeasurable sport: A deep learning expected possession value framework for soccer. In: 13th MIT Sloan Sports Analytics Conference (2019)
12. Fernando, T., Wei, X., Fookes, C., Sridharan, S., Lucey, P.: Discovering Methods of Scoring in Soccer Using Tracking Data . In: Lucey, P., Yue, Y., Wiens, J., Morgan, S. (eds.) Proceedings of the 2nd KDD Workshop on Large Scale Sports Analytics (2015)
13. Gudmundsson, J., Wolle, T.: Football analysis using spatio-temporal tools. Computers, Environment and Urban Systems **47**, 16–27 (2014). https://doi.org/10.1016/j.compenvurbsys.2013.09.004
14. Gyarmati, L., Anguera, X.: Automatic Extraction of the Passing Strategies of Soccer Teams. In: Lucey, P., Yue, Y., Wiens, J., Morgan, S. (eds.) Proceedings of the 2nd KDD Workshop on Large Scale Sports Analytics (2015)
15. Gyarmati, L., Stanojevic, R.: QPass: a Merit-based Evaluation of Soccer Passes. In: Lucey, P., Yue, Y., Wiens, J., Morgan, S. (eds.) Proceedings of the 3rd KDD Workshop on Large Scale Sports Analytics (2016)
16. Haaren, J.V., Davis, J., Hannosset, S.: Strategy Discovery in Professional Soccer Match Data. In: Lucey, P., Yue, Y., Wiens, J., Morgan, S. (eds.) Proceedings of the 3rd KDD Workshop on Large Scale Sports Analytics (2016)
17. Haaren, J.V., Dzyuba, V., Hannosset, S., Davis, J.: Automatically Discovering Offensive Patterns in Soccer Match Data. In: Fromont, E., De Bie, T., van Leeuwen, M. (eds.) Proceedings of International Symposium on Intelligent Data Analysis. LNCS, vol. 9385, pp. 286–297 (2015)
18. He, M., Cachucho, R., Knobbe, A.: Football Player's Performance and Market Value. In: Davis, J., van Haaren, J., Zimmermann, A. (eds.) Proceedings of the 2nd Workshop on Machine Learning and Data Mining for Sports Analytics. CEUR Workshop Proceedings, vol. 1970, pp. 87–95 (2015)
19. Ho, J., Ermon, S.: Generative adversarial imitation learning. In: Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems 29, pp. 4565–4573 (2016), http://papers.nips.cc/paper/6391-generative-adversarial-imitation-learning.pdf
20. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**(8), 1735–1780 (1997). https://doi.org/10.1162/neco.1997.9.8.1735
21. Horton, M., Gudmundsson, J., Chawla, S., Estephan, J.: Classification of Passes in Football Matches using Spatiotemporal Data. In: Lucey, P., Yue, Y., Wiens, J., Morgan, S. (eds.) Proceedings of the 1st KDD Workshop on Large Scale Sports Analytics (2014)
22. Jordet, G., Bloomfield, J., Heijmerikx, J.: The hidden foundation of field vision in English Premier League (EPL) soccer players. In: 7th MIT Sloan Sports Analytics Conference (2013)

23. Kim, K., Grundmann, M., Shamir, A., Matthews, I.A., Hodgins, J.K., Essa, I.A.: Motion fields to predict play evolution in dynamic sport scenes. In: The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010. pp. 840–847 (2010). https://doi.org/10.1109/CVPR.2010.5540128
24. Lasek, J.: EURO 2016 Predictions Using Team Rating Systems. In: van Haaren, J., Kaytoue, M., Davis, J. (eds.) Proceedings of the 3rd Workshop on Machine Learning and Data Mining for Sports Analytics. CEUR Workshop Proceedings, vol. 1842 (2016)
25. Le, H., Kang, A., Yue, Y., Carr, P.: Smooth imitation learning for online sequence prediction. In: Balcan, M.F., Weinberger, K.Q. (eds.) Proceedings of The 33rd International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 48, pp. 680–688 (2016), http://proceedings.mlr.press/v48/le16.html
26. Le, H.M., Carr, P., Yue, Y., Lucey, P.: Data-driven ghosting using deep imitation learning. In: 11th MIT Sloan Sports Analytics Conference (2017)
