Automatically Discovering Offensive Patterns in Soccer Match Data

Jan Van Haaren[1], Vladimir Dzyuba[1], Siebe Hannosset, and Jesse Davis[1]

KU Leuven, Department of Computer Science
Celestijnenlaan 200A, 3001 Leuven, Belgium
{jan.vanhaaren,vladimir.dzyuba,jesse.davis}@cs.kuleuven.be

Abstract. In recent years, many professional sports clubs have adopted camera-based tracking technology that captures the location of both the players and the ball at a high frequency. Nevertheless, the valuable information that is hidden in these performance data is rarely used in their decision-making process. What is missing are the computational methods to analyze these data in great depth. This paper addresses the task of automatically discovering patterns in offensive strategies in professional soccer matches. To address this task, we propose an inductive logic programming approach that can easily deal with the relational structure of the data. An experimental study shows the utility of our approach.

Keywords: Sports analytics, Spatial data, Strategy detection

1 Introduction

Michael Lewis' book *Moneyball* [11] tells the story of Oakland A's General Manager Billy Beane who relies on statistics to build a competitive baseball team despite a tight budget. In recent years, his work has been an example for many other ball sports like basketball, football, and soccer. While several aspects of baseball games can be analyzed in a rather straightforward way, this is much harder for more continuous sports where players can freely move around the pitch. As a result, it can be challenging to quantify the performances of individual players and teams as a whole.

Since simple statistics (e.g., the number of shots on target in soccer) fail to capture the complex interactions among players, companies have started developing tracking technology that captures the location of both the players and the ball at a high frequency (e.g., [16,17,18,20]). These positional data do not only tell *how often* a particular event happened in a match but also *when*, *where*, and *how*. While many professional sports clubs have access to large volumes of performance data, the valuable information that is hidden in these data is only used to a limited extent in their decision-making process. What is missing are the computational methods to analyze these data in greater depth.

In this paper, we propose the task of automatically discovering patterns in offensive strategies in professional soccer matches. More specifically, we are interested in revealing which interactions among players (e.g., a pass from one zone of the pitch to another zone) are most likely to lead to goal attempts. The low-scoring and continuous nature of soccer matches makes this a challenging task. To address this task, we propose an inductive logic programming approach that can easily deal with the relational structure of the data.

The contributions of this paper are as follows:

- We propose using **advanced data mining algorithms** to analyze positional sports data. Most of the techniques that have been proposed to date are statistical and cannot easily deal with the relational nature of these data.
- We present an **inductive logic programming approach** to automatically discover patterns that frequently appear in successful offensive strategies.
- We perform an **empirical study** on a large volume of soccer matches.

2 Related Work

This section provides an overview of the related work on supervised knowledge discovery and sports analytics. The relevant background on inductive logic programming, which is the core of our approach, is provided in Section 4.

Knowledge discovery. The problem addressed in this paper is an instance of supervised descriptive rule discovery [8]. A common variant of this problem is subgroup discovery [5]. Although early variants already supported multi-relational data [22], the data are typically merged into a single table before applying subgroup discovery algorithms [10]. By contrast, inductive logic programming techniques allow us to work directly with the relational (logical) representation of data. This is important for our task, where we want to capture both spatial and temporal patterns as well as interactions among groups of players. An alternative perspective on relational data mining relies on database theory [7].

Sports data analysis. The amount of available data about various sports is constantly increasing, most importantly tracking data and event data [14]. Within soccer, the analysis of tracking data focuses on discovering individual or collective movement patterns, e.g., spectral clustering of trajectories [6], strategy analysis with occupancy maps [12], or formation analysis via minimum entropy partitioning [1]. Gyarmati et al. use event data to discover motif patterns in pass sequences [4]. Most of the research studies large datasets encompassing multiple teams or even leagues, whereas we focus on a single team, with the ultimate goal to improve its performance.

