MLSA13 - Proceedings of “Machine Learning and Data Mining for Sports Analytics”, workshop @ ECML/PKDD 2013

Albrecht Zimmermann
Jesse Davis

Report CW 650, November 2013

KU Leuven
Department of Computer Science
Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

MLSA13 - Proceedings of "Machine Learning and Data Mining for Sports Analytics", workshop @ ECML/PKDD 2013

Albrecht Zimmermann
Jesse Davis

Report CW 650, November 2013

Department of Computer Science, KU Leuven

Abstract

This document contains the papers presented at MLSA13, the first workshop on Machine Learning and Data Mining for Sports Analytics, organized at ECML/PKDD 2013, held in Prague.

The application of analytic techniques is rapidly gaining traction in both professional and amateur sports circles. The majority of techniques used in the field so far are statistical. While there has been some interest in the Machine Learning and Data Mining community, it has been somewhat muted so far. The goal of this workshop has therefore been two-fold. The first is to raise awareness about this emerging application area. The second is to bring members of the sport analytics community into contact with typical ECML/PKDD contributors, and to highlight what the community has done and can do in the field.

Keywords : Sports Analytics.
CR Subject Classification : I.2, H.2.8

MLSA13 - Proceedings of "Machine Learning and Data Mining for Sports Analytics", workshop @ ECML/PKDD 2013

Albrecht Zimmermann and Jesse Davis

September 27th, Prague, Czech Republic

Preface

This document contains the papers presented at MLSA13, the first workshop on Machine Learning and Data Mining for Sports Analytics, organized at ECML/PKDD 2013, held in Prague.

There were 12 submissions, covering a wide variety of topics. Each submission was reviewed carefully by at least 3 program committee members. Based on the reviews, 10 papers were accepted and presented at the workshop.

The authors of two submissions have expressed the intention to submit their work elsewhere and their papers are therefore only included as abstracts in the proceedings. The full set of papers, and the slides of the workshop presentations, can be downloaded at `http://dtai.cs.kuleuven.be/events/MLSA13/`.

We are grateful to all the authors of the submitted papers, the program committee members, and the reviewers for their time and efforts. We would also like to thank the workshop chairs, Andrea Passerini and Niels Landwehr, and the organizers of the ECML/PKDD 2013, Filip Železný, Hendrik Blockeel, Kristian Kersting, and Siegfried Nijssen.

Program Committee

Jim Albert (Bowling Green State University, USA)
Sathyanarayan Anand (University of Pennsylvania, USA)
Christopher J. Anderson (Cornell University, USA)
Dan Barnett (Analysis Marketing Ltd)
Ulf Brefeld (Technical University of Darmstadt, Germany)
Georgios Chalkiadakis (Technical University of Crete, Greece)
Kurt Driessens (Maastricht University, The Netherlands)
Martin Eastwood (pena.lt/y)
Jeremias Engelmann (Karlsruhe Institute of Technology, Germany)
Juffi Frnkranz (Technical University of Darmstadt, Germany)
Thomas Grtner (Fraunhofer IAIS, Germany)
Kristian Kersting (Fraunhofer IAIS, Germany)
Arno Knobbe (Leiden University, The Netherlands)
Andrey Kolobov (University of Washington, USA)
Richard Maclin (University of Minnesota-Duluth, USA)
Yannis Manolopoulos (Aristotle University of Thessaloniki, Greece)
Max Marchi (Baseball Prospectus)
Johan Pisch (Soccerlogic)
Ravi Ramineni (Analyse Football)
Zoltan Szlavik (IBM, The Netherlands)
Guy Van den Broeck (University of California, Los Angeles, USA)
Marieke van Erp (VU Amsterdam, The Netherlands)

Why do sports officials dropout?

Fabrice Dosseville[1], François Rioult[2], and Sylvain Laborde[1,3]

[1] CESAMS EA 4260, Université de Caen Basse-Normandie, F-14032 Caen, France
e-mail: fabrice.dosseville@unicaen.fr
[2] GREYC-CNRS UMR 6072, Université de Caen Basse-Normandie, France
e-mail: francois.rioult@unicaen.fr
[3] Deutsche Sporthochschule, Institute of Psychology, Am. Sportpark Müngersdorf 6, 50933 Cologne, Germany
e-mail: sylvain.laborde@yahoo.fr

Abstract. Sports officials' recruitment and retention is currently an issue for many sports. The sources of stress are numerous but seem to have a reduced impact on sport officials' dropout. To examine potential reasons of sport officiating dropout, 1718 sport officials were asked to fill a survey about their motivation, the way they trained and are evaluated, perceived stress, the qualities and skills required for officiating, and how they live their function, for a total of 135 questions. Data mining was used to extract information from the data set and transform it into an understandable structure for further use. Results show that intention to dropout among sports officials is related to the main motivation for which they begin officiating: obligation and needs of their sport association to have a sport official.

Strategic Pattern Discovery in RTS-games for E-Sport with Sequential Pattern Mining.

Guillaume Bosc[1], Mehdi Kaytoue[1], Chedy Raïssi[2], and Jean-François Boulicaut[1]

[1] Université de Lyon, CNRS, INSA-Lyon, LIRIS, UMR5205, F-69621, France [2] NRIA Nancy Grand Est, France `firstname.lastname@insa-lyon.fr`, `raissi@inria.fr`

Abstract. Electronic sport, or e-sport, denotes the extreme practice of video games where so-called cyber-athletes compete in world-wide tournaments. As for any sport, such professionals are surrounded by sponsors and practice within professional teams. These professional games are even broadcast by commentators over specialized TV channels. StarCraft II (Blizzard Entertainment) is one of the most competitive video game and has now its own world-wide players ranking system (based on the well-known ELO system) and annual world cup competition series (WCS) with a US$1.6 millions prize pool for the year 2013. Each match between two opponents can be recorded as an exhaustive and noisy sequence of actions. Analyzing these sequences yields an important outcome for game strategy prediction and in-depth game understanding. In this work we report a preliminary study on StarCraft II professional players strategies' discovery based on sequential pattern mining.

Maps for Reasoning in Ultimate

Jeremy C. Weiss[1] and Sean Childers[2]

[1] University of Wisconsin-Madison, Madison, WI, USA
[2] New York University, New York City, NY, USA

Abstract. Existing statistical ultimate (Frisbee) analyses rely on data aggregates to produce numeric statistics, such as completion percentage and scoring rate, that assess strengths and weaknesses of individuals and teams. We leverage sequential, location-based data to develop completion and scoring maps. These are visual tools that describe the aggregate statistics as a function of location. From these maps we observe that player and team statistics vary in meaningful ways, and we show how these maps can inform throw selection and guide both offensive and defensive game planning. We validate our model on real data from high-level ultimate, show that we can characterize both individual and team playing, and show that we can use map comparisons to highlight team strengths and weaknesses.

1 Introduction

The growth of ultimate (Frisbee) in numbers and maturity is leading to rapid changes in the field of ultimate statistics. Within the past several years, tracking applications on tablets have begun providing teams with information about the performance on an individual and group level, *e.g.*, through completion percentages. As these applications are maturing, the application developers need feedback to identify better ways to collect data so that the analysts can query the application to help them guide team strategy and individual development. Likewise the analysts must interact with team leadership to identify the answerable questions that will result in beneficial adjustments to team identity and game approach.

Existing ultimate statistics are akin to the baseball statistics used prior to the spread of sabermetrics[1]. Table 1 shows some of the basic statistics kept on individuals. A similar table is kept for team statistics. These data are relatively easy to capture using a tracking application, as sideline viewers can input the pertinent information as games progress. However, they lose much in terms of capturing the progression of the game and undervalue certain player qualities, for example, differentiating between shutdown defense and guarding idle players (both result in low defensive statistics).

To address some of these shortcomings, first we introduce completion and scoring maps for the visualization of location-based probabilities to help capture high-level strategy. Similar shifts towards visual statistics are taking place in other sports, *e.g.*, in basketball[2]. Completion and scoring maps can be used to

Player	Points	Throws	Completions	%	Goals	Assists	Blocks	Turnovers
Childers	65	88	100	0.88	20	4	10	8
Weiss	40	49	50	0.98	2	10	4	1
Eisenhood	55	20	30	0.67	10	1	20	15

Table 1: Table of the ultimate statistics collected on individuals. Aggregates statistics for teams use similar fields.

reveal team strengths and weaknesses and provide a comparison between teams. Second, we show how these maps can be used to shape individual strategy by recommending optimal throw choices. Finally, we discuss how the maps could be used to guide defensive strategy.

We review the basics of ultimate and ultimate statistics in Section 2. In Section 3 we introduce our visual maps for scoring and completion. In Section 4 we move from conceptual maps to maps based on empirical data. We discuss use cases for maps in Section 5 and offer a broader discussion for continued improvement in statistical analysis of ultimate in Section 6.

2 Background

To begin, we review the basic rules of ultimate. Ultimate is a two-team, seven-on-seven game played with a disc on a rectangular pitch with endzones, and the goal of the game is to have possession of the disc in the opponent's endzone, *i.e.*, a score. The player with the disc must always be touching a particular point on the ground (the pivot) and must release the disc within 10 seconds. If the disc touches the ground while not in possession or the first person to touch the disc after its release is the thrower, the play results in a turnover and a member of the other team picks up the disc with the intent of scoring in the opposite endzone. Each score is worth one point, and the game typically ends when the first team reaches 15.

