Big Data
Volume 7, Number 1, 2019

DOI: 10.1089/big.2018.0054

ORIGINAL ARTICLE

Learning to Rate Player Positioning in Soccer

Uwe Dick* and Ulf Brefeld

Abstract

We investigate how to learn functions that rate game situations on a soccer pitch according to their potential to lead to successful attacks. We follow a purely data-driven approach using techniques from deep reinforcement learning to valuate multiplayer positionings based on positional data. Empirically, the predicted scores highly correlate with dangerousness of actual situations and show that rating of player positioning without expert knowledge is possible.

Keywords: deep learning; spatiotemporal data; reinforcement learning; scoring function

Introduction

Professional game scouts and soccer coaches assess video footage to analyze strengths and weaknesses of opposing teams. For instance, identifying successful attack patterns of an opposing team may assist coaches to quickly devise strategies to counter those patterns. Due to the low-scoring nature of the game, successful attack patterns do not only involve those that lead to actual goals.[1] Instead, it may be sufficient to find those patterns that lead to either clear goal scoring opportunities or, even more general, allow a team to enter a "danger zone," for example, the last 25 m of the pitch.[2]

From a conceptual point of view, the execution of a successful (good) attack pattern should raise the likelihood of scoring a goal. That is, the likelihood of scoring after executing the pattern should be higher than before executing the pattern. We follow this conceptual line and explore likelihoods of "being successful" from arbitrary game settings. We argue that, by being able to derive such valuations, we will be able to find good (and bad) attacking patterns by measuring differences in likelihoods.

In this article, we investigate approaches to valuate player positioning on a soccer pitch. We aim at assessing a game setting, including all player and ball positions and current movement vectors at a given time, with regard to its potential of leading to a successful attack for the team that has possession of the ball. We thus focus on tracking data of a set of European topflight soccer matches.[3,4] The recorded data consist of x/y coordinates of all players of both teams and the ball, measured at 25 frames per second. An additional flag indicates which team is in possession of the ball at each time point and if the ball is in play or if the game is halted, for example, due to a foul.

We extract sequences of open play where one team retains ball possession without either losing the ball or the play being stopped. Each such sequence ends with either one of these events or with that team performing a "success action." Following the discussion above, a "success action" can, for example, consist of the team entering the final 25 m of the pitch. In case of a successful action, we label the sequence as positive, otherwise as negative.

Using these sequences, our approach is to learn a scoring function that maps game situations to real numbers using ideas and methods from deep reinforcement learning (RL). RL has gained a lot of attention over the past years. Particularly, combining RL with "deep models" often leads to impressive performances on some very hard (and often game-related) learning problems.[5–7]

Our contribution is as follows: we (1) model soccer matches as Markov processes of game settings that include positional data and movement vectors of all players and the ball. In this model, these *state* sequences are created by actions of the two teams who take the role of the controller or policy. We (2) propose a convolutional

Institute of Information Systems, Leuphana University, Lüneburg, Germany.

*Address correspondence to: *Uwe Dick, Leuphana University, Universitätsallee 1, Lüneburg 21335, Germany,* E-mail: uwe.dick@leuphana.de

neural network to learn the value function of this policy. The value function thus rates game situations according to *how good* they are for a team. (3) Empirical results on positional data from topflight soccer matches show the effectiveness of the proposed approach.

Related Work

We pursue a purely data-driven approach by learning valuations from spatiotemporal data from real topflight soccer matches. To that end, we do not make use of any expert knowledge on identifying dangerous situations or attacking patterns. Instead, we use only raw data and infer dangerous situations and "good" attacking patterns using that data alone.

Our approach is somewhat diametral to the one taken by Link et al.,[1] who also presented a measure of dangerousity of game situations. They defined dangerousity as a function of four aspects, such as the area where a player is in possession of the ball or the pressure put on a player by the opposing team, respectively. Those metrics were evaluated using positional data of real soccer matches. However, the parameters of the model were adjusted manually based on expert knowledge. The authors derived some performance indicator metrics from dangerousity, such as *action value* that is defined as the difference in dangerousity between ball possession of two different players. As a limitation, the article defined dangerousity only inside an area starting 34 m from the opponents' goal. They also computed a shot density estimate that calculates some likelihood of a successful shot from a certain position. Lucey et al.[4] presented a model for shot success prediction using handcrafted features from spatiotemporal data. A similar problem for basketball was studied by Cervone et al.,[8] who proposed hierarchical statistical models to predict the expected number of points after ball possession.

Copete et al.[9] predicted player and ball movements in RoboCup 2D Soccer Simulation League games based on two-dimensional (2D) trajectory data. They employed two different models for ball movement and player movements. For ball movement, they used a feed-forward deep neural network model that takes as input the current positions and outputs the next position of the ball. Player movements were predicted using a recursive deep autoencoder architecture. Bialkowski et al.[3] learned the roles of players based on spatiotemporal data that describe their spatial arrangement on the pitch. They used a minimum entropy model to partition the data into player roles. They showed that distinct formation classes, such as a 4-1-4-1 formation, can be discovered automatically using their method. Spatiotemporal soccer data were used by Fernando et al.[10] to group and compare scoring approaches of teams. Van Haaren et al.[11] used event and positional data to discover relevant ball possession phases. They manually assigned weights to certain events such as shot or cross and scored clustered ball possession phases according to their summed weights.

A Bayesian topic model was proposed by Wang et al.[12] to learn and infer tactical patterns from event data, that is, data about which player passed the ball when and where to another player. The authors reported on automatically identified tactical patterns, that is, passing sequences in certain regions of the pitch for teams. They also tracked the usage of attacking patterns over the course of a game. Knauf et al.[2] also aimed to identify tactical patterns. Using positional data, they devised convolutional kernels to capture multiple trajectories in space and time and reported about game initiation and scoring opportunity patterns. If, in contrast to patterns for predefined situations, all frequent patterns in multiple trajectory data are of interest, the approach taken by Haase and Brefeld[13] can be applied.

Van Haaren et al.[14] proposed an inductive logic programming approach to learn pass sequences that capture attack regularities. Game event data were also exploited by Lucey et al.[15] to visualize plays that start from specific areas of the pitch as well as by Brandt and Brefeld[16] who deployed page rank algorithms on pass sequences to capture team interaction. Rein and Memmert[17] provided a good overview over related approaches in the context of soccer tactics.

Contribution

This section details the contributions of our work. We will first specify the kind of data we will be using to learn ratings of players but postpone details on statistics of the actual data set that is used in our experiments to the Experimental Setup section. In the Learning section, we describe the model and the learning procedure.