3 Dataset

Through our collaboration with a Belgian soccer club, we obtained play-by-play data for 70 soccer matches in the 2013/2014 and 2014/2015 seasons. The dataset consists of 59 matches in the Belgian Pro League, nine matches in the

UEFA Europa League and two matches in the Belgian Cofidis Cup. The data were collected by data provider Prozone [18]. We first discuss the structure of the data and then introduce additional hierarchical information to enrich the dataset.

3.1 Structure of the data

The data for each match is provided as an XML file which consists of three parts: a *match sheet* with information on the players and managers, a *sequence of events*, and *tracking data* for all players as well as the ball. While the first two parts are available for all matches, the third part is only available for 10 Jupiler Pro League and 4 UEFA Europa League matches.

The match sheet contains each player's name, position on the pitch, jersey number, and team. In addition, it also specifies which players were starters and which players were substitutes.

The sequence of events contains roughly 2,600 events per match. Over 40 different types of events are recorded. The most frequent events include passes between players, players running with the ball, players receiving a ball, players shooting towards goal, players fouling another player, players crossing the ball, and players clearing the ball. Furthermore, events exist to mark the start and end of each half as well as yellow cards, red cards, and substitutions.

The following information is available for each event: the type of the event, the players that are involved, a timestamp, the start location of the event, and the end location of the event if applicable. Depending on the type of event, additional information is available such as the body part involved (e.g., foot or head), type of play (i.e., open or set play), or whether or not a shot was blocked.

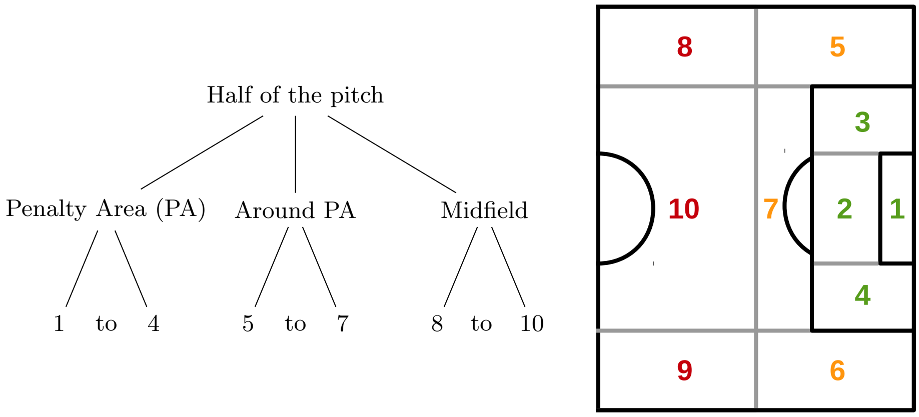
3.2 Hierarchical information

Since we prefer more general patterns to very specific patterns, we enrich the dataset with hierarchical information about both the pitch and the players. This information groups together parts of the pitch and players that fulfill a similar role and hence can be treated in a similar way. As a result, this information facilitates generalizing from very specific to more general knowledge.

We divide each half of the pitch into ten zones resulting into twenty different zones as is shown on the right side of Figure 1. Assuming the team of interest always plays from left to right, we define a hierarchy as follows. We group together zones 1 to 4 as the *penalty area*, zones 5 to 7 as the *area around the penalty area*, and zones 8 to 10 as the *midfield*. The division is identical for the defensive and offensive half of the pitch.

Similarly, we group together players that play in a similar position. We define four groups of players for the team of interest: *goalkeepers*, *defenders* (i.e., center backs, full backs, wing backs, and sweepers), *midfielders* (i.e., defensive midfielders, central midfielders, attacking midfielders, and wing midfielders), and *attackers* (i.e., wingers, supporting strikers, and strikers).

Fig. 1. Each half of the pitch is divided into ten zones, which we group together into three bigger areas. Zones 1 to 4 are the *penalty area*, zones 5 to 7 the *area around the penalty area*, and zones 8 to 10 the *midfield*. The division is identical for the defensive and offensive half of the pitch.