Collecting statistics can help teams understand the skills and weaknesses of their players and strategies. Table 1 shows some statistics kept to help assess player strengths and weaknesses. While these aggregate statistics can be useful, visual statistics and analyses offer a complementary characterization of individual and team ability. In addition to tabulating player and team statistics over games, we collect locations of throws and catches, giving us location- and sequence-specific data.

3 Completion and scoring maps

Let us consider a game of ultimate. A game is a sequence of points, which are sequences of possessions, which themselves are sequences of plays (throws). Each play, a thrower, a receiver (if any), and their respective locations are recorded.

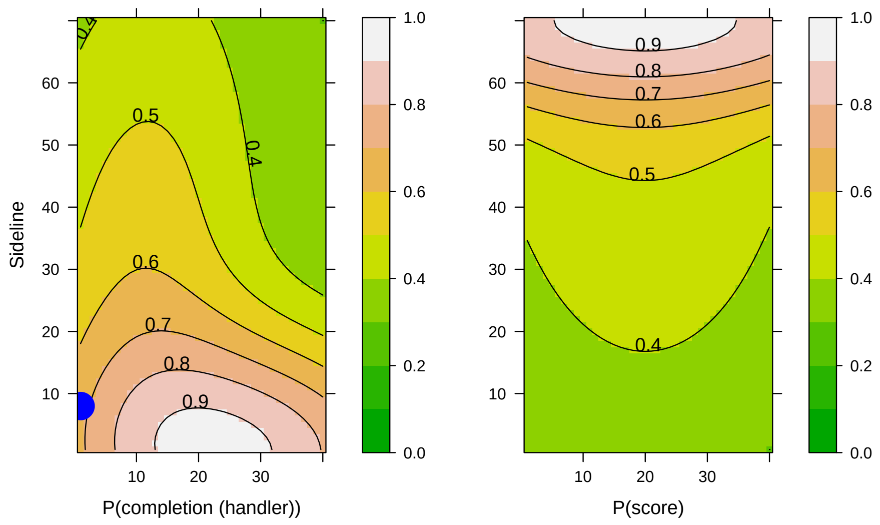


Fig. 1: Completion map (left) for a handler (a thrower) and scoring map (right). The blue circle denotes the thrower location.

It is recorded if a turnover occurs and specifically whether the turnover was due to a block, an interception, or a throwaway. If a completion occurs in the opponent's endzone, a score is recorded.

We introduce a model over throws $\tau \in T$, where throws are specified by players x and locations z. Each player x_i, for $i = 1, 2, \ldots n$, completes a throw τ with probability $p_\tau = p(x^0, z^0, x^1, z^1)$, where τ includes the tuple $((x^0, z^0), (x^1, z^1))$, denoting that player x^0 throws to x^1 from location z^0 to z^1 on the pitch.

Given throws, we can construct a **completion map**. A completion map shows the probability of completion of a throw from player x_i to receiver x_j, based on the receiver location z_j. A map is defined for every starting location z_i of every player x_i. Figure 1 (left) provides a completion map for a player trapped on the sideline (blue dot). As is shown, long throws and cross-field throws are the most difficult throws in ultimate (on average).

Chaining together throws, we define a path ρ that corresponds to a sequence of throws $\tau^0, \tau^1, \ldots, \tau^k$ for some k, where the superscripts denote the throw sequence index. Note the set of paths is countable but unbounded. We also make the Markov assumption that the probability of completing a pass $p(x^i, z^i, x^j, z^j)$ is independent of all passes prior given x^i and z^i. We define a possession ρ' as a path that ends in either a score (1) or a turnover (0). The probability of scoring starting with (x^0, z^0) is then:

$$p(\text{Score}|x^0, z^0) = \sum_{\rho'} \mathbb{1}[\rho']p(\rho') = \sum_{\rho'} \mathbb{1}[\rho'] \prod_{\tau^i \in \rho'} p(x^i, z^i, x^{i+1}, z^{i+1}) \qquad (1)$$

where $\mathbb{1}[\rho']$ equals 1 if the possession results in a score and 0 otherwise. Unfortunately the probability is difficult to compute, but we can approximate it by introducing the probability $p(\text{Score}|z^0) = \frac{1}{n}\sum_{x_i} p(\text{Score}|x_i, z^0)$ that the team scores from a location z^0 on the field, which is the marginal probability of scoring over players. We will use this approximation in Section 5.

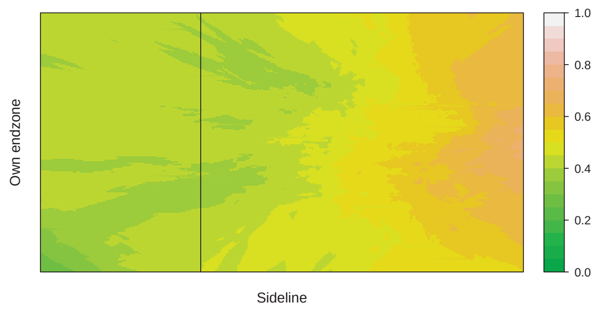
For now, we can use $p(\text{Score}|z^0)$ to define our **scoring map**. A scoring map provides the probability that a team will score from a location z^0 for every location z^0 on the field. As shown in Figure 1 (right), the probability of scoring is high when the disc is close to the opponent's endzone, and low when the disc is far away. From Figure 1 (right) we see it is also advantageous to have the disc in the middle of the field. Ultimate experience suggests that such an increase in scoring probability exists because more in-bounds playing field is accessible with short throws.

To foreshadow, we can use the completion and scoring maps in conjunction to better understand ultimate. We will use it recommend where to throw the disc, how to game-plan for high wind situations, and how to make defensive adjustments. First however, we use data and nearest neighbor methods to show that our model maps reflect existing ultimate beliefs.

4 Data maps

While using simplified models to construct completion and scoring maps (mixtures of Gaussians[3] are used in Figure 1) to depict belief about probabilities in ultimate, we want to verify their validity empirically. We collected data using the UltiApps tracking application[4] based on 2012 film of the Nexgen Tour[5], a team of college-level all-stars who bus from city to city to play the best club teams around the United States. The application stores information in a database with tables for games, teams, points, possessions, and plays, recording the player name (on offense and defense) and location of each throw. We collected data from 13 games, 10 of which were included in our analysis (the other 3 had coding errors). The 10 games included 237 points, 501 possessions, and 3195 throws. We extract relevant information into a single *throws* table. The *throws* table contains IDs of the thrower, receiver and defenders, their locations, the outcome of the throw, indices for possession, point, and game, all ordered sequentially in time.

From the *throws* table, we produce empirical completion and scoring maps. We do this using k-nearest neighbors[6], with k=100. Then for any location z, we find the nearest neighbors and average the probability of scoring from those k positions to get our estimate. Figure 2 shows the results for Nexgen and their opponents. Comparing Figure 1 (right) with Figure 2, we see that the empirical scoring maps show lower scoring probabilities across the pitch, though as our



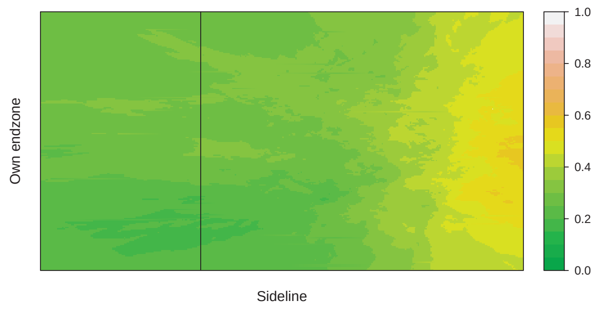


Fig. 2: Empirical scoring maps for Nexgen (top) and opponents (bottom).

model predicted, the proximity to the scoring endzone does improve the chances substantially.

5 Applications

Completion and scoring maps can be used directly as aids for players, but they also have other applications. First we show how maps can determine throw choice. Next, we apply throw choice to show how maps can be used to change offensive strategy given external factors. Finally, we show how defensive ability to manipulate the completion map could guide defensive strategy.

Throw choice The best throw a player can make is the one that maximizes the team's probability of scoring the point. Note that the probability of scoring the point is distinct from the probability of scoring the possession. In this section we relate the two using completion and scoring maps to look at the expected point-scoring outcomes by throw choice.