Preliminaries and data specification

In our analyses, we use tracking data consisting of a sequence of x/y coordinates of all players and the ball for a set of soccer games, sampled at 25 frames per second. We additionally make use of indicator variables that state, at every point in time, which team is in possession

of the ball and whether the ball is in play or if the game is halted, respectively.*

We learn to valuate game settings with regard to their potential of staging a successful attack. We define a game setting as the positions and moving directions of all players and the ball and also include the information, which team has possession of the ball. That is, a state is one frame of the input data. As a preprocessing step, we extract sequences or *episodes* of open play where one team retains ball possession without either losing the ball or the play being stopped due to fouls or the ball being out of the playing field. Each such ball possession phase ends with the team controlling the ball by either losing the ball or the play being stopped, or with that team performing a "success action."

In the experiments, we consider two different versions of success actions. The first scenario declares a ball possession phase of a team as a success if that team carries the ball into the final 25 m of the opponent's half. The second scenario defines an episode as a success if the team has ball possession inside the area 18 m around the opponent's goal (2 m outside the penalty box). Note, however, that other definitions of success are easily incorporated, such as shots on target or goals.[1]

We aim to learn a function that assigns scores to situations such that situations within a successful episode should have higher scores than those of unsuccessful ones. The function thus imposes a ranking of the situations and the optimal function ranks all successful situations higher than unsuccessful ones. A natural performance metric for such binary ranking problems is the area under the ROC curve (AUC).[18]

Learning

We learn valuations of game settings by making use of recent developments in deep RL. Deep RL approaches have shown impressive performance in a variety of domains recently.[5–7] However, most of these approaches aim to learn an optimal controller based on experience data that are sampled from interactions of the learned controller with the environment. This is clearly impossible for learning from real soccer matches as we cannot alter player and ball movements realistically. The current problem therefore constitutes a *batch* RL problem. In batch RL problems, valuations and policies have to be learned from a fixed set of interactions and rewards that is sampled before the learning process.

*A game is for instance halted by a foul.

Note that we are not focusing on learning optimal controllers as this translates to learning how players would behave optimally on the pitch, which is not in the scope of this article. Instead, we are interested in *how good* real players behave on the pitch and how likely the behavior results in a success.

RL is a learning paradigm for learning controllers or policies that perform optimally in an environment that can be modeled as a Markov decision process. However, as discussed above, the aim of this article is to learn valuations instead of controllers. Such cases can be modeled as Markov reward processes (MRP), for example, van Seijen et al.[19] An MRP consists of a state space S, a transition function $p(s, s') \in [0, 1]$ that describes the likelihood of transitioning from state $s \in S$ to state $s' \in S$, and a reward function $R(s, s') \in \mathbb{R}$ that determines the expected immediate reward for transitioning from s to s'. A discount factor $\gamma \in [0, 1)$ determines the relative influence of future rewards over the immediate reward.

For the task at-hand, sequences of states are separated into episodes $e = (s_1, s_2, \ldots, s_T)$. Episodes have episode-dependent start and end states and length or number of time steps $_T$ per episode, and at the end of every episode, there is a positive reward of 1 if, and only if, the episode is successful and 0 otherwise. The *return* G is defined as the cumulative discounted reward that is observed after starting in state s_t until the end of an episode

$$G(s_t) = \sum_{t'=t}^{T-1} \gamma^{t'-t} R(s_{t'}, s_{t'+1}). \quad (1)$$

We aim at finding the *value function* V, which maps each state s_t to its expected return over all possible state transitions—and therefore all possible episodes that contain s_t

$$V(s_t) = E_{p,T}[G(s_t)]. \quad (2)$$

The remainder of this section is structured as follows. We will first present the deep convolutional model that takes a representation of a game situation—or state—as input and outputs a rating—or value function $_V$—of that state. A batch RL algorithm is presented next, and in a last step, we show the exact learning procedure.

Deep model. The great resurgence of RL methods comes from its impressive performance when used with deep learning models as state, state-action, and/

or policy representation.[5–7] We follow the same line and model states—which in our case are game settings as described in the Preliminaries and Data Specification section—using deep convolutional networks. To this end, game settings are represented as "2D images" with nine channels, that is, a three-dimensional tensor with dimensions (image width, image height, 9). The first channel has a value of one at each pixel where a player of the team playing from left to right is located. The second channel represents the team playing from right to left, and the third channel encodes the position of the ball. Channels 4 and 5 encode partial speeds in x and y directions of players of the team playing from left to right. That is, if player A of that team is located at pixel (p_x, p_y) and has a current movement vector of (s_x, s_y), then the location (p_x, p_y) has the value s_x in channel 4 and s_y in channel 5. The movements of the other team are encoded in channels 6 and 7, the ball movement in channels 8 and 9. All other values are set to 0. We additionally input which team has possession of the ball as a single input, which is either +1 or −1. In our experiments, we also use a second input representation to investigate the influence of movement vectors on the predictive performance of the proposed model. This representation only uses the first three channels of the "input image," that is, the positions of both teams and the ball.

Figure 1 shows the general architecture of the deep model. The image-like input is fed into a three-layer convolutional network with each layer having 32 kernels of size 6×6 and ReLu activation units.[20] The last convolutional layer is followed by one fully connected layer with 256 nodes and ReLu activation units. The output of the deep model consists of one node representing the predicted value $_V$ of the input game setting.

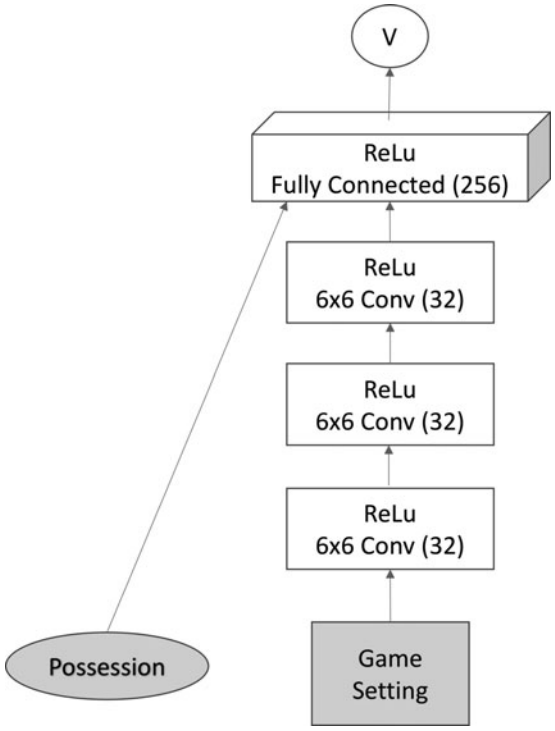


FIG. 1. Visualization of the state model.