4 Background

This section provides the relevant background on first-order logic, inductive logic programming [13], and the inductive logic programming system Aleph [19].

4.1 First-order logic

First-order logic (FOL) is commonly used as representation language for relational data. In this paper, we consider a subset of FOL, where the alphabet consists of only three symbols. *Constants* start with a lower-case letter and refer to specific objects (e.g., a player $\mathtt{p_i}$). *Variables* start with an upper-case letter and range over multiple objects (e.g., `Players`). *Predicates* represent relations between objects (e.g., a pass $\mathtt{Pass}(\mathtt{p_i}, \mathtt{p_j})$).

Using these three symbols, we can define the following four constructs: *atoms* $\mathtt{p}(\mathtt{t_1}, ..., \mathtt{t_n})$, where the $\mathtt{t_i}$ are constants or variables; *literals*, which are atoms or their negations; *clauses*, which are disjunctions over finite sets of literals; and *definite clauses*, which are clauses containing precisely one positive literal. Definite clauses are often written in implication form $B \implies H$, where B is a conjunction of literals and H is a single literal. A *definite program* is a finite set of definite clauses. Definite programs form the basis of logic programming. We assume all variables to be universally quantified.

4.2 Inductive logic programming and Aleph

Inductive logic programming (ILP) [3] is a well-known framework for learning models, in the form of definite programs, from relational data. ILP offers the benefits of being able to directly model important relationships and it also facilitates incorporating domain knowledge into the learning process. Informally, ILP attempts to learn a definite program that, in combination with background knowledge, can be used to distinguish positive and negative examples. The ILP learning task can be defined as follows:

Given: A target predicate `T`, background knowledge `BK`, a non-empty set of *positive* examples `E+` of `T`, and a set of *negative* examples `E−` of `T`.
Learn: A set of definite clauses `S` such that $BK \wedge S \models E^{+}$ and $BK \wedge S \not\models E^{-}$.

It is often not possible to ensure $BK \wedge S \not\models E^{-}$ in practice. Hence, this condition is relaxed and clauses in S are permitted to cover some negative examples.[1] The goal in the relaxed setting is to achieve a balance between the number of positive and negative examples that each clause covers.

In this paper, we employ the widely-used Aleph ILP system [15,19,21]. Aleph applies a two-step approach to learn a clause. In the *saturation* step, the system first selects a random positive example, called the seed example, and finds all facts in the background knowledge that are true for this example. It forms a clause where the body is the conjunction of all these facts and the head is the target predicate. This is the most-specific clause (i.e., the bottom clause) that covers the seed example. In the *search* step, the system performs a top-down search over clause bodies that generalize the bottom clause. The key idea is that a subset of the facts can be used to explain the seed example's label and that this explanation is likely to apply to other examples as well.

5 Approach

This section introduces our ILP approach to automatically discover patterns that frequently appear in successful offensive strategies. We explain how we pre-process the data and learn the clauses.

5.1 Pre-processing the data

As explained in Section 3, the dataset consists of one long sequence of events for each match. We split each sequence into a number of *phases*, each of which is a subsequence of related events. A phase typically starts with a goal kick or a throw-in and ends when the ball goes out of play or a foul is made. We only consider passes, crosses, set pieces and shots, and discard all other events. We also only consider phases in which the team of interest is *dominant*, which is when its players are involved in at least half of the events. Although this rarely happens, both teams can be seen as the dominant team in the same phase. However, this is not a problem since we are only looking at the team of interest.

[1] By cover, we mean that a clause, in combination with BK, can be used to derive that the target predicate T is true for a given example.

