Extracting Semantic Information from Dynamic Graphs of Geometric Data

Devavrat Vivek Dabke(✉) and Bernard Chazelle

Princeton University, Princeton, NJ 08544, USA
ddabke@princeton.edu

Abstract. In this paper, we demonstrate the utility of dynamic network sequences to provide insight into geometric data; moreover, we construct a natural syntactic and semantic understanding of these network sequences for useful downstream applications. As a proof-of-concept, we study the trajectory data of basketball players and construct "interaction networks" to express an essential game mechanic: the ability for the offensive team to pass the ball to each other. These networks give rise to a library of player configurations that can in turn be modeled by a jump Markov model. This model provides a highly compressed representation of a game, while capturing important latent structures. By leveraging this structure, we use a Transformer to predict trajectories with increased accuracy.

Keywords: Geometric data · Networks · Dynamic networks · Machine learning

1 Introduction

Multi-agent systems are fascinating both for their geometric properties and for their complex interactions. In a variety of contexts, we would like to understand their underlying dynamics, moving beyond the construction of black-box models that simply replicate their behavior. Thus we strive to produce a model that leverages dynamic networks (i.e. networks with a topology that changes over time) to encode a geometric system. We can then extract "semantic" information about these network sequences to crystallize our understanding of the underlying dynamics. One paragon example of a multi-agent geometric system includes fast-paced "invasion" sports like basketball, soccer, and hockey [8].

We study the trajectories of players as they move across the court. To develop a rich understanding of the dynamics of basketball players, we develop a model with:

1. Formation discovery: a semantic understanding of the functional roles of players;
2. High compression: an efficient representation of a game, as player trajectory data is large and difficult to interpret;
3. Predictive power: a mechanism for generating synthetic basketball data and predicting trajectories of players.

R. M. Benito et al. (Eds.): COMPLEX NETWORKS 2021, SCI 1016, pp. 474–485, 2022.
https://doi.org/10.1007/978-3-030-93413-2_40

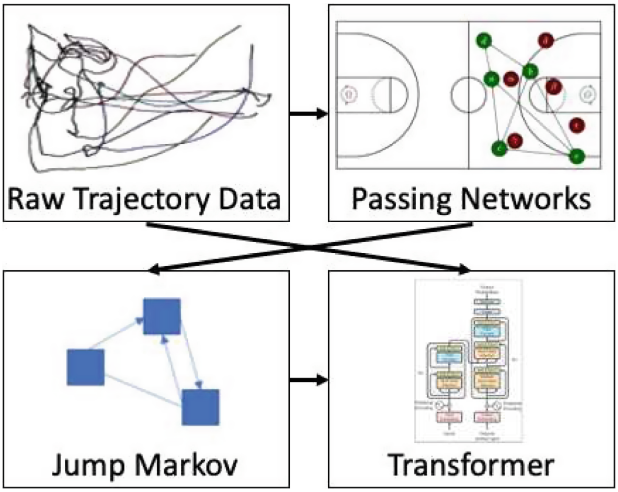

Fig. 1. Overview of the data analysis pipeline. First, the raw trajectory data is converted into interaction networks. Second, by comparing graphs up to isomorphism, we can construct a "library" of possible configurations. We can then construct a jump Markov model by taking the empirical maximum likelihood estimator with graphs as the state space. Finally, we can feed in the raw trajectory data and the graph data from the jump Markov model into a Transformer model for prediction. This article opens the door to future work on inferring game semantics and strategies from actual games.

In summary, the synthesis of semantic and geometric data is the main contribution of this paper, which is a theoretical innovation, as well as a practical one: on an important downstream task (trajectory prediction), we greatly improve on past results. We propose a novel pipeline to convert geometric data from a complex, interactive multi-agent system into semantic sequences. This new perspective provides better insight into the underlying dynamics, as well as stronger results in important applications like trajectory prediction.