Batch RL. Several methods for estimating value functions from data were investigated and used in the literature over the years.[19,21–24] Temporal difference (TD) methods are among the most popular of those; they minimize the difference of values between states and future states to learn the value function. For convenience, in the remainder of this section, we present the λ-return algorithm,[25] which can be considered the forward view of the $TD(\lambda)$ algorithm[19] that we use—in combination with the deep model described above—in the experimentations.

The estimated *n-step return* on a sample episode $e=(s_1, s_2, \ldots, s_T)$ of state s_t, $1 \leq t \leq T$, based on an estimate $\hat{V}$ of the value function is defined for $n < T-t$ as

$$G^n_{\hat{V}}(s_t) = \sum_{t'=1}^{n} \gamma^{t'-1} R(s_{t+t'-1}, s_{t+t'}) + \gamma^n \hat{V}(s_{t+n}) \qquad (3)$$

and for $n \geq T-t$, it is defined as the real return $G(s_t)$ in Eqn. 1. The λ-return algorithm updates estimates of the value function by computing the λ-return

$$G^{\lambda}_{\hat{V}}(s_t) = (1-\lambda) \sum_{n=1}^{T-t-1} \lambda^{n-1} G^n_{\hat{V}}(s_t) + \sum_{m=T-t}^{\infty} \lambda^m G^{T-t}_{\hat{V}}(s_t), \qquad (4)$$

where the last term $G^{T-t}_{\hat{V}}(s_t)$ is the real return as defined above. The infinite sum $\sum_{m=T-t}^{\infty} \lambda^m$ accounts for the intuition that we *should not look beyond the end of the episode*. Note that this definition is slightly different from the one used, for example, in the study of van Seijen et al.[19]

Our model learns $\hat{V}$ by minimizing the mean squared TD error over the training data set $D = e^1, \ldots, e^m$

$$\min \sum_{e \in D} \sum_{i=1}^{T^e} \left(\Delta \hat{V}(s_i^e)\right)^2 \qquad (5)$$

with TD error

$$\Delta \hat{V}(s) = \hat{V}(s) - G^{\lambda}_{\hat{V}}(s). \qquad (6)$$

Learning. The learning procedure works as follows. For an episode consisting of states $e=(s_1, s_2, \ldots, s_T)$, we first compute a forward pass of the net for each of the states $s_2, \ldots, s_T$ to compute value function values $\hat{V}(s_2), \ldots, \hat{V}(s_T)$. We then compute λ-returns for all states $G^{\lambda}_{\hat{V}}(s_1), G^{\lambda}_{\hat{V}}(s_2), \ldots, G^{\lambda}_{\hat{V}}(s_T)$. Note that we assume that the last state of an episode s_T is followed by a virtual state s_E that has a value $\hat{V}(s_E)=0$. Thus, to stay in sync with notation in the Batch RL section, episodes now have a virtual length of $T+1$ with $s_{T+1}=s_E$. Only transitions from s_T to s_E may yield rewards of 1 if the episode ends in a success. $R(s_t, s_{t+1})=0$ in all other cases. Also note that it follows that for state s_T, λ-returns consist of either 0 if the episode does not end in a success or 1 otherwise (c.f. Eqn. 4). As a last step, we use these λ-returns as targets for the squared loss of Eqn. 5 and perform back propagation to compute gradients and update all parameters θ of the deep model. Figure 2 shows a visualization for the computation of $G^{\lambda}_{\hat{V}}(s_t)$.

This approach is similar to that of Harb and Precup,[26] who used a recurrent NN to learn a deep recurrent Q-network (DRQN). Hausknecht and Stone[27] also leveraged DRQNs to learn Atari games, whereas Foerster et al.[28] used DRQN-like networks in a multiagent learning setting.

Empirical Results

Experimental setup

The data used in our experiments consist of five games of topflight European soccer. Preprocessing of data is described in the Preliminaries and Data Specification section. As mentioned, we consider two different versions of success actions, one that declares an episode a success if the team carries the ball into the final 25 m of the opponent's half. The second one considers ball possession inside the area 18 m around the opponent's goal a success. The final data sets consist of 380 successful episodes and 715 unsuccessful episodes with an overall number of 11,045 states for the first success measure and 224 successful episodes and 866 unsuccessful episodes with 11,340 states for the second success measure. During training, all frames of an episode are randomly flipped in x and/or y direction to boost training sample size. The ball possession flag is flipped accordingly. Boosting virtual training size is common practice in (deep) machine learning, for example, He et al.[29] for image recognition.

To evaluate how well the learned value functions reflect the actual likelihood of success, we compare predictions to actual outcomes of ball possession sequences. To that end, we compute the value function at certain points in time for a ball possession phase,

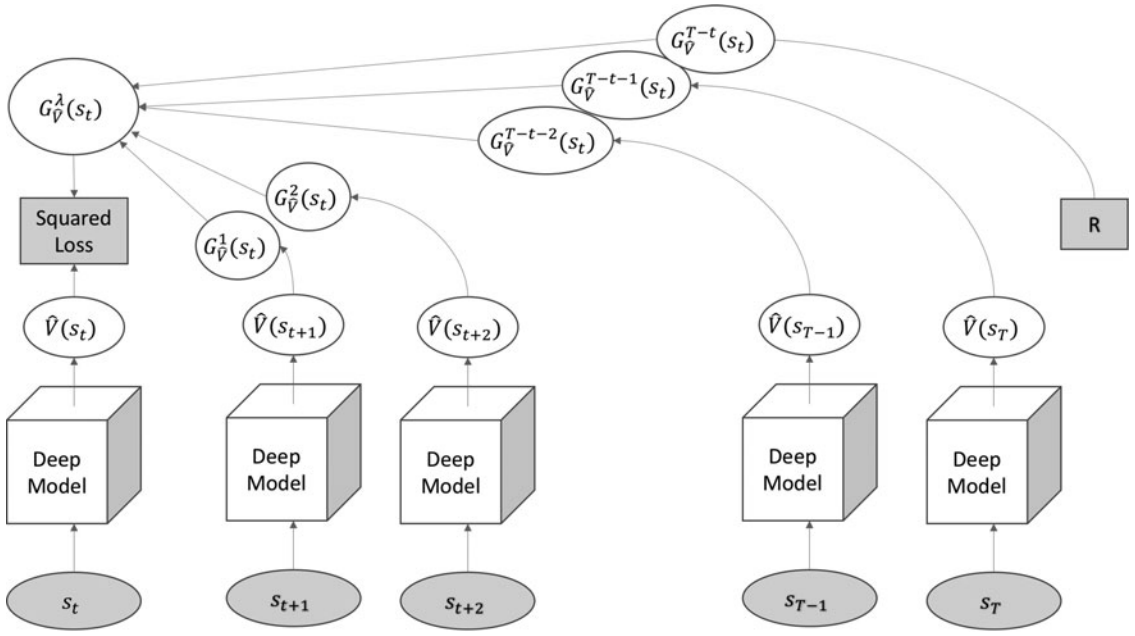


FIG. 2. Visualization of computation of $G^{\lambda}_{\hat{V}}(s_t)$ for an episode of length T.