Building examples. In our setting, we define *positive* examples as phases during which the team of interest attempts a shot, and we label all other phases as a *negative* examples. Thus, the target predicate is shot(Phase), which denotes whether the team attempted a shot in a phase Phase. In the background knowledge, we represent each phase as a set of ground facts using four predicates. The $\texttt{pass}(\texttt{Phase}, \texttt{Player}_1, \texttt{Player}_2, \texttt{Zone}_1, \texttt{Zone}_2)$ predicate denotes that in a phase Phase a player $\texttt{Player}_1$ in zone $\texttt{Zone}_1$ passed the ball to $\texttt{Player}_2$ in zone $\texttt{Zone}_2$. Similarly, the $\texttt{cross}(\texttt{Phase}, \texttt{Player}_1, \texttt{Player}_2, \texttt{Zone}_1, \texttt{Zone}_2)$ and $\texttt{set\_piece}(\texttt{Phase}, \texttt{Player}_1, \texttt{Player}_2, \texttt{Zone}_1, \texttt{Zone}_2)$ predicates denote crosses and set pieces. For positive examples, we discard all events following a shot.

Adding background knowledge. We add the hierarchical information about both the pitch and the players as background knowledge (see Section 3.2). What follows are two examples of such clauses for the pass predicate.

$$\texttt{pass}(\texttt{Ph}, \texttt{pl}_1, \texttt{pl}_2, \texttt{Z}_1, \texttt{Z}_2) \implies \texttt{pass}(\texttt{Ph}, \texttt{pMidfielder}, \texttt{pAttacker}, \texttt{Z}_1, \texttt{Z}_2) \quad (1)$$

$$\texttt{pass}(\texttt{Ph}, \texttt{P}_1, \texttt{P}_2, \texttt{z}_2, \texttt{z}_7) \implies \texttt{pass}(\texttt{Ph}, \texttt{P}_1, \texttt{P}_2, \texttt{zPenaltyArea}, \texttt{zMidfield}) \quad (2)$$

Assuming player $\texttt{pl}_1$ is a midfielder and player $\texttt{pl}_2$ is an attacker, Equation 1 denotes that if $\texttt{pl}_1$ passes the ball to $\texttt{pl}_2$, then also a midfielder passes the ball to an attacker. Assuming zone $\texttt{z}_2$ belongs to the penalty area and zone $\texttt{z}_7$ belongs to the midfield (see Figure 1), Equation 2 denotes that a player who passes the ball from $\texttt{z}_2$ to $\texttt{z}_7$ also passes the ball from the penalty area to midfield.

As a practical optimization akin to view materialization in databases, we specify the background knowledge in this way rather than by introducing additional predicates.

5.2 Learning the clauses

The Aleph system supports many different learning modes and search strategies [19]. We apply the induce_max search strategy. In contrast to the default search strategy, this strategy uses each positive example as a seed example. While slower, it produces a larger set of clauses that are potentially of interest to the user. However, this is a natural choice when doing exploratory data mining as our goal is to generate interesting clauses as opposed to learning a very compact predictive model, which is the traditional goal of ILP.

Since we are interested in as many potentially interesting clauses as possible, we run Aleph with as few restrictions as reasonably possible. We set the maximum number of literals per clause (i.e., clauselength) to 5, the minimum number of positive examples covered (i.e., minpos) to 5, the maximum number of negative examples covered (i.e., noise) to 25, and the minimum precision of acceptable clauses (i.e., minacc), which is the ratio between the number of positive examples covered and the total number of examples covered, to 5%.

We sort the learned clauses in descending order according to their m-estimates [2,9], which are smoothed versions of their precisions.

6 Experimental Study

In this section, we present the dataset as well as the different experimental setups, define the research questions, and discuss the experimental results.

6.1 Dataset and experimental setups

After pre-processing the raw data as described in Section 5, the dataset contains $3,803$ examples (phases), including 526 (13.8%) positive examples (shots), and $26,338$ ground facts in total, including $24,786$ passes (94.1%), $1,063$ crosses (4.0%), and 489 set pieces (1.9%). An average example consists of 6.93 ground facts, including 6.52 passes, 0.28 crosses, and 0.13 set pieces. Furthermore, there are 34 constants corresponding to the players of the team of interest.