2 Related Work

The analysis of basketball player trajectory began shortly after cutting-edge technology was developed that allowed for comprehensive player tracking [11]. Early models did reasonably well in role discovery and compression. Over the past decade, however, predictive power has greatly increased, especially for the most common prediction task, which is trajectory prediction.

2.1 Trajectory Prediction

Trajectory prediction focuses on forecasting the movement of players given their history. Generically, trajectory prediction is a much broader, and relatively old, discipline, but, for concreteness, we restrict ourselves to those that involve sports [13, 15].

Generally, state-of-the-art techniques involve black-box machine learning models that predict K frames of data from L initial frames [7,10,18]. Some models explicitly construct roles for players, while others avoid this complication. Whether or not explicit formation or role discovery is important to trajectory prediction is not well-established, however.

2.2 Role Discovery

Discovering a player's function on the court is an interesting challenge. The naive approach of tracking players by their personal identity across plays, games, seasons, and teams may yield a more confused analysis. Therefore, one common approach to analyzing sports data, especially with trajectories, is to develop some kind of "role" categorization, e.g. a point guard.

Work first done in [11], extended in [19], patented in [3], and updated in [9] proposes an extensive set of efficient methods for classifying roles, especially within field hockey. Role discovery is an important strand of research, as it emphasizes the semantics of the game. While trajectory prediction is a compelling problem in its own right, role discovery highlights the underlying structure of a particular game. Through role discovery, we can provide interpretable labels or classifications to particular player formations and movements.

Role discovery has also garnered popular attention [1,2,4,12,20], with a variety of approaches in constructing and classifying roles.

2.3 Network Analysis

Network analysis for sports data is comparatively old, with some early efforts in soccer beginning in 1979 [6]. The most relevant type of network analysis, however, has been on *passing networks* and investigates the frequency with which players pass the ball [5,14]. This research direction emphasizes the study of aggregate network properties, e.g. the *centrality* of a player on a particular team [8]. Network analysis thus far has considered the network of passing frequency over an entire game rather than the specific dynamics during the game itself.

3 Semantic Geometric Pipeline

3.1 Geometric Data

Our principal dataset contains the position of the offense, the defense, and the ball—expressed as (x, y)-coordinates—across an entire game capture at 25 frames per second (i.e. 40ms between frames). Notably, basketball is divided into *possessions*, where the teams alternate between defensive and offensive roles. Consider five indexed points as offensive players $O = \{o_1, o_2, o_3, o_4, o_5\} \subset \mathbb{R}^2 : |O| = 5$ and five indexed points as defensive players $D = \{d_1, d_2, d_3, d_4, d_5\} \subset \mathbb{R}^2 : |D| = 5$. One frame F of data is the ordered pair (O, D). Each possession P is a sequence of frames of data and a game is a sequence of possessions.

3.2 Dynamic Passing Networks

From this dataset, we construct *dynamic passing networks*. These networks are defined over the offense (i.e. for basketball, there are five nodes); two players on the same team have an edge joining them if there is no defender in-between. Figure 2 provides some intuition for this construction.

More precisely, given two sets of five offensive players $\mathcal{O} \subset \mathbb{R}^2$ and five defensive players $\mathcal{D} \subset \mathbb{R}^2$, we can define a graph G over vertices $\mathcal{O}$. For $o_i, o_j \in \mathcal{O}$, edge (o_i, o_j) is in the graph if there is no $d \in \mathcal{D}$ such that the *line of sight* $l(o_i, o_j)$ intersects with *occlusion field* $F_r(d)$. The line of sight between two points x, y is the line segment joining them. An occlusion field of radius r at point p is the corresponding l_2-ball centered at p (n.b. $r = 3$, to represent the average 3-foot radius of basketball players).