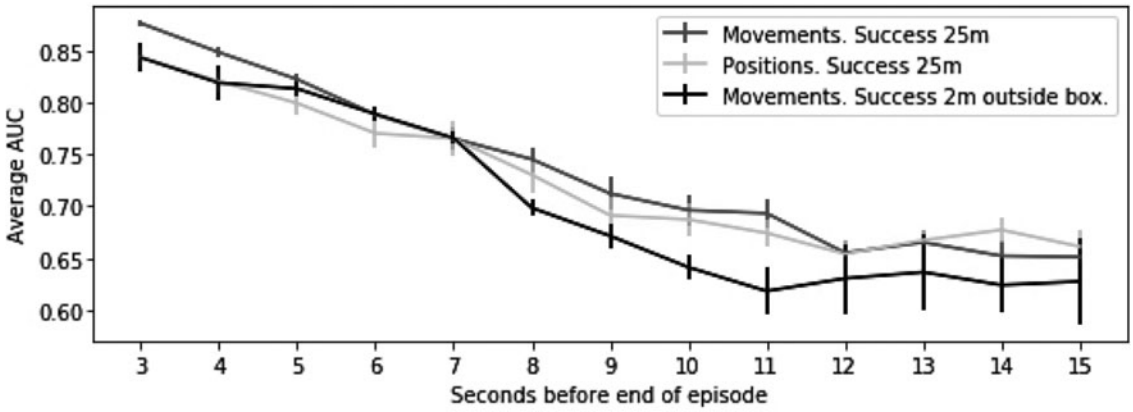


FIG. 3. Average AUC and standard errors for differing time points before the end of an episode as computed in LOGO setting. We compare the model using movement information with the one that uses positions only. The *x*-axis determines seconds before the end of an episode. AUC, area under the ROC curve; LOGO, leave-one-game-out.

for example, 5 seconds before the end of the phase, and evaluate whether the possession phase ends in a success. This renders the evaluation a binary ranking problem and we use the AUC[18] to quantify the quality of the scoring functions.

As a second measure, we evaluate valuations at certain regions of the pitch. We compute the value function when the ball is at a certain distance from the opponents' goal line and again compare to actual success rates by means of AUC.

Training procedure

We implemented the model of the Learning section using the TensorFlow library.[†] We set the batch size to 30 episodes and train the model using the *Adam* optimization algorithm.[30] We set discount factor $\gamma = 0.95$ as defined in the Learning section and use $\lambda = 0.7$ for the λ-return algorithm in Eqn. 4 due to a grid-search. We employ an early stopping criterion by using a *leave-one-game-out* (LOGO) cross-validation (see the Results section) on the *training* set and choosing the best number of training batches for training on the complete training set.

Results

Our experiment aims to shed light on how well the learned value functions transfer between games and teams.

[†]https://www.tensorflow.org/

Quantitative evaluation. We implement a "LOGO" cross-validation where training is performed on four of the five games and evaluation metrics are computed on the remaining game. We test both input representations as described in the Learning section, namely one that uses only the positional data of all players and the ball—which we name *Positions* in the figures—and one that also encodes the current movement vectors (*Movements*).

Time-dependent AUC. We vary the time of evaluating game states before the end of a ball possession phase between 3 and 15 seconds. Figure 3 shows the averaged AUCs on the test data over all five games for both input representations and both success definitions. As expected, the AUC becomes smaller the earlier in a possession phase we score the game state. However, even 13 seconds before the end of a phase, the AUC is still strictly above 0.5, which would indicate uninformed guessing. Considering the inherently random nature of soccer, the AUC scores show that the learned value functions indeed capture important aspects of player and ball positioning and movement vectors. We can also see that the results are roughly comparable for both considered success measures.

Area-dependent AUC. The AUCs for scores taken in specific areas on the pitch are shown in Figure 4. Our approach is able to learn value functions of player positioning for all areas of the pitch. Not surprisingly, the closer the ball possessing team is to the opponent's goal,

the higher the average AUC. The closer a team gets to the dangerous zone, the less chance influences entering into it; often, a single pass or straight run does the job. The left figure also shows that using only positions without movement information performs worse than using all available information. A similar tendency can also be observed in Figure 3; however, the differences are not significant. The right figure shows that AUC results for the second success measure (18 m around the goal) are comparable, although not as good as the ones for the first success measure (25 m in front of the goal). This is likely due to a reduced number of positive training episodes (see the Experimental Setup section) and due to an increased influence of chance on the outcomes of episodes (see the Visualization of Outcomes section).

Note that a trivial baseline would simply be the ball's distance from the goal as a measure of dangerousity. Figure 4 shows that our approach easily beats this baseline because this heuristic would be a diagonal line and yield an AUC of 0.5.

Visualization. To get an idea of situations rated with high scores, we show eight such settings in Figure 5. They all show settings from of one game where the red team has ball possession 5 m into the opponent's half, playing from left to right. Figures are sorted according to decreasing scores. Note that these valuations were used for computing the corresponding AUC in Figure 5. We can see that in most of the high-scoring valuations (2–7 in Fig. 5), the ball is played to a relatively open player on the wing. In several of the situations, the game is in a "dynamic" state (2, 6, 7, and 8 in Fig. 5) where defensive players of the defending team and offensive players of the attacking team are running with high pace toward the goal. Only two situations are more "classical" built-up plays where most of the defenders are positioned behind the ball (3 and 5 in Fig. 5).

Figure 6 shows six lowest scoring situations, also 5 m into the opponent's half. We can see that all six game situations contain a pass that is played to a white player that is tightly covered and does not seem to have any promising passing options into the forward direction. In fact, all the depicted episodes end with a failure shortly after the shown game situations.