We investigate the performance of the proposed approach in five setups: discovering spatial patterns with and without hierarchical information, player interaction patterns with and without hierarchical information, and the combined setup with the hierarchical information, in order to evaluate the utility of each type of background knowledge.

6.2 Research questions

In this experimental study, we address the following three research questions:

- **Q1: Do the learned clauses capture the relevant regularities?** The ultimate goal of the analysis is to describe succesful offensive actions of the team. We quantify the capacity of the proposed approach to accomplish this by computing the average m-estimate of the top-ten clauses.
- **Q2: Does the hierarchical knowledge improve the quality of the learned clauses?** One motivation for using ILP is its ability to represent relational data such as the player and zone hierarchies in a natural way. We investigate whether the addition of the hierarchies improves the quality of the learned clauses.
- **Q3: Do the learned clauses describe meaningful patterns?** The purpose of this work is to discover patterns that help the team understand what works well and what does not work well in terms of creating goal-scoring opportunities. Therefore, we qualitatively analyze the discovered patterns.

The proposed approach is meant to facilitate offline performance analysis, e.g., between matches or even seasons. Therefore, it is not necessary to produce instant results. Nevertheless, for the sake of completeness, we report running times for each setup. All experiments are run on a single core of a Linux machine with an Intel Xeon E5645 CPU running at 2.40 GHz and 128 Gb of RAM. We allow Aleph to run for 48 hours in each setup.

Table 1. For each setup, we report the number of clauses returned by Aleph, the maximum and average m-estimate of the precision [9] for the top-ten clauses, and the runtime. Adding hierarchical information barely improves the quality of the clauses in the spatial setup, whereas it considerably improves the quality of the clauses in the player interaction setup. In the setup marked by (⋆), Aleph exceeds the runtime threshold of 48 hours. Hence, we compute the m-estimate on the intermediate output.

			m-est. of prec. (top 10)		
Setup	**Hierarchy**	**Rules**	**Maximum**	**Average**	**Time (min.)**
Spatial		276	0.7396	0.6638	1.15
	✓	323	0.7396	0.7065	441.76
Player interactions		91	0.7396	0.4855	2.95
	✓	257	0.7396	0.6606	2,761.64
Combined	✓	(⋆) 426	0.6374	0.6138	2,880.00

6.3 Results and discussion

We first address Q1 and Q2 by comparing the five setups using statistics on the sets of discovered clauses. We then address Q3 by evaluating the utility of the clauses for the first four setups from a performance analysis point of view.

Quantitative analysis (Q1 and Q2). Table 1 contains an overview of the experimental results. We expect that adding hierarchical information allows Aleph to find clauses of higher quality. We observe a considerable improvement in terms of average m-estimate in the player interaction setup, while this increase is rather modest in the spatial setup. However, the runtime cost of adding hierarchical information is substantial since the search space becomes much larger. In the player interaction setup, Aleph still manages to explore the whole search space and to generate high-quality candidate clauses in terms of m-estimate, which it fails to accomplish in the combined setup.

Qualitative analysis (Q3). Table 2 presents the top-three clauses in terms of their m-estimates for discovering spatial patterns both with and without hierarchical information. These settings have two of their three top-ranked clauses in common (i.e., clauses A and B). Clause A describes a situation where the ball is passed between two players in the left defensive zone (`d5`), from the defensive midfield (`d10`) to the right offensive wing (`o9`), and between two players in the offensive midfield (`o10`). Clause B describes a situation where the ball is passed between two players in the right defensive zone (`d6`) and from the defensive midfield (`d10`) to both the left defensive wing (`d8`) and the left offensive wing (`o8`). Both clauses suggest that the team is particularly successful at creating goal attempts when moving the ball from one flank of the pitch to the other.