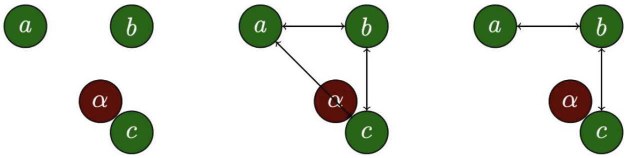

Fig. 2. A deeper look at a snapshot of a basketball game. Green nodes with Roman characters are offensive players. Red nodes with Greek letters are defensive players. On the left, we see player positions with occlusion fields. In the center, we see all offensive lines of sight. On the right, we see the occlusion network with only the offensive edges.

For each frame, we construct these networks by starting with a complete graph with the offensive players as the nodes; we then remove edges from this graph if a defensive player occludes the straight line of sight between a pair of offensive ones. Figure 3 depicts a passing network for one frame.

We convert each frame of the basketball game from two sets of (x, y)-coordinates to a graph. We further compress this representation by only considering graphs up to isomorphism, which allows us to store a label to a representative graph per frame. This procedure thus converts a sequence of frames of position data into a sequence of labels.

Notably, this representation sheds the direct geometry of the basketball game. This sequence of labels provides a purely semantic and highly compressible representation of the game and is justified by three physical assumptions, which are validated by our results:

1. From one frame to the next, there can be at most one edge that changes and this edge change can only occur as a result of well-behaved player trajectories;
2. It is possible for two different geometries to produce the same graph, a sequence of graphs must come from a real play, and thus provide enough information on the possession;
3. Basketball is “fast-paced” enough that it can be assumed to follow a Markovian property. Mainly, players do not have time to consider the history of the game to factor into a future strategy, and instead either follow a set strategy or respond nearly instantaneously to their current environment.

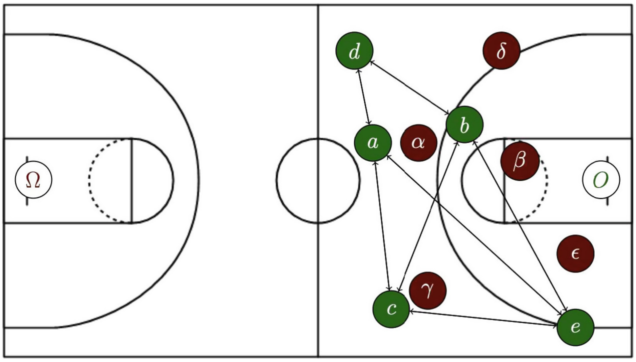

Fig. 3. A frame of the basketball game with the constructed passing network. Green circles with latin letters are offensive players, while red circles with greek letters are defensive players.

3.3 Jump Markov Model

We construct the maximum likelihood estimator of a *jump Markov model* that characterizes this sequence of labels. A Markov model assumes that a sequence of labels exhibits the Markovian property, namely that the label of one entry in the sequence is only influenced by the previous entry and no others. Such a sequence can be characterized by "transition" probabilities: for every possible pair of labels (x, y), the frequency with which y appears right after x in our sequences; we call this a (time-homogeneous, discrete, finite) Markov chain, which is a stochastic process E_k.

A twist on such a model is a "jump" Markov model that additionally assigns a *hold time* to each state: namely, since a basketball game is a continuous-time process, we can also capture the average length of time that our sequence of frames does not change. More precisely, consider a Poisson process, which is a continuous-time stochastic process with rate λ and associated counting process $\{N(t)\}$. Overlapping the discrete-time Markov chain on the continuous-time Poisson process (where the "events" that occur are transitions of the Markov chain) yields a jump Markov model, defined as the continuous-time stochastic process $X(t)$ such that

$$X(t) = E_{N(t)}$$

which is our stochastic process of interest [16].

3.4 A Transformer

Finally, to validate the empirically constructed jump Markov model, we use a Transformer, which is a state-of-the-art deep neural network that excels at two tasks: sequence completion and sequence translation. Transformer is generally a staple of natural language processing, but can work in a variety of different sequence-related tasks [17]. For our experiments, we retained all of the standard architectural elements of the out-of-the-box Transformer, only making a mild alterations as necessary (described in the following sections).