Visualization of outcomes. To get a better idea of the influence of player decisions and chance on soccer results, Figure 7 shows three high-scoring situations 10 m into the opponent's half on the left and corresponding outcomes of the episodes on the right (success measure 18 m around the goal). The first row shows an unsuccessful episode where the ball possessing player is pushed to the corner, nicely reflected by a decrease in the valuation, and in fact looses the ball for a goal kick. The second episode yields a success because the white player crosses the ball to his teammate

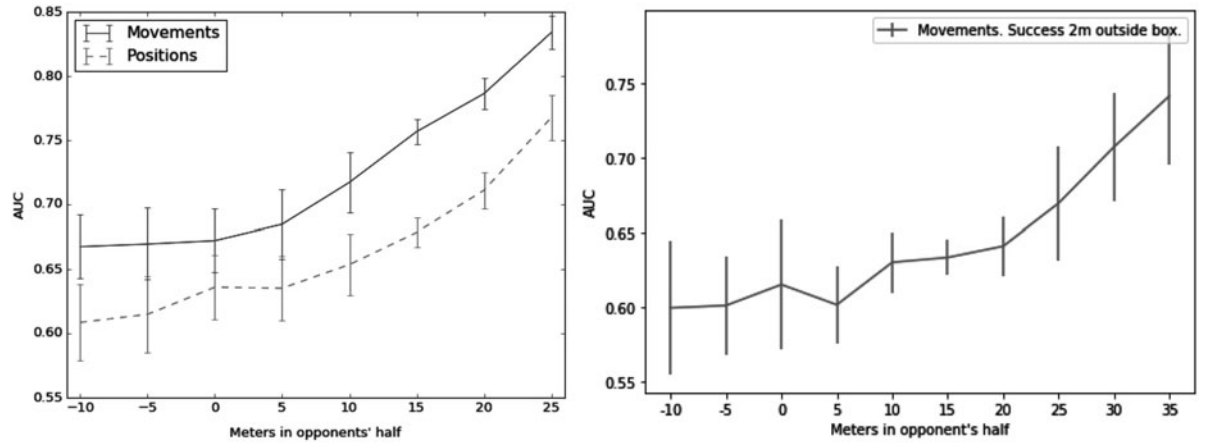


FIG. 4. Average AUC and standard errors for different valuation areas as computed in LOGO setting. The *x*-axis determines how deep into the opponents' half the valuations are computed. Left: Scenario with success definition of 25 m in opponent's half. We compare the model using movement information with the one that uses positions only. Right: Ball control no more than 2 m outside opponent's box is considered success.

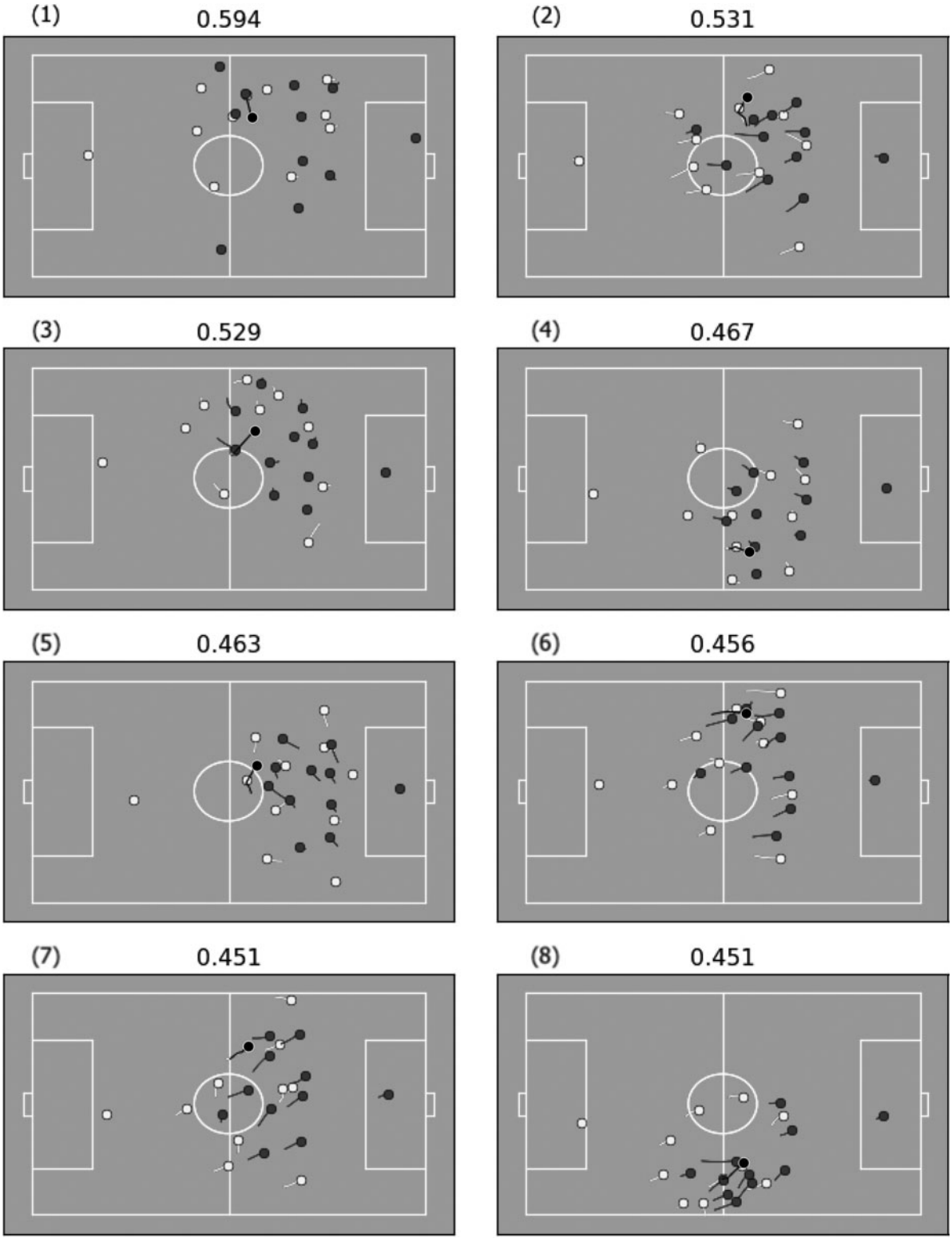


FIG. 5. Game situations with white team playing from left to right and having ball possession 5 m into the opponent's half. Figures are sorted and numbered according to valuation, which is placed above each subfigure.