Clause D, which leverages the hierarchical information, describes a situation where the ball is passed from the area around the defensive penalty area (`dAPA`) into the defensive penalty area (`dPA`), from the right defensive zone (`d6`) to the right defensive wing (`d9`), and from the offensive midfield (`o10`) to the central offensive area around the penalty area (`o7`). This pattern most probably depicts a counter-attack following a set piece from the opponent.

Table 3 presents the top-three clauses in terms of their m-estimates for discovering player interaction patterns both with and without hierarchical information. These settings have only one of their three top-ranked clauses in common (i.e., clause A). Clause A describes a situation where the goalkeeper (`p1`) passes the ball to a central defender (`p21`) and an attacking midfielder (`p8`) passes the ball to an offensive wing midfielder (`p18`). This pattern makes sense from a performance point of view as both `p8` and `p18` are generally considered key players and responsible for creating a large number of goal-scoring opportunities.

Clause B describes a situation where an offensive full back (`p2`) passes the ball to an offensive wing midfielder (`p18`) and the latter player passes the ball to another wing midfielder (`p9`). This pattern makes sense as well as `p2` has had a foot in many goals scored by the team of interest. Clause C describes a similar

Table 2. Top-three clauses in terms of their m-estimates for discovering spatial patterns with and without hierarchical information. For each clause, we report the total number of examples covered and the number of positive examples covered.

	Clause (C)	$\|C\|$	$\|C^+\|$
	Without hierarchy		
A	pass(d10, o9) ∧ pass(d5, d5) ∧ pass(o10, o10)	5	5
B	pass(d10, d8) ∧ pass(d10, o8) ∧ pass(d6, d6)	5	5
C	pass(d10, o9) ∧ pass(d5, d8) ∧ pass(o10, o7) ∧ pass(o9, o10)	5	5
	With hierarchy		
D	pass(d6, d9) ∧ pass(dAPA, dPA) ∧ pass(o10, o7)	5	5
A	pass(d10, o9) ∧ pass(d5, d5) ∧ pass(o10, o10)	5	5
B	pass(d10, d8) ∧ pass(d10, o8) ∧ pass(d6, d6)	5	5

Table 3. Top-three clauses in terms of their m-estimates for discovering player interaction patterns with and without hierarchical information. For each clause, we report the total number of examples covered and the number of positive examples covered.

	Clause (C)	$\|C\|$	$\|C^+\|$
	Without hierarchy		
A	pass(p1, p21) ∧ pass(p8, p18)	5	5
B	pass(p18, p9) ∧ pass(p2, p18)	6	5
C	pass(p2, p26) ∧ pass(p3, p1)	8	5
	With hierarchy		
A	pass(p1, p21) ∧ pass(p8, p18)	5	5
D	pass(att, att) ∧ pass(mid, att) ∧ pass(mid, def) ∧ pass(p4, p16)	5	5
E	pass(def, att) ∧ pass(def, mid) ∧ pass(opp, p2) ∧ pass(p8, opp)	7	6

Table 4. Top-three clauses in terms of their weighted relative accuracies for discovering spatial patterns with hierarchical knowledge. For each clause, we report the weighted relative accuracy and m-estimate. These clauses are more general and less pure than the top-ranked clauses according to m-estimate for the same setup.

	Clause (C)	$\|C\|$	$\|C^+\|$	WRAcc	m-est.
A	$\texttt{pass(oMF, oMF)} \wedge \texttt{pass(oMF, oPA)} \wedge \texttt{pass(oPA, oAPA)}$	62	18	0.025	0.275
B	`pass(o4, o7)`	43	15	0.024	0.324
C	`set_piece(dAPA, dPA)`	51	16	0.024	0.295

pattern involving a goalkeeper (`p1`), a central defender (`p3`), an offensive full back (`p2`), and a central midfielder (`p26`).