Recall from Equation 1 that we get the probability of scoring from considering all possible paths given (x^0, z^0). Now we can consider the probability of scoring based on where the player chooses to throw. The probability of scoring with a throw to (x^1, z^1) is given by the probability of completing the first throw times

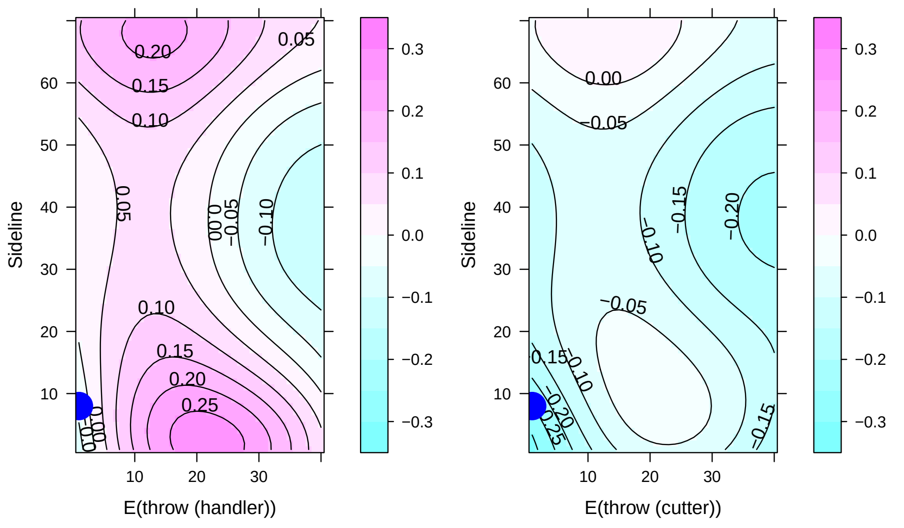


Fig. 3: Expected score maps for handlers (left) and cutters/windy conditions (right), *i.e.*. lower probability of completion on the right (lowered by 20 percent). Note the best decisions are a short throw (left) but a long throw/huck (right).

the probability of scoring from the receiver location:

$$\begin{aligned} p(\text{Score}|(x^0,z^0),(x^1,z^1)) &= p(x^0,z^0,x^1,z^1)p(\text{Score}|x^1,z^1) \\ &\approx p(x^0,z^0,x^1,z^1)p(\text{Score}|z^1) \end{aligned}$$

This approximation is assuming that the particular receiver of the first throw x^1 does not affect the scoring probability. Using this approximation, we can use the completion map to get $p(x^0,z^0,x^1,z^1)$ and the scoring map to get $p(\text{Score}|z^1)$. By decomposing the probability, we now have an approximation to the probability of scoring given our throw choice.

To find the (approximately) optimal throw, however, we should consider the expected value given throw choice, not the probability of scoring the possession. The expected value can be determined by assigning a value of +1 to a score, and -1 to an opponent score. To simplify further, let us assume that each team only gets one chance to score. If neither team scores in their first possession, the expected value is 0. Then we can determine the expected value using the completion map, the scoring map, and the opponent scoring map. Figure 3 shows the expected value map given the maps from Figure 1. Then, we can find the maximum expected value, and instruct the player to throw to that location.

Note that the difference in expected values may seem small–just a fraction of a score. However, they add up. In the Nexgen games there were an average of 300 throws per game. If a situation presents itself, for example, 10 times in a game, choosing an action that makes a difference of 0.1 each time results in scoring an extra point on average.

Offensive strategy in the wind External factors can govern the completion and scoring maps. For example, windy conditions lower the probability of completion, and thus the probability of scoring on a possession. By understanding or approximating changes in the probabilities, a team can change its mindset. The map in Figure 3 (right) shows the new expected value map if you lower the probability of completion in the completion and scoring graphs. While the original expectation graph in Figure 3 (left) recommended throwing a short pass (called a reset), in high wind the graph recommends a long pass (called a huck).

Defensive strategy We noted that offensive game-planning, *e.g.* should a team throw long, high-risk throws, is affected by knowledge of location-based scoring probabilities. Similarly, defensive strategies will affect the opponent completion and scoring probabilities as well. Using scoring maps that take into account defensive positioning, we could identify the minimax outcomes that govern optimal offensive strategy given defensive strategy. That is, the defense should employ the strategy that results in the smallest maximum expected value on the map, and the offense should choose the throw that maximizes the expected value on the map.

6 Discussion

The analysis presented highlights the capabilities of completion and scoring maps. Many other uses of the maps and location-based data would be interesting. For one, we can use these maps to characterize players. We can answer questions such as: how do player completion rates change across the field, and does the player have weaknesses or strengths in particular regions? We can also use the expectation maps in conjunction with the *throws* table to assess how much "value" each player contributes to the team by summing up the change in expected values for plays in which the player was involved. Furthermore, we can perform visual comparative analyses between teams (or players). Subtracting the scoring maps from one another (or a baseline) can help identify regions of relative weakness; the empirical comparative scoring map (difference in maps in Figure 2) is shown in Figure 4.

While location-based tracking adds a visual and predictive component that helps describe optimal ultimate play, it does not provide other pertinent information that teams and players must address when making decisions on the field. For example, the location of players not involved the movement of the disc affect the choices made. An alternative analysis could track not only the disc movement but all 14 player's movements. Also, while our analysis uses the sequence of throws (to determine possessions and scores), the analysis is atemporal. Many throws are relatively easy off of disc movement because the defenese is out of

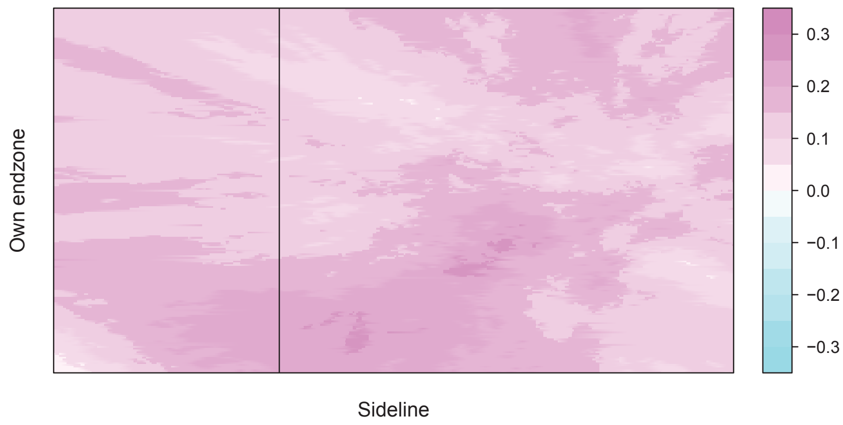


Fig. 4: Empirical scoring map difference (purple indicates Nexgen outperforming opponents).

position, and an atemporal model does not capture these elements of the game. Another important factor, weather condition, goes unmodeled. Incorporating these into our models would help refine our analysis and additional insight into ultimate strategy. Finally, the number of throws available for analysis will always be relatively small, particularly against uncommon strategies or in unusual conditions. Developing strategies and assessing individual ability in the face of limited data will be challenging and should be considered in ultimate analyses.

Acknowledgments

We would like to acknowledge Ultiworld and UltiApps for their continued support and development of ultimate statistics.

References

1. J. Albert, "An introduction to sabermetrics," *Bowling Green State University (http://www-math. bgsu. edu/˜ albert/papers/saber. html)*, 1997.
2. K. Goldsberry, "Courtvision: New visual and spatial analytics for the nba," MIT Sloan Sports Analytics Conference, 2012.
3. B. S. Everitt and D. J. Hand, "Finite mixture distributions," *Monographs on Applied Probability and Statistics, London: Chapman and Hall, 1981*, vol. 1, 1981.
4. "UltiApps stat tracker." `http://ultiapps.com/`. Accessed: 2013-06-28.
5. "NexGen." `http://nexgentour.com/`. Accessed: 2013-06-28.
6. B. V. Dasarathy, "Nearest neighbor ({NN}) norms:{NN} pattern classification techniques," 1991.

Predicting the NFL Using Twitter

Shiladitya Sinha[1], Chris Dyer[1], Kevin Gimpel[2], and Noah A. Smith[1]

[1] Carnegie Mellon University, Pittsburgh PA 15213, USA
[2] Toyota Technological Institute at Chicago, Chicago IL 60637, USA

Abstract. We study the relationship between social media output and National Football League (NFL) games, using a dataset containing messages from Twitter and NFL game statistics. Specifically, we consider tweets pertaining to specific teams and games in the NFL season and use them alongside statistical game data to build predictive models for future game outcomes (which team will win?) and sports betting outcomes (which team will win with the point spread? will the total points be over/under the line?). We experiment with several feature sets and find that simple features using large volumes of tweets can match or exceed the performance of more traditional features that use game statistics.

1 Introduction

Twitter data has been used to predict and explain a variety of real-world phenomena, including opinion polls [18], elections [23], the spread of contagious diseases [20], and the stock market [2]. This is evidence that Twitter messages in aggregate contain useful information that can be exploited with statistical methods. In this way, Twitter may offer a way to harness the "wisdom of crowds" [22] for making better predictions about real-world events.

In this paper, we consider the relationship between National Football League (NFL) games and the Twitter messages mentioning the teams involved, in order to make predictions about games. We focus on the NFL because games are broadcast widely on television throughout the US and teams play at most once per week, enabling many to comment on games via social media. NFL football also has active betting markets. The most well-known is the point spread line, which is a handicap for the stronger team chosen by bookmakers to yield equal bets on both sides. Factoring in the bookmaker's commission, a betting strategy that predicts the winner "with the spread" in more than 53% of games will be profitable. In this paper, we build models to predict game and betting outcomes, considering a variety of feature sets that use Twitter and game statistical data. We find that simple features of Twitter data can match or exceed the performance of the game statistical features more traditionally used for these tasks.

Our dataset is provided for academic research at `www.ark.cs.cmu.edu/football`. It is hoped that our approach and dataset may be useful for those who want to use social media to study markets, in sports betting and beyond.

2 Problem Domain and Related Work

Each NFL regular season spans 17 weeks from September to January, with roughly one game played per week by each team. In each game, the **home team** plays at their own stadium and hosts the **away team**. The most popular wager in NFL football is to choose the team that will win given a particular handicap called the **point spread**. The point spread is a number set by bookmakers that encodes the handicap for the home team. It is added to the home team's score, and then the team with the most points is called the winner **with the spread (WTS)**. For example, if the NY Giants are hosting the NY Jets and the point spread is -4, then the Giants will have to win by at least 4 in order to win WTS. If the Giants win by fewer than 4, the Jets win WTS.[3] Also popular is to wager on whether the total number of points scored in the game will be above or below the **over/under line**.