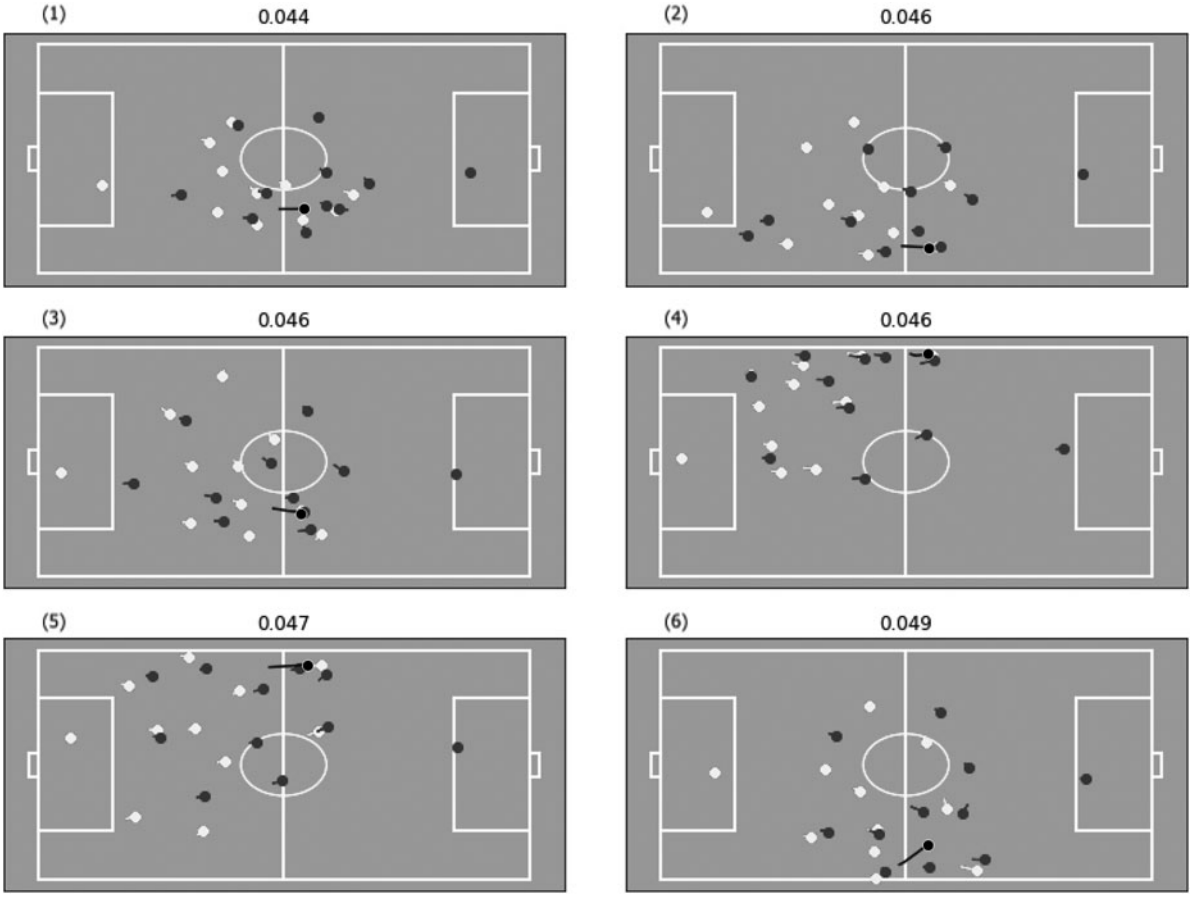


FIG. 6. Game situations with white team playing from left to right and having ball possession 5 m into the opponent's half. Figures are sorted in increasing order according to valuation, which is placed above each subfigure.

successfully, and the last episode is unsuccessful because the ball is actually headed away by the dark gray defender. Nevertheless, valuations for the last two episodes are very similar because both end with high balls into the penalty box, but only one is successful.

Visualization of score development. A frequent quantity of interest is to identify "surprising" elements in a game. In the context of the presented investigation, "surprising" patterns correspond to those that change the valuation of positional settings dramatically and in a favorable way. Based on the learned valuations, we compare valuations taken at different timestamps during a ball possession phase and pick only situations realizing the largest TDs. Figure 8 depicts three patterns from the white team that took 4 seconds each and where the valuations change the most. The white team plays again from left to right. The middle and right panels of Figure 8 show long diagonal balls that are played to an open player on the right wing. In both situations, the player, if able to control the ball, gets possession in or very close to the dangerous zone. The left panel of Figure 8 shows a white player who gets possession of the ball close to the last line of defense with another open player free on the wing. The play starts in the own half and consists of two passes through the center. Appendix contains a more fine-grained visualization of temporal changes of valuations during a ball possession phase.

Conclusion

We proposed a deep RL approach to learn valuations of multiplayer positionings using positional data. The purely data-driven approach did not rely on any