Clauses D and E leverage the hierarchical information about player roles as they include both specific players (e.g., `p4` and `p16`) and positions (e.g., `mid` and `att`). Clause D describes an attack over the left wing involving both an offensive full back (`p4`) and an offensive wing midfielder (`p16`), while clause E describes a situation where an offensive full back (`p2`) intercepts a pass from an opponent (`opp`) and an attacking midfielder (`p8`) attempts a possibly risky pass that is briefly intercepted or touched by an opponent.

Alternative qualitative analysis (Q3). We observed that the top-ranked clauses according to m-estimate are markedly specific. Therefore, we compare these clauses with the top-ranked clauses in the same set of clauses according to *weighted relative accuracy*, which is a common quality measure that aims to balance rule coverage and specificity:

$$WRAcc\,(C) = \frac{|C|}{|E|} \cdot \left(\frac{|C^+|}{|C|} - \frac{|E^+|}{|E|} \right)$$

Table 4 presents the top-three clauses in terms of $WRAcc$ for discovering spatial patterns with hierarchical knowledge. These patterns have a substantially higher coverage, while their m-estimates are much lower. In the same setup, the average m-estimate for the top-ten clauses was 0.707. This contrasts with Op De Beéck et al. [15], where in a similar setting, the coverage of the top-ranked clauses according to m-estimate ranges from 30 to 90 examples. This suggests that different quality measures could reveal different patterns in a dataset. Therefore, if the initial results are unsatisfactory from the domain perspective, ranking the clauses with another quality measure is a reasonable next step.

Clause A describes an attack through the middle, where the ball is passed between two players in the offensive midfield (`oMF`), from the offensive midfield to the offensive penalty area (`oPA`), and from the offensive penalty area to the area around the offensive penalty area (`oAPA`). Clause B describes a pass from the right side of the offensive penalty area (`o4`) to the area in front of the offensive penalty area (`o7`). Clause C describes a set piece from the area around the defensive penalty area (`dAPA`) into the defensive penalty area (`dPA`). Hence, this clause describes a situation where a counter-attack results in a goal-scoring opportunity. These tactical patterns are different from the patterns in Table 2.

7 Lessons Learned

This paper investigated the task of automatically discovering recurring patterns in successful offensive strategies in soccer matches. More specifically, we aimed to reveal both spatial (e.g., a pass from one zone to another) and player interaction (e.g., a pass from one player to another) patterns that are likely to lead to goal attempts. We presented an inductive logic programming approach for this task and demonstrated it is suitable on data from professional soccer matches.

While undertaking this study, we learned the following lessons. First, it is possible to apply inductive logic programming to the task of revealing recurring patterns in soccer match data. It provides the advantages of coping with the relational nature of the data in a straightforward way. Furthermore, it produces interpretable results, which facilitates debugging the data as well as analyzing the results. Second, the discovered patterns make sense from a soccer perspective and are interesting to a domain expert. However, taking the next step forward would require the full tracking data (i.e., the positions of the players and the ball at regular intervals) as this will allow for more fine-grained analysis. Fortunately, this type of data is becoming commonplace. Third, selecting the most interesting clauses is difficult as there is no natural metric or heuristic for this task and a human domain expert is still needed to assist in the interpretation.

In the future, we wish to further expand our current approach. We want to take the order of the events as well as the positions of the players and the ball into account. We also want to account for the differences in playing style of the opponents. Furthermore, we wish to develop a tool that visualizes the discovered patterns (e.g., on a soccer pitch as partially shown in Figure 1). This would help to communicate the patterns in a more intuitive way to a domain expert.

Acknowledgments. Jan Van Haaren is supported by the Agency for Innovation by Science and Technology (IWT). Vladimir Dzyuba is supported by the Research Foundation Flanders (FWO) by means of the project "Instant Interactive Data Exploration". Jesse Davis is partially supported by the Research Fund KU Leuven (OT/11/051), EU FP7 Marie Curie Career Integration Grant (#294068) and FWO-Vlaanderen (G.0356.12).