Point spreads and over/under lines are set by sports betting agencies to reflect all publicly available information about upcoming games, including team performance and the perceived outlook of fans. Assuming market efficiency, one should not be able to devise a betting strategy that wins often enough to be profitable. In prior work, most have found the NFL point spread market to be efficient overall [16, 17, 4], or perhaps only slightly inefficient [6, 5]. Others pronounced more conclusively in favor of inefficiency [25, 9], but were generally unable to show large biases in practice [10].[4] Regardless of efficiency, several researchers have designed models to predict game outcomes [11, 21, 8, 15, 7, 1].

Recently, Hong and Skiena [12] used sentiment analysis from news and social media to design a successful NFL betting strategy. However, their main evaluation was on in-sample data, rather than forecasting. Also, they only had Twitter data from one season (2009) and therefore did not use it in their primary experiments. We use large quantities of tweets from the 2010–2012 seasons and do so in a genuine forecasting setting for both winner WTS and over/under prediction.

3 Data Gathering

We used Twitter (`www.twitter.com`) as our source of social media messages ("tweets"), using the "garden hose" (10%) stream to collect tweets during the 2010–2012 seasons. For the 2012 season, this produced an average of 42M messages per day. We tokenized the tweets using `twokenize`, a freely available Twitter tokenizer developed by O'Connor et al. [19].[5] We obtained NFL game statistics for the 2010–2012 seasons from NFLdata.com (`www.nfldata.com`). The data include a comprehensive set of game statistics as well as the point spread and total points line for each game obtained from bookmakers.

[3] If the Giants win by *exactly* 4, the result is a **push** and neither team wins WTS.

[4] Inefficiencies have been attributed to bettors overvaluing recent success and undervaluing recent failures [24], cases in which home teams are underdogs [5], large-audience games, including Super Bowls [6], and extreme gameday temperatures [3].

[5] `www.ark.cs.cmu.edu/TweetNLP`

Table 1. Hashtags used to assign tweets to New York Giants (top) and New York Jets (bottom). If a tweet contained any number of hashtags corresponding to exactly one NFL team, we assigned the tweet to that team and used it for our analysis.

`#giants #newyorkgiants #nygiants #nyg #newyorkfootballgiants #nygmen #gmen #gogiants #gonygiants #gogiantsgo #letsgogiants #giantsnation #giantnation`
`#jets #newyorkjets #nyjets #jetsjetsjets #jetlife #gojets #gojetsgo #letsgojets #jetnation #jetsnation`

Table 2. Yearly pregame, postgame, and weekly tweet counts.

season	pregame	postgame	weekly
2010	40,385	53,294	185,709
2011	130,977	147,834	524,453
2012	266,382	290,879	1,014,473

3.1 Finding Relevant Tweets

Our analysis relies on finding relevant tweets and assigning them to particular games during the 2010–2012 NFL seasons. We can use timestamps to assign the tweets to particular weeks of the seasons, but linking them to teams is more difficult. We chose a simple, high-precision approach based on the presence of hashtags in tweets. We manually created a list of hashtags associated with each team, based on familiarity with the NFL and validated using search queries on Twitter. There was variation across teams; two examples are shown in Table 1.[6] We discarded tweets that contained hashtags from more than one team. We did this to focus our analysis on tweets that were comments on particular games from the perspective of one of the two teams, rather than tweets that were merely commenting on particular games without associating with a team. When making predictions for a game, our features only use tweets that have been assigned to the teams in those games.

For the tasks in this paper, we created several subsets of these tweets. We labeled a tweet as a **weekly tweet** if it occurred at least 12 hours after the start of the previous game and 1 hour before the start of the upcoming game for its assigned team. **Pregame tweets** occurred between 24 hours and 1 hour before the start of the upcoming game, and **postgame tweets** occurred between 4 and 28 hours after the start of the previous game.[7] Table 3.1 shows the sizes of these sets of tweets across the three NFL seasons.

[6] Although our hashtag list was carefully constructed, some team names are used in many sports. After noticing that many tweets with `#giants` co-occurred with `#kyojin`, we found that we had retrieved many tweets referring to a Japanese professional baseball team also called the Giants. So we removed tweets with characters from the Katakana, Hiragana, or Han unicode character classes.

[7] Our dataset does not have game end times, though NFL games are nearly always shorter than 4 hours. Other time thresholds led to similar results in our analysis.

Table 3. Highly weighted features for postgame tweet classification. *home*/*away* indicates that the unigram is in the tweet for the home or away team, respectively.

predicting home team won			predicting away team won		
home: win	*home*: victory	*away*: loss	*away*: win	*away*: congrats	*home*: lost
home: won	*home*: WIN	*away*: lost	*away*: won	*away*: Go	*home*: loss
home: Great	*away*: lose	*away*: refs	*away*: Great	*away*: proud	*home*: bad

To encourage future work, we have released our data for academic research at `www.ark.cs.cmu.edu/football`. It includes game data for regular season games during the 2010–2012 seasons, including the point spread and total points line. We also include tweet IDs for the tweets that have been assigned to each team/game.

4 Data Analysis

Our dataset enables study of many questions involving professional sports and social media. We briefly present one study in this section: we measure our ability to classify a postgame tweet as whether it follows a win or a loss by its assigned team. By using a classifier with words as features and inspecting highly-weighted features, we can build domain-specific sentiment lexicons.

To classify postgame tweets in a particular week k in 2012, we train a logistic regression classifier on all postgame tweets starting from 2010 up to but not including week k in 2012. We use simple bag-of-words features, conjoining unigrams with an indicator representing whether the tweet is for a home or away team. In order to avoid noise from rare unigrams, we only used a unigram feature for a tweet if the unigram appeared in at least 10 tweets during the week that the tweet was written. We achieved an average accuracy of 67% over the tested weeks. Notable features that were among the top or bottom 30 weighted features are listed in Tab. 3. Most are intuitive ("win", "Great", etc.). Additionally, we surmise that fans are more likely to comment on the referees ("*away*: refs") after their team loses than after a win.

5 Forecasting

We consider the idea that fan behavior in aggregate can capture meaningful information about upcoming games, and test this claim empirically by using tweets to predict outcomes of NFL games on a weekly basis. We establish baselines using features derived from statistical game data, building upon prior work [7], and compare accuracies to those of our predictions made using Twitter data.

5.1 Modeling and Training

We use a logistic regression classifier to predict game and betting outcomes. In order to measure the performance of our feature sets, and tune hyperparameters

Table 4. List of preliminary feature sets using only game statistics, numbered for reference as F_i. *Denotes that the features appear for both the home and away teams.

point spread line (F_1)	over/under line (F_2)
avg. points beaten minus missed spread by in current season* (F_3)	avg. points beaten minus missed over/under by in current season* (F_4)
avg. points scored in current season* (F_5)	avg. points given up in current season* (F_6)
avg. total points scored in current season* (F_7)	avg. (point spread + points scored) in current season* (F_8)
home team win WTS percentage in home games in current season away team win WTS percentage in away games in current season (F_9)	avg. interceptions thrown in current season* avg. fumbles lost in current season* avg. times sacked in current season* (F_{10})

for our model as the season progresses, we use the following scheme: to make predictions for games taking place on week $k \in [4, 16]$ in 2012, we use all games from weeks $[1, 16]$ of seasons 2010 and 2011, as well as games from weeks $[1, k-3]$ in 2012 as training data.[8] We then determine the L_1 or L_2 regularization coefficient from the set $\{0, 1, 5, 10, 25, 50, 100, 250, 500, 1000\}$ that maximizes accuracy on the development set, which consists of weeks $[k-2, k-1]$ of 2012. We follow this procedure to find the best regularization coefficients separately for each feature set and each test week k. We use the resulting values for final testing on week k. We repeat for all test weeks $k \in [4, 16]$ in 2012. To evaluate, we compute the accuracy of our predictions across all games in all test weeks. We note that these predictions occur in a strictly online setting, and do not consider any information from the future.

5.2 Features

Statistical Game Features We start with the 10 feature sets shown in Tab. 4 which only use game statistical data. We began with features from Gimpel [7] and settled upon the feature sets in the table by testing on seasons 2010–2011 using a scheme similar to the one described above. These 10 feature sets and the collection of their pairwise unions, a total of 55 feature sets, serve as a baseline to compare to our feature sets that use Twitter data.

Twitter Unigram Features When using tweets to produce feature sets, we first consider an approach similar to the one used in Sec. 4. In this case, for a given game, we assign the feature (*home*/*away*, unigram) the value log(1+unigram frequency over all weekly tweets assigned to the *home*/*away* team). As a means of

[8] We never test on weeks 1–3, and we do not train or test on week 17; it is difficult to predict the earliest games of the season due to lack of historical data and week 17 sees many atypical games among teams that have been confirmed or eliminated from play-off contention.

noise reduction, we only consider (*home*/*away*, unigram) pairs occurring in at least 0.1% of the weekly tweets corresponding to the given game; this can be determined before the game takes place. This forms an extremely high-dimensional feature space in contrast to the game statistics features, so we now turn to dimensionality reduction.

Dimensionality Reduction To combine the above two feature sets, we use **canonical correlation analysis** (CCA) [13]. We use CCA to simultaneously perform dimensionality reduction on the unigram features and the game statistical features to yield a low-dimensional representation of the total feature space.