For our experiments, we divided the frames into short sequences of length 50; with a standard 80-10-10 allocation, these sequences were then split into a training, validation, and test set respectively.

Baseline: Sequence Completion. To establish a baseline, we first used Transformer Decoder for sequence completion: given 40 frames of positional data, we used Transformer to predict 10 frames of positional data. For this task, we removed the standard lookup embedding layer for transformer, and instead directly concatenated all player positions to construct a vector in $\mathbb{R}^{10\times 2} = \mathbb{R}^{20}$ (10 players, each with an x and y coordinate).

Comparison: Sequence Translation. For our second task, we leveraged Transformer to "translate" between the sequences produced in the state space of the jump Markov model to positional data. In particular, we first converted positional data into passing networks. Then, we compared all of these networks up to isomorphism to create a "library" of possible networks; each possible network was assigned a unique token. We could thus convert sequences of passing networks into sequences of tokens. Finally, feeding this sequence of tokens into Transformer yielded predicted positional data, which we compared against the original raw trajectories. This setup corresponds directly to Fig. 1.

4 Results

4.1 Markovian Property

First, we consider the sequence of graphs that we construct from the geometric data. Understanding the dynamics of these graphs provides insight into the overall dynamics of the game. It is key to find a suitable model that captures this behavior. Fortunately, our data seems to express a Markovian property and we can therefore use a Markov model to capture the essential elements of our graph sequences. In this section, we provide empirical validation for this claim.

To begin with, we recall our assumption that dynamic passing networks can only change by one edge at a time. From a theoretical perspective, we could imagine that, with a high enough sampling frequency, a tie would be unlikely; and, we could also enact some tie-breaking scheme. Empirically, 99.3% of changes in number of edges are within one edge, which we could bring to 100% if we could sample more frequently. In fact, given that 99.3% is quite close to saturation, we can conclude that our sampling frequency is nearly correct: neither too often nor too sparse.

This assumption, that only one edge changes at one time, is convenient, as it allows us to study the change in number of edges of the graphs, which is much simpler to analyze. In other words, we can convert a sequence of graphs into a sequence of number of edges and learn much about our system without having to rely on a full classification of the graphs. We state a fundamental assumption: if the number of edges does not

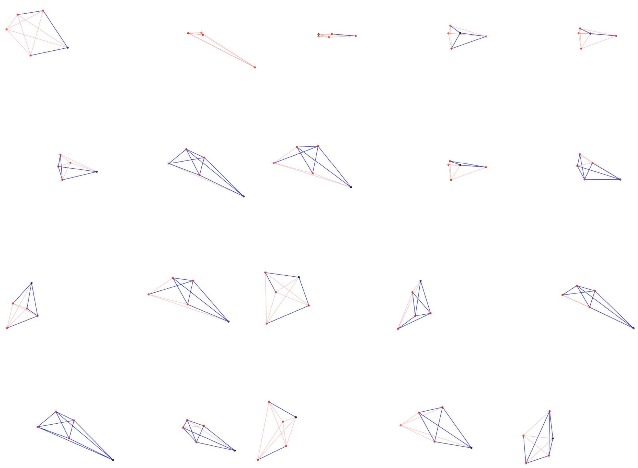

Fig. 4. Sample library of graphs for a possession, descending in order of frequency from left to right and top to bottom.

change in a graph sequence, then the graph (modulo isomorphism) does not change either; this fact follows directly from our assumption that only edge can change at a time. Figure 5 indicates the number of edges in the passing network sequence over time (i.e. per frame).

If we model the number of edges as a Markov chain, we can verify the Markovian property of our edge count data by checking how a change in edges predicts the next change. The edge change probabilities over an entire game are given in Table 1. This data for the game accurately reflects the distribution for each possession, as well, which obviates some concerns about variability within possessions.