References

1. Bialkowski, A., Lucey, P., Carr, P., Yue, Y., Sridharan, S., Matthews, I.: Identifying Team Style in Soccer Using Formations Learned from Spatiotemporal Tracking Data. In: Proceedings of the Workshop on Spatial and Spatio-Temporal Data Mining. pp. 9–14 (2014)

2. Cestnik, B.: Estimating Probabilities: A Crucial Task in Machine Learning. In: Proceedings of the 9th European Conference on Artificial Intelligence. vol. 90, pp. 147–149 (1990)
3. Džeroski, S., Lavrač, N.: An Introduction to Inductive Logic Programming (2001)
4. Gyarmati, L., Kwak, H., Rodriguez, P.: Searching for a Unique Style in Soccer. arXiv:1409.0308 (2014)
5. Herrera, F., Carmona, C., González, P., del Jesus, M.: An Overview on Subgroup Discovery: Foundations and Applications. Knowledge and Information Systems 29(3), 495–525 (2011)
6. Knauf, K., Brefeld, U.: Spatio-Temporal Convolution Kernels for Clustering Trajectories. In: Proceedings of the Workshop on Large-Scale Sports Analytics (2014)
7. Knobbe, A.J.: Multi-Relational Data Mining. Ph.D. thesis, Utrecht University (2004)
8. Kralj Novak, P., Lavrač, N., Webb, G.: Supervised Descriptive Rule Discovery: A Unifying Survey of Contrast Set, Emerging Pattern and Subgroup Mining. Journal of Machine Learning Research 10, 377–403 (2009)
9. Lavrač, N., Džeroski, S., Bratko, I.: Handling Imperfect Data in Inductive Logic Programming. Advances in Inductive Logic Programming 32, 48–64 (1996)
10. Lavrač, N., Cestnik, B., Gamberger, D., Flach, P.: Decision Support Through Subgroup Discovery: Three Case Studies and the Lessons Learned. Machine Learning 57(1-2), 115–143 (2004)
11. Lewis, M.: Moneyball: The Art of Winning an Unfair Game. W. W. Norton & Company (2004)
12. Lucey, P., Oliver, D., Carr, P., Roth, J., Matthews, I.: Assessing Team Strategy Using Spatiotemporal Data. In: Proceedings of the 19th International Conference on Knowledge Discovery and Data Mining. pp. 1366–1374 (2013)
13. Muggleton, S., De Raedt, L.: Inductive Logic Programming: Theory and Methods. The Journal of Logic Programming 19, 629–679 (1994)
14. Mutschler, C., Ziekow, H., Jerzak, Z.: The DEBS 2013 Grand Challenge. In: Proceedings of the 7th International Conference on Distributed Event-based Systems. pp. 289–294 (2013)
15. Op De Beéck, T., Hommersom, A., Van Haaren, J., van der Heijden, M., Davis, J., Overbeek, L., Nagtegaal, I.: Mining Hierarchical Pathology Data Using Inductive Logic Programming. Proceedings of the 15th Conference of Artificial Intelligence in Medicine (2015)
16. Opta Sports: `http://www.optasports.com`, accessed: 2015-07-24
17. PlayfulVision: `http://www.playfulvision.com`, accessed: 2015-07-24
18. Prozone: `http://www.prozonesports.com`, accessed: 2015-07-24
19. Srinivasan, A.: The Aleph Manual. Machine Learning at the Computing Laboratory, Oxford University (2001)
20. STATS' SportVU: `http://www.stats.com/sportvu`, accessed: 2015-07-24
21. Vavpetič, A., Lavrač, N.: Semantic Subgroup Discovery Systems and Workflows in the SDM-Toolkit. The Computer Journal 56(3), 304–320 (2013)
22. Wrobel, S.: An Algorithm for Multi-Relational Discovery of Subgroups. In: Proceedings of the 1st European Symposium on Principles of Data Mining and Knowledge Discovery. pp. 78–87 (1997)