For a paired sample of vectors $\mathbf{x}_1^i \in \mathbb{R}^{m_1}$ and $\mathbf{x}_2^i \in \mathbb{R}^{m_2}$, CCA finds a pair of linear transformations of the vectors onto $\mathbb{R}^k$ so as to maximize the correlation of the projected components and so that the correlation matrix between the variables in the canonical projection space is diagonal. While developed to compute the degree of correlation between two sets of variables, it is a good fit for **multi-view learning problems** in which the predictors can be partitioned into disjoint sets ('views') and each is assumed sufficient for making predictions. Previous work has focused on the semi-supervised setting in which linear transformations are learned from collections of predictors and then regression is carried out on the low dimensional projection, leading to lower sample complexity [14]. Here, we retain the fully supervised setting, but use CCA for dimensionality reduction of our extremely high-dimensional Twitter features. We experiment with several values for the number of components of the reduced matrices resulting from CCA.

Twitter Rate Features As another way to get a lower-dimensional set of Twitter features, we consider a feature that holds a signed representation of the level of increase/decrease in a team's weekly tweet volume compared to the previous week. In computing these **rate** features, we begin by taking the difference of a team's weekly tweet volume for the week to be predicted v_{curr}, and the team's weekly tweet volume for the previous week in which they had a game v_{prev} or the team's average weekly tweet volume after its previous game v_{prevavg}. We will use v_{old} to refer to the subtracted quantity in the difference, either v_{prev} or v_{prevavg}. This difference is mapped to a categorical variable based on the value of a parameter Δ which determines how significant we consider an increase in volume from v_{old} to be. Formally, we define a function $\text{rate}_S : \mathbb{Z} \times \mathbb{Z} \times \mathbb{N} \to \{-2, -1, 0, 1, 2\}$, $(v_{\text{old}}, v_{\text{curr}}, \Delta) \mapsto \text{sign}(v_{\text{curr}} - v_{\text{old}}) \lfloor \frac{|v_{\text{curr}} - v_{\text{old}}|}{\Delta} \rfloor$ that is decreasing in its first argument, increasing in its second argument, and whose absolute value is decreasing in its third argument.

This idea of measuring the rate of change in tweet volumes is further generalized by categorizing the difference in volume $(v_{\text{curr}} - v_{\text{old}})$ by computing its percentage of v_{old}, or formally as a function $\text{rate}_P : \mathbb{Z} \times \mathbb{Z} \times (0, 1] \to \{-2, -1, 0, 1, 2\}$, $(v_{\text{old}}, v_{\text{curr}}, \theta) \mapsto \text{sign}(v_{\text{curr}} - v_{\text{old}}) \lfloor \frac{|v_{\text{curr}} - v_{\text{old}}|}{\theta \cdot v_{\text{old}}} \rfloor$ which has the same functional

Table 5. Example of how the rate_S feature is defined with $\Delta = 500$ (left) and how the rate_P feature is defined with $\theta = .2$.

v_{old}	v_{curr}	$\text{rate}_S(v_{\text{old}}, v_{\text{curr}}, 500)$
2000	$(3000, \infty)$	2
2000	$(2500, 3000]$	1
2000	$[1500, 2500]$	0
2000	$[1000, 1500)$	-1
2000	$[0, 1000)$	-2

v_{old}	v_{curr}	$\text{rate}_P(v_{\text{old}}, v_{\text{curr}}, .2)$
2000	$(2800, \infty)$	2
2000	$(2400, 2800]$	1
2000	$[1600, 2400]$	0
2000	$[1200, 1600)$	-1
2000	$[0, 1200)$	-2

properties as the rate_S function. Examples of how the rate_S and rate_P functions are defined are provided in Table 5. Thus, we may take $v_{\text{old}} = v_{\text{prev}}$ or $v_{\text{old}} = v_{\text{prevavg}}$, and categorize the difference using a static constant Δ or a percentage θ of v_{old}, giving us four different versions of the rate feature.

In preliminary testing on the 2010 and 2011 seasons, we found that the rate_S feature worked best with $v_{\text{old}} = v_{\text{prev}}$ and $\Delta = 500$, so we also use these values in our primary experiments below with rate_S. For rate_P, we experiment with $\theta \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$ and $v_{\text{old}} \in \{v_{\text{prev}}, v_{\text{prevavg}}\}$.

5.3 Experiments

We consider three prediction tasks: winner, winner WTS, and over/under. Our primary results are shown in Tab. 7. We show results for all three tasks for several individual feature sets. We also experimented with many conjunctions of feature sets; the best results for each task over all feature set conjunctions tested are shown in the final three rows of the table.

The Twitter unigram features alone do poorly on the WTS task (47.6%), but they improve to above 50% when combined with the statistical features via CCA. Surprisingly, however, the Twitter unigram features alone perform better than most other feature sets on over/under prediction, reaching 54.3%. This may be worthy of follow-up research. On winner WTS prediction, the Twitter rate_S feature (with v_{prev} and $\Delta = 500$) obtains an accuracy above 55%, which is above the accuracy needed to be profitable after factoring in the bookmaker's commission. We found these hyperparameter settings (v_{prev} and Δ) based on preliminary testing on the 2011 season, in which they consistently performed better than other values; the success translates to the 2012 season as well. Interestingly, the Twitter rate features perform better on winner WTS than on straight winner prediction, while most statistical feature sets perform better on winner prediction. We see a similar trend in Tab. 6, which shows results with Twitter rate_P features with various values for θ and v_{old}.

We observed in preliminary experiments on the 2011 season that feature sets with high predictive accuracy early on in the season will not always be effective later, necessitating the use of different feature sets throughout the season. For each week $k \in [5, 16]$, we use the training and testing scheme described in Sec. 5.1 to compute the feature set that achieved the highest accuracy on average over the previous two weeks, starting with week 3. This method of feature selection is similar to our method of tuning regularization coefficients. Over 12 weeks and 177 games in the 2012 season, this strategy correctly predicted the winner **63.8%** of the time, the winner WTS **52.0%** of the time, and the over under **44.1%** of the time. This is a simple way of selecting features and future work might experiment with more sophisticated online feature selection techniques. We expect there to be room for improvement due to the low accuracy on the over/under task (44.1%) despite there being several feature sets with much higher accuracies, as can be seen in Tab. 7.

Another simple method of selecting a feature set for week $k \in [4, 16]$ is choosing the feature set achieving the highest accuracy on average over *all* previous weeks, starting with week 3, using the same scheme described in Sec. 5.1. This feature set can be thought of as the best feature set at the point in the season at which it is chosen. In Fig. 1 we observe that the best feature set changes very frequently, going through 8 different feature sets in a 13-week period.

Table 6. rate_P winner and winner WTS accuracies for different values of θ and v_{old}.

	v_{prev}		v_{prevavg}	
θ	winner	WTS	winner	WTS
0.1	51.0	51.4	51.0	50.0
0.2	53.8	51.0	52.4	45.7
0.3	51.4	52.4	52.4	54.3
0.4	54.8	49.5	51.4	49.5
0.5	52.9	45.2	53.4	49.5

6 Conclusion

We introduced a new dataset that includes a large volume of tweets aligned to NFL games from the 2010–2012 seasons. We explored a range of feature sets for predicting game outcomes, finding that simple feature sets that use Twitter data could match or exceed the performance of game statistics features. Our dataset is made available for academic research at `www.ark.cs.cmu.edu/football`.

Acknowledgments We thank the anonymous reviewers, Scott Gimpel at NFL-data.com, Brendan O'Connor, Bryan Routledge, and members of the ARK research group. This research was supported in part by the National Science Foundation (IIS-1054319) and Sandia National Laboratories (fellowship to K. Gimpel).

Table 7. Accuracies across prediction tasks and feature sets. Lower pane shows oracle feature sets for each task, with the highest accuracies starred.

	prediction tasks		
features	winner	WTS	over/under
point spread line (F_1)	60.6	47.6	48.6
over/under line (F_2)	52.3	49.0	48.6
F_3	56.3	50.0	50.0
F_4	52.3	54.8	50.5
F_5	65.9	51.0	44.7
F_{10}	56.7	51.4	46.6
$\bigcup_i F_i$	63.0	47.6	51.0
Twitter unigrams	52.3	47.6	54.3
CCA: $\bigcup_i F_i$ and Twitter unigrams, 1 component	47.6	50.4	43.8
2 components	47.6	51.0	43.8
4 components	50.5	51.9	44.2
8 components	47.6	48.1	42.3
Twitter rates$_S(v_{\text{prev}}, \Delta = 500)$	51.0	55.3	52.4
$F_5 \cup F_9 \cup$ Twitter rate$_P(v_{\text{prev}}, \theta = .2)$	65.9*	51.4	48.1
$F_3 \cup F_{10} \cup$ Twitter rate$_P(v_{\text{prev}}, \theta = .1)$	56.3	57.2*	48.1
$F_3 \cup F_4 \cup$ Twitter rates$_S(v_{\text{prev}}, \Delta = 200)$	54.8	49.0	58.2*

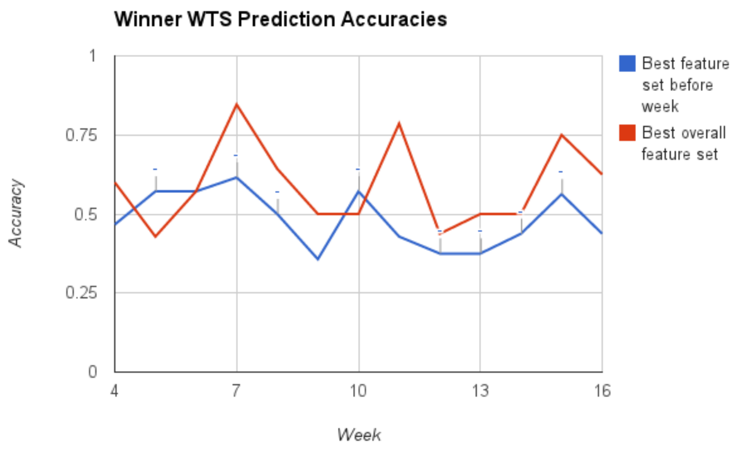


Fig. 1. Weekly accuracies for the best overall feature set in hindsight, and the best feature set leading up to the given week for winner WTS prediction. Marks above the 'Best feature set before week' line indicate weeks where the best feature set changed.