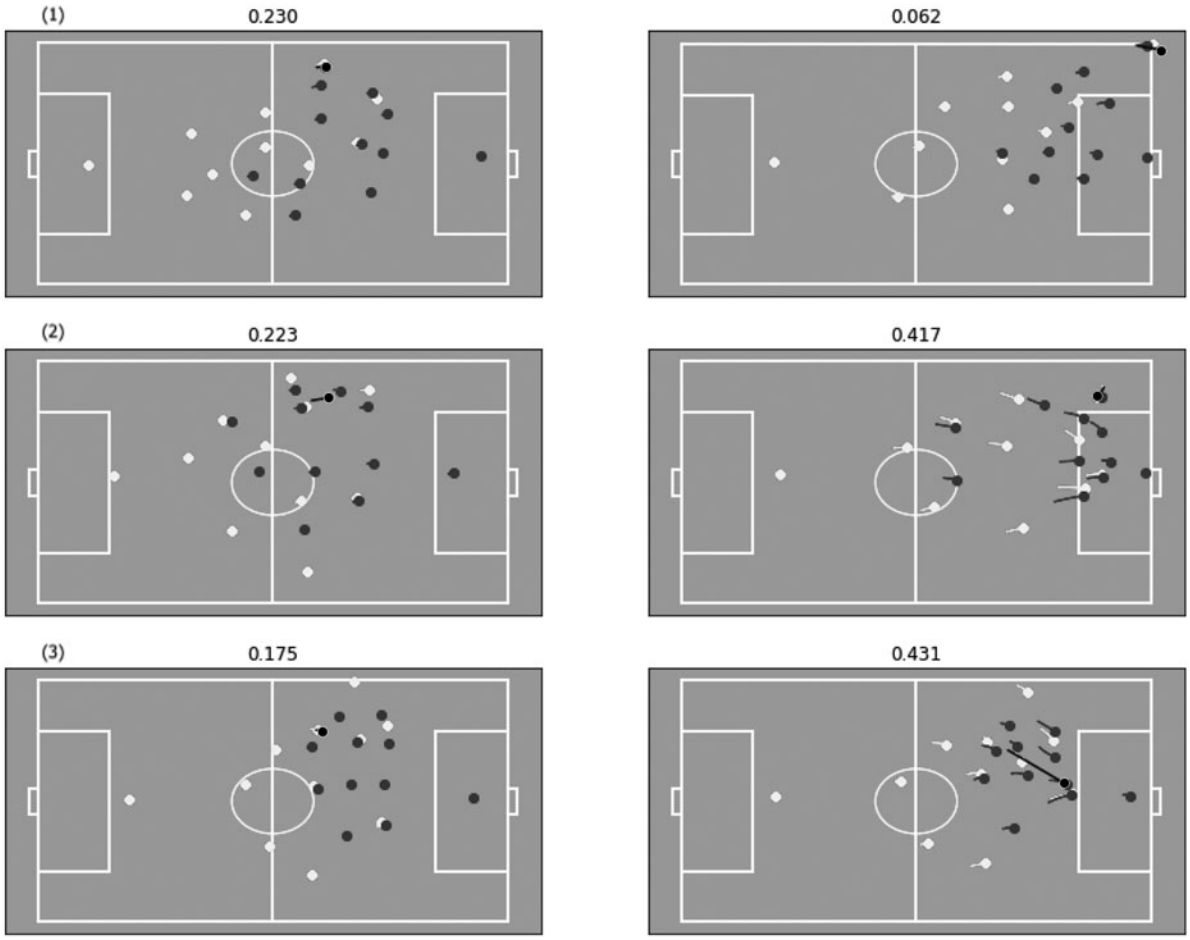


FIG. 7. Three snapshots (numbered 1 to 3) of ball possession phases of white team playing from left to right. Left: Snapshots when ball is located 10 m into the opponent's half. Right: End of the respective ball possession phase.

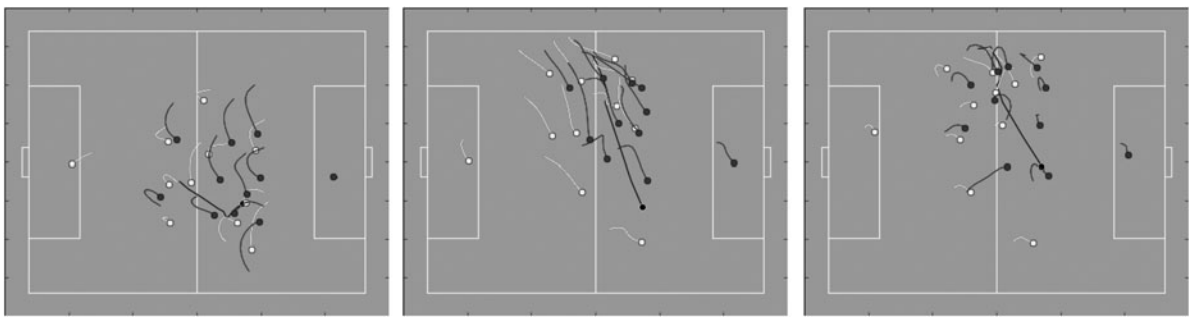


FIG. 8. Largest changes in valuation over a 4 second span. White team plays from left to right.

prior knowledge of the domain and successfully learnt valuations, which can often be interpreted in a meaningful way. To the best of our knowledge, this work constitutes the first purely data-driven approach to machines that read and understand games and, thus, bridges the gap toward computational tactics.

For instance, correlations between our dangerousness metric and traditional performance indicators like playing speed, passing, or expansion of teams could be used to group historic episodes against an opponent. These groups could then be used to automatically devise strategic insights for this opponent, for example, indicating that counterattacks were more promising than slow playmaking or that cross passes led to more dangerous situations. These insights could then be implemented in the respective game plan.

Acknowledgment

The authors thank Hendrik Weber and Deutsche Fußball Liga (DFL)/Sportcast GmbH for providing the positional data.

Author Disclosure Statement

No competing financial interests exist.

References

1. Link D, Lang S, Seidenschwarz P. Real time quantification of dangerousity in football using spatiotemporal tracking data. PLoS One. 2016;11:e0168768.
2. Knauf K, Memmert D, Brefeld U. Spatio-temporal convolution kernels. Mach Learn J. 2016;102:247–273.
3. Bialkowski A, Lucey P, Carr P, et al. Large-scale analysis of soccer matches using spatiotemporal tracking data. In: 2014 IEEE International Conference on Data Mining, Shenzhen, 2014, 725–730. doi:10.1109/ICDM.2014.133
4. Lucey P, Bialkowski A, Monfort M, et al. Quality vs quantity: Improved shot prediction in soccer using strategic features from spatiotemporal data. In: Proc. 8th Annual MIT Sloan Sports Analytics Conference. Boston, MA, 2014, pp. 1–9.
5. Mnih V, Badia AP, Mirza M, et al. Asynchronous methods for deep reinforcement learning. In: Proceedings of Machine Learning Research (PMLR), New York, 2016, pp. 1928–1937.
6. Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning. Nature. 2015;518:529–533.
7. Silver D, Huang A, Maddison CJ, et al. Mastering the game of go with deep neural networks and tree search. Nature. 2016;529:484–489.
8. Cervone D, DAmour A, Bornn L, Goldsberry K. A multiresolution stochastic process model for predicting basketball possession outcomes. J Am Stat Assoc. 2016;111:585–599.
9. Copete JL, Suzuki J, Wei Q, et al. Estimation of players actions in soccer matches based on deep autoencoder. Japanese Society for Artificial Intelligence, Technical Report, 2015, p. 7.
10. Fernando T, Wei X, Fookes C, et al. Discovering methods of scoring in soccer using tracking data. Sidney, Large-Scale Sports Analytics, 2015.
11. Van Haaren J, Hannosset S, Davis J. Strategy discovery in professional soccer match data. In: Proceedings of the KDD-16 Workshop on Large-Scale Sports Analytics, New York: ACM, 2016.
12. Wang Q, Zhu H, Hu W, et al. Discerning tactical patterns for professional soccer teams: An enhanced topic model with applications. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York: ACM, 2015. pp. 2197–2206.
13. Haase J, Brefeld U. Mining positional data stream. In: Appice A, Ceci M, Loglisci C, et al. (Eds.): New Frontiers in Mining Complex Patterns, Cham, Switzerland: Springer, 2015, pp 102–116.
14. Van Haaren J, Dzyuba V, Hannosset S, Davis J. Automatically discovering offensive patterns in soccer match data. In: International Symposium on Intelligent Data Analysis, Cham, Switzerland: Springer, 2015, pp. 286–297.
15. Lucey P, Oliver D, Carr P, et al. Assessing team strategy using spatiotemporal data. In: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, New York: ACM, 2013, pp. 1366–1374.
16. Brandt M, Brefeld U. Graph-based approaches for analyzing team interaction on the example of soccer. In: Proceedings of the ECML/PKDD Workshop on Machine Learning and Data Mining for Sports Analytics, Cham, Switzerland: Springer, 2015.
17. Rein R, Memmert D. Big data and tactical analysis in elite soccer: Future challenges and opportunities for sports science. SpringerPlus. 2016;5:1410.
18. Bradley AE. The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognit. 1997;30:1145–1159.
19. van Seijen H, Sutton RS. True online TD (lambda). In: Proceedings of Machine Learning Research (PMLR), Beijing, China, 2014, pp. 692–700.
20. Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks. In: Proceedings of Machine Learning Research (PMLR), Sardinia, Italy, 2011.
21. Bertsekas DP. Approximate policy iteration: A survey and some new methods. J Control Theory Appl. 2011;9:310–335.
22. Kocsis L, Szepesvári C. Bandit based monte-carlo planning. In: European conference on machine learning, Cham, Switzerland: Springer, 2006, pp. 282–293.
23. Lagoudakis MG, Parr R. Least-squares policy iteration. J Mach Learn Res. 2003;4:1107–1149.
24. Tesauro G. Temporal difference learning and TD-gammon. Commun ACM. 1995;38:58–68.
25. Sutton RS, Barto AG, et al. Reinforcement learning: An introduction. Cambridge, MA: MIT Press, 1998.
26. Harb J, Precup D. Investigating recurrence and eligibility traces in deep Q-networks. NIPS 2016, Deep Reinforcement Learning Workshop, 2016.
27. Hausknecht M, Stone P. Deep recurrent q-learning for partially observable MDPs. In: 2015 AAAI Fall Symposium Series, Palo Alto, CA: AAAI PRESS, 2015.
28. Foerster J, Assael YM, de Freitas N, Whiteson S. Learning to communicate with deep multi-agent reinforcement learning. In: Advances in Neural Information Processing Systems, Barcelona, Spain: NIPS, 2016, pp. 2137–2145.
29. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
30. Kingma D, Ba J. Adam: A method for stochastic optimization. In: Proceedings of the 3rd International Conference on Learning Representations (ICLR), 2015. arXiv preprint. 2014;arXiv:1412.6980.