Table 1. Probability table for changes in number of edges. The first row indicates that if the previous change was a decrease in number of edges, then there was a 7%, 87%, and 6% chance respectively that the next change in number of edges was a decrease, no change, or increase. The second row follows the same pattern given the previous change was no change in the number of edges. The third row is the same for an increase in number of edges.

	–	0	+
–	0.07	0.87	0.06
0	0.05	0.89	0.05
+	0.07	0.85	0.08

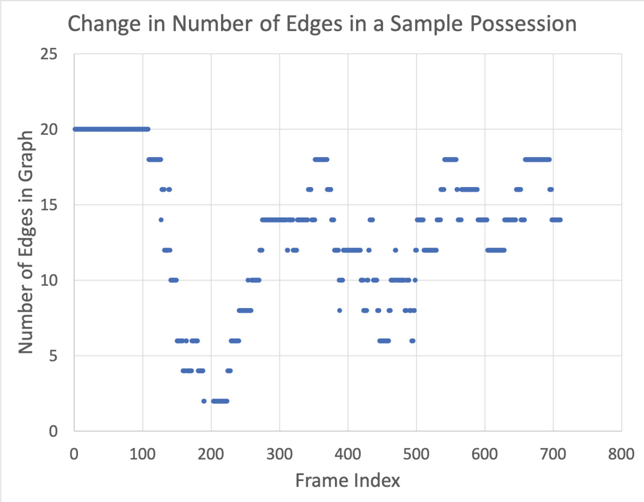

Fig. 5. The number of edges for a particular possession in the game. Mathematically, the number of edges are a random walk over the frames of data within a possession.

The data in Table 1 shows us that the previous change in edges does not influence a future change in edges, in that edge change increments are in fact independent. This data verifies that the number of edges over the course of a possession is in fact Markovian, which is a useful yet surprising property.

Moreover, this suggests that the graph sequence itself is Markovian because of our underlying assumption that one edge change at a time can occur. Therefore, we can use a Markov model to approximate the sequence of graphs over the course of a possession. Finding the maximum likelihood estimator of a Markov chain is also quite simple, as it is given by the empirical transition probabilities.

4.2 Jump Markov Model

While interpreting the sequences of graphs as a Markov chain is a big step towards understanding the underlying dynamics, minor adjustments help us refine the model. In particular, we observe from Table 1 that the overwhelming behavior is for the number of edges in the graph sequences, and hence the sequences themselves, not to change.

Labeling the graph sequences up to isomorphism and constructing the transition probability matrix reveals that the majority of the density is placed on the diagonal, which indicates a self-transition. This behavior warrants some additional investigation. To wit, a basketball game is obviously continuous, but our Markov chain with a transition matrix is discrete. For this reason, we lift the Markov model into a jump Markov model, which allows for different "hold times" per state, i.e. the distribution of time spent in a particular state before a transition occurs. In the naive Markov chain, these hold times are modeled as self-transitions, whereas in a jump Markov model, these are modeled as occurring along some exponential distribution.

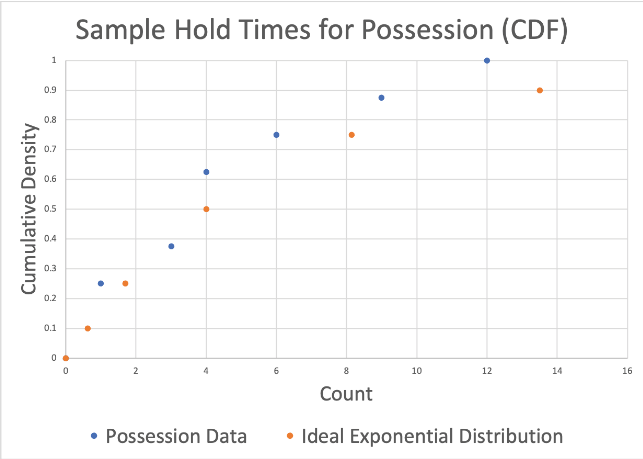

Fig. 6. Sample hold time distribution as a cumulative density function of a representative graph in a representative possession. The blue dots are the true data, which have an average of 5, giving an exponential distribution with parameter $\frac{1}{5}$. The orange dots represent an ideal exponential distribution with the same parameter.