References

1. Baker, R.D., McHale, I.G.: Forecasting exact scores in national football league games. International Journal of Forecasting 29(1), 122–130 (2013)
2. Bollen, J., Mao, H., Zeng, X.J.: Twitter mood predicts the stock market. Journal of Computational Science 2(1) (2011)
3. Borghesi, R.: The home team weather advantage and biases in the nfl betting market. Journal of Economics and Business 59(4), 340–354 (2007)
4. Boulier, B.L., Stekler, H.O., Amundson, S.: Testing the efficiency of the National Football League betting market. Applied Economics 38(3), 279–284 (February 2006)
5. Dare, W.H., Holland, A.S.: Efficiency in the NFL betting market: modifying and consolidating research methods. Applied Economics 36(1), 9–15 (2004)
6. Dare, W.H., MacDonald, S.S.: A generalized model for testing the home and favorite team advantage in point spread markets. Journal of Financial Economics 40(2), 295–318 (1996)
7. Gimpel, K.: Beating the NFL football point spread (2006), unpublished manuscript
8. Glickman, M.E., Stern, H.S.: A state-space model for National Football League scores. JASA 93(441), 25–35 (1998)
9. Golec, J., Tamarkin, M.: The degree of inefficiency in the football betting market : Statistical tests. Journal of Financial Economics 30(2), 311–323 (December 1991)
10. Gray, P.K., Gray, S.F.: Testing market efficiency: Evidence from the NFL sports betting market. The Journal of Finance 52(4), 1725–1737 (1997)
11. Harville, D.A.: Predictions for National Football League games via linear-model methodology. JASA 75(371), 516–524 (1980)
12. Hong, Y., Skiena, S.: The wisdom of bookies? sentiment analysis versus the NFL point spread. In: Proc. of ICWSM (2010)
13. Hotelling, H.: Relations between two sets of variates. Biometrika 28(3–4), 321–377 (1936)
14. Kakade, S.M., Foster, D.P.: Multi-view regression via canonical correlation analysis. In: Proc. of COLT (2007)
15. Knorr-Held, L.: Dynamic rating of sports teams. The Statistician 49(2), 261–276 (2000)
16. Lacey, N.J.: An estimation of market efficiency in the nfl point spread betting market. Applied Economics 22(1), 117–129 (1990)
17. Levitt, S.D.: How do markets function? an empirical analysis of gambling on the National Football League. Economic Journal 114(495), 2043–2066 (2004)
18. O'Connor, B., Balasubramanyan, R., Routledge, B.R., Smith, N.A.: From Tweets to polls: Linking text sentiment to public opinion time series. In: Proc. ICWSM (2010)
19. O'Connor, B., Krieger, M., Ahn, D.: Tweetmotif: Exploratory search and topic summarization for twitter. Proc. of ICWSM pp. 2–3 (2010)
20. Paul, M.J., Dredze, M.: You are what you Tweet: Analyzing Twitter for public health. In: Proc. of ICWSM (2011)
21. Stern, H.: On the probability of winning a football game. The American Statistician 45(3), 179–183 (1991)
22. Surowiecki, J.: The Wisdom of Crowds. Anchor (2005)
23. Tumasjan, A., Sprenger, T.O., Sandner, P.G., Welpe, I.M.: Predicting elections with Twitter: What 140 characters reveal about political sentiment. In: Proc. of ICWSM (2010)

24. Vergin, R.C.: Overreaction in the NFL point spread market. Applied Financial Economics 11(5), 497–509 (2001)
25. Zuber, R.A., Gandar, J.M., Bowers, B.D.: Beating the spread: Testing the efficiency of the gambling market for National Football League games. Journal of Political Economy 93(4), 800–806 (1985)

Use of Performance Metrics to Forecast Success in the National Hockey League

Joshua Weissbock, Herna Viktor, and Diana Inkpen

University of Ottawa, Ottawa, Canada
{jweis035, hviktor, Diana.Inkpen}@uottawa.ca

Abstract. Predicting success in hockey is an important area of research which has received little attention in the sports data mining community. We are the first to propose a machine learning approach to forecast success in the National Hockey League. Our approach combines traditional statistics, such as goals for and against, and performance metrics such as possession and luck, in order to build a classification model. We construct several classification models with novel features such as possession and luck in order to build a classification model. Our results indicate that Neural Networks construct the most robust classification models. This confirms the work of earlier researchers, who have also employed Neural Networks in other sports data mining domains. Our results also show the statistics of PDO (which shows, in the short term, the teams playing better or worse than the expected variance) does not aid the prediction.

Keywords: Machine Learning, Hockey, NHL, Classifiers, Neural Networks, Support Vector Machines

1 Introduction

Predicting success in hockey is a subject that has not received much attention compared to other major sports. This may be due to the fact that it is hard to analyze a game of hockey, due to its continuous nature and lack of events (goals). This paper describes a National Hockey League (NHL) Case Study in which we to construct a classification model to predict the outcome of a hockey game. We create classification models to predict success in the National Hockey League (NHL), more specifically, to determine which team is likely to win a game. We use both traditional statistics that are readily available such Goal Differential and Special Teams Success Rate, as well as performance metrics (or "advanced statistics"), used by bloggers and statisticians employed by teams, which have been shown to be more predictive of future success. We further break down these two groups of statistics to see how much they contribute to the success of our classifier.

The rest of this paper is organized as follows. Section 2 provides some background and related research in the field of sports data mining. This is followed, in Section 3, with a discussion of our NHL case study. Section 4 details our experimental approach and results. Section 5 concludes the paper.

2 Background and Related Work

In hockey, five players and a goalie per team are on an ice surface and play for a total of 60 minutes. The goal of the game is to put a rubber puck into the opposing team's net using a 1.5 to 2m long stick made of wood or a composite material. The team who scores the most goals in a game is the winner. In the regular season, if a game is tied after 60 minutes, the teams play an extra 5 minutes of sudden death overtime and after that the game is decided by a shootout. In the playoffs, after 60 minutes, additional 20 minute overtime periods are played until a team scores. As far as the authors are aware, there is no previous work, in the machine learning community, to predict the winner in a hockey game.

Machine learning has been used in other major sports with a varying degree of success to predict the outcome of games, championships and tournaments. Most of the researchers employed neural networks for this task. Chen et al. [1] were among the first to use neural networks for making predictions in sports. They used neural networks to make predictions in greyhound racing and their classifier was shown to be able to make a profit. Huang and Chang [2] used neural networks to make predictions of game winners in the 2006 World Cup and was able to achieve an accuracy of 75%. Purucker [3] used neural networks to make predictions in the National Football League using only four categories he was able to make prediction accuracy of 78.6%. Pardee [4] used neural networks to make predictions for the outcome of the NCAA football bowl games and returned a similar accuracy of 76%. Loeffelholz et al. [5] use neural networks to predict outcomes in the National Basketball Association (NBA) and using common statistics found in the box score of NBA games his model was able to predict with 74.33% accuracy. While neural networks are primarily used in literature, authors have mentioned the use of other classifiers; however, these have not worked as well as neural networks such as [6].

3 National Hockey League Case Study

The NHL is North America's top hockey league comprising of 30 teams: 23 from the United States and 7 from Canada. Teams play a total of 82 games each during the regular season from October to April for a total of 1230 games. There are 2 conferences of 15 teams and each conference is made up of 3 divisions of 5 teams. Within divisions teams play each other 6 times a year, within a conference teams play each other 4 times a year and teams play teams from the other conference 1-2 times a year. At the end of the regular season, the top 8 teams from each conference qualify for the playoffs. The eventual winner wins four best-of-seven series in an elimination tournament and becomes the Stanley Cup Champion.

In our NHL Case Study, data were collected for a period of nearly three months during the 2012-2013 season, for a total of of 517 games between 16 February and 28 April 2013. Due to the lockout this year, this represents about 3/4 of the entire season as teams played 48 games (720 in total). A Python script was created to automate this process, but daily work was required to verify the data and ensure it was collected appropriately. If data were missed, as there is no historical record, it would be difficult to recalculate as it would require iterating through all games and calculating all the statistics.