27. Le, H.M., Yue, Y., Carr, P., Lucey, P.: Coordinated multi-agent imitation learning. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017. pp. 1995–2003 (2017), http://proceedings.mlr.press/v70/le17a.html
28. Lucey, P., Bialkowski, A., Carr, P., Foote, E., Matthews, I.: Characterizing Multi-Agent Team Behavior from Partial Team Tracings: Evidence from the English Premier League. In: Hoffmann, J., Selman, B. (eds.) 26th AAAI Conference on Artificial Intelligence. pp. 1387–1393 (2012)
29. Lucey, P., Bialkowski, A., Monfort, M., Carr, P., Matthews, I.: Quality vs Quantity: Improved Shot Prediction in Soccer using Strategic Features from Spatiotemporal Data. In: 9th MIT Sloan Sports Analytics Conference (2015)
30. Maystre, L., Kristof, V., Ferrer, A.J.G., Grossglauser, M.: The Player Kernel: Learning Team Strengths Based on Implicit Player Contributions. In: van Haaren, J., Kaytoue, M., Davis, J. (eds.) Proceedings of the 3rd Workshop on Machine Learning and Data Mining for Sports Analytics. CEUR Workshop Proceedings, vol. 1842 (2016)
31. Nsolo, E., Lambrix, P., Carlsson, N.: Player valuation in european football. In: Brefeld, U., Davis, J., Haaren, J.V., Zimmermann, A. (eds.) Proceedings of the 5th Workshop on Machine Learning and Data Mining for Sports Analytics co-located with 2018 European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2018), Dublin, Ireland, September 10th, 2018. pp. 42–54 (2018). https://doi.org/10.1007/978-3-030-17274-9_4
32. Osa, T., Pajarinen, J., Neumann, G., Bagnell, J.A., Abbeel, P., Peters, J.: An Algorithmic Perspective on Imitation Learning (2018)
33. Ross, S., Gordon, G., Bagnell, D.: A reduction of imitation learning and structured prediction to no-regret online learning. In: Gordon, G., Dunson, D., Dudk, M. (eds.) Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 15, pp. 627–635 (2011), http://proceedings.mlr.press/v15/ross11a.html
34. Salustowicz, R., Wiering, M., Schmidhuber, J.: Learning team strategies: Soccer case studies. Machine Learning **33**(2-3), 263–282 (1998). https://doi.org/10.1023/A:1007570708568
35. Sammut, C.: Behavioral cloning. In: Sammut, C., Webb, G.I. (eds.) Encyclopedia of Machine Learning, pp. 93–97 (2010). https://doi.org/0.1007/978-0-387-30164-8_69

36. Sarkar, S., Chakraborty, S.: Pitch actions that distinguish high scoring teams: Findings from five European football leagues in 2015-16 . Journal of Sports Analytics **4**(1), 1–14 (2018). https://doi.org/10.3233/JSA-16161
37. Schultze, S.R., Wellbrock, C.M.: A weighted plus/minus metric for individual soccer player performance. Journal of Sports Analytics **4**(2), 121–131 (2018). https://doi.org/10.3233/JSA-170225
38. Sha, L., Lucey, P., Yue, Y., Wei, X., Hobbs, J., Rohlf, C., Sridharan, S.: Interactive sports analytics: An intelligent interface for utilizing trajectories for interactive sports play retrieval and analytics. ACM Transactions on Computer-Human Interaction **25**(2), 13:1–13:32 (2018). https://doi.org/10.1145/3185596
39. Vercruyssen, V., Raedt, L.D., Davis, J.: Qualitative Spatial Reasoning for Soccer Pass Prediction. In: van Haaren, J., Kaytoue, M., Davis, J. (eds.) Proceedings of the 3rd Workshop on Machine Learning and Data Mining for Sports Analytics. CEUR Workshop Proceedings, vol. 1842 (2016)
40. Vroonen, R., Decroos, T., Haaren, J.V., Davis, J.: Predicting the Potential of Professional Soccer Players. In: Davis, J., Kaytoue, M., Zimmermann, A. (eds.) Proceedings of the 4th Workshop on Machine Learning and Data Mining for Sports Analytics. CEUR Workshop Proceedings, vol. 1971, pp. 1–10 (2017)
41. Yam, D.: A data driven goalkeeper evaluation framework. In: 13th MIT Sloan Sports Analytics Conference (2019)