Figure 6 provides a sample hold time distribution for a representative graph in a typical possession. In this case, we see the various hold times for the particular labeled graph, and for reference, an ideal exponential distribution. It appears that a jump Markov model with explicit hold times per graph provides a more robust perspective on state changes.

A jump Markov model displays the stable configurations of players throughout the game. The probability transition matrix for the model suggests the "semantics" of the dynamics, insofar as we can track likely changes in configurations for the players. The hold times demonstrate the overall "stability" of a configuration, i.e. how much time is spent in a particular conformation.

We also can generate a "library" of graphs that provide insight into which configurations appear most often and what typical sequences of graphs look like. Figure 4 is such a representative library that is presented in descending order by frequency of occurrence in a possession.

4.3 Translating with Transformer

Our final set of experiments comes from using a Transformer model. We attempt a standard task in invasion sports, namely predicting the trajectory of player positions across the court. We use a 40–10 split, predicting 10 frames of data from 40 given frames.

The first model is a naive setup where we attempt to predict trajectory data from the geometric data present in a frame alone. Using the Transformer encoder, the position data is fed as a sequence to just the encoder system (6 encoder layers) and a final linear layer to convert the Transformer memory into a concatenated vector of position data.

The second model is the full Transformer setup as described in Sect. 3.4 where the encoder accepts a full sequence of geometric data and the decoder operates on the geometric data as described.

Holding all factors constant, such as learning rate (0.0001), optimization algorithm (Adam), batch size (64), embedding dimension (200), number of layers (6), number of attention heads (8), and all other hyperparameters, we can see the effects of translating from the sequence of graphs through ablation.

The validation loss is set as the sum mean-square error of the 10 frames to be predicted. Namely, when predicting frames $x_{41}, \ldots, x_{50}$ with model output $\hat{x}_{41}, \ldots, \hat{x}_{50}$, the validation error is

$$\frac{1}{10} \sum_{i=41}^{50} (x_i - \hat{x}_i)^2$$

Importantly, this experimental setup provides us with direct insight into how useful and fruitful the jump Markov model is. By using a naive baseline with Transformer, we essentially are performing an ablation analysis, where we see the effect of our entire graph extraction setup. In this context, the MSE score (i.e. the validation error defined above) provides us with a useful quantitative result on the utility of our entire pipeline.

Over the course of several training runs that were run to convergence, the naive model achieves an MSE score of approximately $\mathbf{280}$, while the translation model achieves a best MSE score of $\mathbf{94}$ (lower MSE is better). This dramatic decrease in MSE through translation represents the effect of including the graph data, instead of attempting to learn position data directly.

5 Conclusions

This paper presents a semantic analysis of dynamic networks that are derived from geometric data. This idea is both novel in principle, but we demonstrate a quantitative benefit, as well, of using this approach. By converting geometric position data to graph data, we can leverage the gleaned structure to improve the overall accuracy of downstream applications, like trajectory prediction. Additionally, the construction of a jump Markov model provides clarity into the overall structure of a game through two important properties: first, a transition matrix that indicates which configuration of players lead to others and second, hold times, which suggest the stability of particular configurations in context. By constructing this jump Markov model, we can develop insights into our data that goes beyond the geometry.

From our experiments, we can conclude that, by incorporation the graph data, our pipeline does provide substantial practical insight. Through the completion-vs-translation setup, we can quantify exactly how much information is extracted out of this setup; the compelling drop in MSE garnered by the use of graph data underscores the importance of extracting and using the graph data.

Future work could use our semantic extraction toolkit on a variety of dynamic networks; dynamic networks appear in a plethora of natural contexts, and studying these sequences can be arduous. By analyzing them semantically, we can provide further insights into the dynamics. Additionally, though semantic extraction is a discipline that is at least one hundred years old, we would like to build a more solid theoretical foundation for the particular case of semantic extraction from graphs.