Cite this article as: Dick U, Brefeld U (2019) Learning to rate player positioning in soccer. *Big Data* 7:1, 71–82, DOI: 10.1089/big.2018.0054.

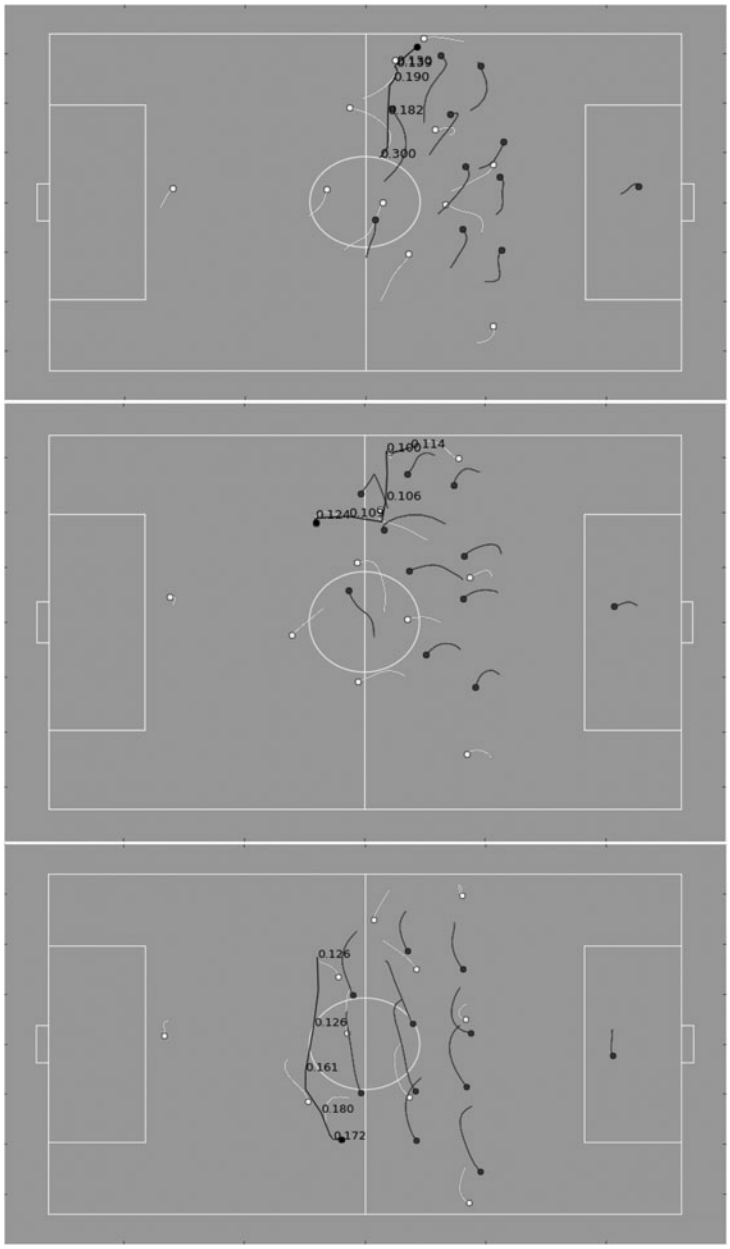
Abbreviations Used

AUC = area under the ROC curve
LOGO = leave-one-game-out
MRP = Markov reward processes
RL = reinforcement learning
TD = temporal difference

(*Appendix follows*→)

Appendix

To help getting a better understanding of how valuations change during a ball possession phase, in Appendix Figure A1, we show three pictures that depict valuations over the span of 15 seconds. Scores are shown close to where the ball is once every second of the possession phase. Again, valuations are generally smaller the further away the red team is from the danger zone. However, other factors also influence the valuation, as can be seen, for example, by looking at the beginning of the phase (Appendix Fig. A1, top), where the play starts with a valuation of 0.300 but rapidly declines while the ball is played to the left wing. From there the score gets smaller still until it is played to a defender. However, in Appendix Figure A1 (bottom), the score gets larger again despite the ball being roughly the same distance from the goal line. The difference is that in the latter stages, the ball carrying player is not as closely marked and has options to play the ball wide while he was closely marked in the earlier stages.



Appendix Fig. A1 Figures show temporal differences during 15 seconds of a ball possession phase where the white team playing from left to right has the ball: initial 5 seconds of the phase (top), followed by next 5 seconds (center), and final 5 seconds (bottom). Valuations are printed once every second close to the respective position of the ball.