Goals For	Total number of goals team scored in season (so far).
Goals Against	Total number of goals scored against the team in season (so far).
Goal Differential	Difference between Goals For and Goals Against.
Power Play Success Rate	Ratio where team scored a goal while opposing team had one less man on the ice.
Power Kill Success Rate	Ratio of times team stopped opposing team from scoring while they were down a man due to a penalty.
Shot Percentage	Ratio of goals team scored compared to shots taken.
Save Percentage	Ratio of goals allowed compared to shots stopped by goalie.
Winning Streak	Number of consecutive games won without a loss.
Conference Standings	Team teams current ranking in the standings.
Fenwick Close %	Ratio representing amount of time a team has posession of the puck compared to its opposition.
PDO	Luck, the addition of the teams Sv% and Sh%, over time it regresses to 100%.
5/5 Goals For/Against	Ratio of goals scored by and against team while both teams have 5 players on the ice.

Table 1. All features collected for games.

All statistics collected can be seen in table 1. Recall that some of them were collected before the game, namely traditional and advanced statistics. The traditional statistics are the ones that are easily available from box scores and include Goals For (GF), Goals Against (GA), Goal Differential (GlDiff), Power Play Success Rate (PP%), Power Kill Success Rate (PK%), Shot Percentage (Sh%), Save Percentage (Sv%), Winning Streak and Conference Standing. These were readily available from www.TSN.ca and www.NHL.com. After the game we collected more statistics such as who won and lost, the score, as well as the shots for and against each team. This gives us the power to be able to collect statistics over a smaller subset of games (i.e., the average number of shots against over the last 3 games) instead of seasonal totals and averages. Performance Metrics (or advanced statistics) were also collected before each game. These included Fenwick Close % (a statistic of possession which adds up the total shots, missed shots, and goals for and against); PDO, a statistic of luck; and 5-on-5 Goals For and Against ratio. These statistics are not as readily available and there is no historic record of them. It is only possible to find their current values on websites such as www.behindthenet.ca. Daily work was required to make sure they were collected properly, otherwise the work required to recover their values would be enormous.

Team	Location	Fenwick Close %	Gf	GA	GlDiff	PP%	PK%	Sh%	Sv%	PDO	Win Streak	Conf. Standing	5-5F/A	Class
Toronto	Away	44.92	108	100	8	18.7	85	892	919	1027	2	6	1.05	Win
Ottawa	Home	49.85	89	72	17	29.8	89.4	929	939	1010	3	5	1.12	Loss
Minnesota	Home	48.47	119	126	-7	17.6	80.6	921	911	990	-1	8	0.88	Win
Colorado	Away	46.78	115	149	-34	15.1	80.6	926	909	983	1	15	0.83	Loss
Chicago	Home	55.91	154	99	55	16.9	87	906	928	1022	2	1	1.57	Loss
St. Louis	Away	53.89	126	114	12	19.7	84.5	921	910	989	2	4	1.01	Win

Table 2. Example data for 3 games.

Many of these advanced statistics are just starting to be used by mainstream media and there is evidence that teams are using them to analyze their players; they are also heavily used by bloggers and Internet hockey analysts. They have been shown to be much more predictive of winning, with Fenwick Close having the highest r^2 correlation with points in the standings (0.623 and 0.218 for home and away games) compared to Goals For, Goal Differential, Giveaways, Takeaways, Hits and others [7]. Similarly, looking at the 5-on-5 Goals For/Against ratio, compared to all seasons since 2005, it is founds to have a much higher r^2 correlation with Wins (0.605) and points (0.655) than its traditional statistics counter-parts such as Goals Against / Game, Power Play and Power Kill, and Goals a Game [8].

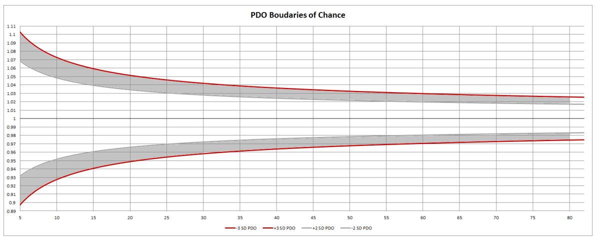


Fig. 1. PDO Boundaries of chance from [9]

Despite its high skill requirement, luck is important to results in the NHL as it makes up 38% of the standings [10]. PDO is an interesting statistic used to analyze hockey. It is not an acronym; rather it is a statistic of luck, luck meaning the results of the gameplay that fall outside of normal boundaries and variance in the players performance [9]. A player cannot maintain a shot percentage that is multiple standard deviations higher or lower than the mean for long periods, nor can a goalie stop every shot that he faces in a season. This is referred to as luck, when the results of the player performance is better (good luck) or worse (bad luck) than the normal average and variance. Hockey seems to be more affected by luck than other major sports due to the low number of events (goals) that happen. A stochastic process can arise that can lead to a goal causing the weaker team to win. Over the long term, luck regresses to the norm, but in the short term you can see which teams have been “luckier” than others. PDO is calculated by the addition of a team’s season Sh% and Sv%. Over time, this will regress to 100%; teams who have a PDO higher than 100% have been lucky, while having a PDO less than 100% means that the team has been performing less than its skill level and are seen as unlucky [11]. Over time, teams regress to the norm; within 25 games PDO will be at $100\% \pm 2\%$ [9]. In the short term, we can see who has been lucky.

4 Experimental Design

WEKA [12], a tool that provides many machine learning algorithms, was used for all classification tasks. Preprocessing of the data was done through the entire period of the data collection, as discussed in Section 3. Daily, the new data was collected and it was ensured that the data were valid[1]. Using a python script, the data were represented as the differential between the statistics of the two teams with the winning team receiving the label "Win" and the losing team receiving the label "Loss". As shown in table 2, the first team’s data would be in the vector V_1 and the second team’s data were in the vector V_2. The Python script calculated $V_1' = V_1 - V_2$ and $V_2' = V_2 - V_1$ and the appropriate Win/Loss labels were attached after[2]. All $517 * 2$ game data vectors were input into WEKA and we used several data mining algorithms. In the first experiment, we looked at how effective traditional, advanced and mixed (both combined) statistics were for predicting success in the NHL. The second part of the experiment further analyzes the “luck” feature to see if can further improve the accuracy.

5 Experimental Results

We consider a binary classification problem in that we want to determine whether a team will win or lose. Using 10-fold cross-validation and a number of WEKA algorithms we input the first three variations of the data sets, namely traditional statistics, advanced statistics and mixed. We used ZeroR to calculate the baseline, this is the results if a classifier were to assign all items to the largest class.

[1] The data sets used in this project are available for future work by others. If you are interested please send a request to the authors by email.

[2] We also modelled the data in a single training example for each game (v.s. one for the win and loss separately) i.e. in the format $V_1 + V_2 + \mathit{label}$ with the label either “HomeWin” or “AwayWin”. The results did not vary from our presented results.

For the others we use WEKA's implementation of a neural network (NN) algorithm (good for noisy data); Naive Bayes (NB) (for a probabilistic approach); SMO, WEKA's Support Vector Machine implementation (as it has shown to do well in previous classification tasks); and, J48, WEKA's C4.5 decision tree implementation (as it produces human readable output). All algorithms were tried with their default WEKA parameters in the first experiment.

	Traditional	Advanced	Mixed
Baseline	50.00%	50.00%	50.00%
SMO	58.61%	54.55%	58.61%
NB	57.25%	54.93%	56.77%
J48	55.42%	50.29%	55.51%
NN	57.06%	52.42%	59.38%

Table 3. Accuracies of the first experiment for 10-fold cross-validation.

The best results were achieved when using Neural Networks on the mixed data, as presented in table 3. All algorithms were first tried with their default parameter values from WEKA. We also explored the statistical significance of our results. Our analyses show that, when using the traditional statistics, all algorithms outperformed the baseline. This was not the case when using advanced statistics. When using both statistics, all algorithms (except J48) constructed models that were more accurate the baseline. With additional tuning, using 5 hidden layers and a momentum of 0.11, the NN classifier produced a model with the highest accuracy (59 : 38%); however, there are no statistically difference between the models built by the NN and SVM classifiers. These values were found by inspection. Further tuning of the other classifiers did not result in higher accuracies. The ROC curve can be seen at figure 2. By splitting the data, 66% for training and the remaining 33% for testing we achieve an accuracy of 57.83%. We made sure that no game was split across the two datasets, that is, the two training examples for a game were both in the training or in the test set. We did error analysis and looked at the automatically classified data to see if any pairs have been labeled Win/Win or Loss/Loss. For the Win/Win or Loss/Lass case, we kept the label with the highest confidence the same and inverted the other label, this increased the overall accuracy of the test data to 59% for 66%-33% training/test data. Ensembler learners were also tried using stacking and voting (with 3 and 5 classifiers using the same subset of algorithms) and the accuracy was similar.

When using the Consistency Subset Evaluation (CfsSubsetEval) feature selection method to find which features contribute the most to the model, we are surprised to see the top three are location (being Home or Away), Goals Against and Goal Differential. We are surprised because previous research indicates that performance metrics are more correlated with long term success in the standings [7, 8]. Our further analysis of the results of the classifiers for the first part can be seen in in table 4. Here we can see the precision, recall, f-score and ROC curve for each classifier using 10-fold cross-validation.