In summary, this paper presents a novel model that leverages dynamic graphs and their semantics to provide a deeper understanding of geometric data. Basketball is but one source of geometric (or even network) data. The basic conceit of extracting semantics from data opens novel avenues of research that we intend to explore further in the future.

References

1. Baker, E.: Redefining Basketball Positions with Unsupervised Learning. Towards data science (2017)
2. Beckham, J.W.: Analytics Reveal 13 New Basketball Positions. Wired (2012)
3. Bialkowski, A., Park, S., Carr, G.P., Matthews, I., Yue, Y.: Analysis Of Team Behaviors Using Role And Formation Information (2018)
4. Cheng, A.: Using Machine Learning to Find the 8 Types of Players in the NBA. Fastbreak Data (2017)
5. Gonçalves, B., Coutinho, D., Santos, S., Lago-Penas, C., Jiménez, S., Sampaio, J.: Exploring team passing networks and player movement dynamics in youth association football. PLoS ONE **12**(1), 1–13 (2017)
6. Gould, P., Gatrell, A.: A structural analysis of a game: the Liverpool v Manchester united cup final of 1977. Soc. Netw. **2**(3), 253–273 (1979)
7. Graber, C., Schwing, A.: Dynamic Neural Relational Inference for Forecasting Trajectories. Cvpr (2020)
8. Gudmundsson, J., Horton, M.: Spatio-temporal analysis of team sports. ACM Comput. Surv. **50**(2), 1–34 (2017)
9. Hobbs, J., Holbrook, M., Frank, N., Sha, L., Lucey, P.: Improved Structural Discovery and Representation Learning of Multi-Agent Data, pp. 1–16 (2019)
10. Li, M.G., Jiang, B., Zhu, H., Che, Z., Liu, Y.: Generative attention networks for multi-agent behavioral modeling. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI, pp. 7195–7202 (2020)
11. Lucey, P., Bialkowski, A., Carr, P., Morgan, S., Matthews, I., Sheikh, Y.: Representing and discovering adversarial team behaviors using player roles. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 2706–2713 (2013)
12. Man, H.: Defining Modern NBA Player Positions - Applying Machine Learning to Uncover Functional Roles in Basketball. Medium.com (2017)
13. Morris, B.T., Trivedi, M.M.: A survey of vision-based trajectory learning and analysis for surveillance. IEEE Trans. Circ. Syst. Video Technol. **18**(8), 1114–1127 (2008)
14. Passos, P., Davids, K., Araújo, D., Paz, N., Minguéns, J., Mendes, J.: Networks as a novel tool for studying team ball sports as complex social systems. J. Sci. Med. Sport **14**(2), 170–176 (2011)
15. Rudenko, A., et al.: Human motion trajectory prediction: a survey. Int. J. Robot. Res. **39**(8), 895–935 (2020)
16. Schuette, C., Metzner, P.: Markov Chains and Jump Processes: An Introduction to Markov Chains and Jump Processes on Countable State Spaces. Freie Universitaet Berlin (2009)

17. Vaswani, A., et al.: Attention is all you need. In: Guyon, I., et al. (eds.) Advances in Neural Information Processing Systems, vol. 30, pp. 5998–6008. Curran Associates, Inc. (2017)
18. Yeh, R.A., Schwing, A.G., Huang, J., Murphy, K.: Diverse generation for multi-agent sports games. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 4605–4614 (2019)
19. Yue, Y., Lucey, P., Carr, P., Bialkowski, A., Matthews, I.: Learning fine-grained spatial models for dynamic sports play prediction. In: Proceedings - IEEE International Conference on Data Mining, ICDM, pp. 670–679 (2014)
20. Christopher, Z.: Redefining Positions and Players In Todays NBA. Using Machine Learning, The Sports Scientist (2019)