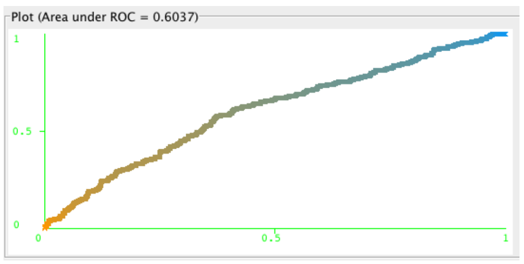


Fig. 2. ROC Curve of tuned Neural Network on the mixed dataset

	Precision	Recall	F-Score	ROC Curve
Baseline	0.5	0.698	0.581	0.5
SMO	0.586	0.586	0.586	0.586
NB	0.567	0.571	0.569	0.588
J48	0.558	0.534	0.545	0.537
NN	0.583	0.594	0.588	0.600

Table 4. Breakdown of each classifier on mixed data for the first experiment using 10-fold cross-validation.

In the second part of our experimental evaluation, we consider the value of PDO in a shortened subset of games. Over the long run, all teams will regress to 100% as players cannot maintain a high (or low) shooting and save percentage for long periods [9]. This is the season value of PDO we used in the first half. We would expect that if we look at the value of PDO over the last n games, we would get a better idea of how lucky a team has been recently, which would be able to help make more accurate predictions. The results of this can be seen in table 5. There does not appear to be any significant change by varying the period of PDO; for the PDOAll and the PDO1 datasets, our statistical significance tests show that all algorithms (except J48) outperform the baseline. This suggests that these performance metrics are not as useful as traditional statistics to predict a single game in the NHL.

5.1 Discussion and Lessons Learned

In the first experiment, we looked at predicting success in the NHL using traditional statistics, performance metrics, and both. After tuning, the best results

	PDO1	PDO3	PDO5	PDO10	PDO25	PDOall
Baseline	50.00%	50.00%	50.00%	50.00%	50.00%	50.00%
SMO	58.61%	58.61%	58.61%	58.61%	58.61%	58.61%
NB	56.38%	56.96%	56.38%	56.58%	56.58%	56.77%
J48	54.93%	55.71%	55.42%	55.90%	55.61%	55.51%
NN	57.64%	56.67%	58.03%	58.03%	57.74%	58.41%

Table 5. Accuracies of the second experiment using 10-fold cross-validation.

come from using Neural Networks with an accuracy of 59.38% (though the difference between NN and SMO was very small). In the long run, it is likely possibly to make a profit off of it. Our results confirm the intuition of earlier researchers to use Neural Networks [1, 2, 13, 3–5]. This choice seems to make the most sense as the algorithm with the best accuracy, as they are known to work well with noisy data such as in sports statistics. What is interesting is that, despite internet hockey analysts showing that performance metrics have a higher correlation with points in the standings, they did not improve our classification rates at all. When we used the Consistency Subset Evaluation feature selection method, it was also interesting to see that the features that added the most value were traditional statistics instead of performance metrics, which have previously shown to be less correlated with points in the standings.

In the second part of our experiments, we considered the predictive power of using a smaller sample size for the PDO statistic of luck. Our results found that using PDO and other performance metrics did not improve our classification results. This seems contrary to hockey analysts who often use statistics such as Fenwick and PDO to predict from mid-season which teams are in playoff standings and will fall (as they tend to have a sub-50% Fenwick with a PDO that is above 100%). The opposite can usually be predicted with teams who have a high Fenwick but a low PDO.

We believe that our model is correct, but we have some future research plans. The first is to aim to increase the accuracy. Other researchers created neural networks to make predictions in other major sports with accuracies in the mid 70s. This may be difficult in hockey, due to luck taking up 38% of the standings. However, we will repeat our experiments in the next hockey season, while collecting additional features and more games, for a large sample size. The additional features that could be tracked that may add value to the classification rate include the number of days of rest between games, time-zone shifts, the affects of long travel, change in altitude, the change in weather, the weather at the arena of the game, gambling odds, injuries on a team, score-adjusted Fenwick, Fenwick when leading or trailing the other team, statistics based on the goal playing, and recent changes in roster are a few that come to mind. Additionally, because of the shortened 2013 season, teams only played 48 games instead of the regular 82. This did not give sufficient time for statistics to regress to the norm and caused surprises in the teams that made it to the playoffs did (such as Toronto with a 44.01% Fenwick-Close), and teams that did not make the playoffs might have (such as New Jersey with a 55.05% Fenwick-Close and a 97.2% PDO). (Note that Toronto was eliminated in the first round of the playoffs.)

6 Conclusion

This paper presented a classifier for the NHL using both traditional, advanced statistics and a mixture. Further examination was given to advanced statistics to see if an improvement in the classifier accuracy could be found.

The best results on our data came from using neural networks with an accuracy of 59.38%. Consistency Subset Evaluation finds that the most important features to the classifier in predicting a single game were location, Goals Against and Goal Differential. While advanced statistics have been shown to make good predictions in the long run (macro scale), traditional stats in this project have performed better in predicting a single game (micro scale).

Future applications of this work would be to aim to predict the winners in the NHL playoffs. Note that the playoffs use four rounds of best-of-seven series, so statistics are more likely to regress to the norm than in a single game. Thus, we are of the opinion that we are more likely to see the better team win. We hypothesize that over a longer series of games you are more likely to see the stochastic processes even out, rather than at the micro scale of a single game, and advanced statistics would show more value. Using a classifier to predict hockey playoff series winner and eventually the Stanley Cup Champion, would be of value to teams as well as to people that bet on games. Another application would be to use betting odds to see if the classifier can make a profit, following the line of thought of Chen et al. [1]. As hockey is somewhat similar to soccer, it would be good to look at machine learning research in soccer for inspiration and see if it would be applicable to hockey.

References

1. Chen, H., Buntin Rinde, P., She, L., Sutjahjo, S., Sommer, C., Neely, D.: Expert prediction, symbolic learning, and neural networks. An experiment on greyhound racing. IEEE Expert **9**(6) (1994) 21–27
2. Huang, K.Y., Chang, W.L.: A neural network method for prediction of 2006 world cup football game. In: Neural Networks (IJCNN), IEEE (2010) 1–8
3. Purucker, M.C.: Neural network quarterbacking. Potentials, IEEE **15**(3) (1996) 9–15
4. Pardee, M.: An artificial neural network approach to college football prediction and ranking. University of Wisconsin (1999)
5. Loeffelholz, B., Bednar, E., Bauer, K.W.: Predicting NBA games using neural networks. Journal of Quantitative Analysis in Sports **5**(1) (2009) 1–15
6. Yang, J.B., Lu, C.H.: Predicting NBA championship by learning from history data. Proceedings of Artificial Intelligence and Machine Learning for Engineering Design (2012)
7. Charron, C.: Breaking news: Puck-possession is important (and nobody told the cbc). http://blogs.thescore.com/nhl/2013/02/25/breaking-news-puck-possession-is-important-and-nobody-told-the-cbc/ (2013) [Online; accessed 12-April-2013].

8. Murphy, B.: Exploring marginal save percentage and if the canucks should trade a goalie. http://www.nucksmisconduct.com/2013/2/13/3987546/exploring-marginal-save-percentage-and-if-the-canucks-should-trade-a (2013) [Online; accessed 12-April-2013].
9. Patrick D: Studying luck & other factors in PDO. http://nhlnumbers.com/2013/1/10/studying-luck-other-factors-in-pdo (2013) [Online; accessed 12-April-2013].
10. Desjardins, G.: Luck in the nhl standings. http://www.arcticicehockey.com/2010/11/22/1826590/luck-in-the-nhl-standings (2010) [Online; accessed 12-April-2013].
11. Charron, C.: PDO explained. http://blogs.thescore.com/nhl/2013/01/21/pdo-explained/ (2013) [Online; accessed 12-April-2013].
12. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann (2005)
13. Young, W.A., Holland, W.S., Weckman, G.R.: Determining hall of fame status for major league baseball using an artificial neural network. Journal of Quantitative Analysis in Sports **4**(4) (2008) 1–44

Finding Similar Movements in Positional Data Streams

Jens Haase[†] and Ulf Brefeld[‡]

Knowledge Mining & Assessment Group
Technische Universität Darmstadt, Germany

[†]`je.haase@gmail.com`
[‡] `brefeld@kma.informatik.tu-darmstadt.de`

Abstract. In this paper, we study the problem of efficiently finding similar movements in positional data streams, given a query trajectory. Our approach is based on a translation-, rotation-, and scale-invariant representation of movements. Near-neighbours given a query trajectory are then efficiently computed using dynamic time warping and locality sensitive hashing. Empirically, we show the efficiency and accuracy of our approach on positional data streams recorded from a real soccer game.

1 Introduction

Team sports has become a major business in many parts of the world. Clubs spend a great deal of money on players, training facilities and other ways to further improve their play. For instance, the German Bundesliga records all games with special cameras, capturing bird's eye views of the pitch, to better analyse player movements and tactics. The recordings capture positions of the players and the ball for every fraction of a second. While simple analyses, such as the overall distance a player covered, heat maps of player positions, etc., can be computed (semi-)automatically, more complex analyses involving tactics and counter-strategies rely on human experts. However, the sheer existence of such data paves the way for automatic analyses using intelligent mining techniques. In this paper, we study efficient techniques for detecting similar movements in positional data streams to provide a basis for the analyses of frequent movements and tactical patterns.

For a soccer game taking about 90 minutes, the recorded data translate into a positional data stream. Standard recording rates of 25 frames per second lead to a representation by about 135,000 snapshots. Every snapshot consists of the 23 positions of the players of the two teams and the ball. In sum, the game is described by more than three million coordinates. As player movements are sequences of such coordinates, it is clear that there are a great deal of comparisons necessary to account for different lengths of such sequences across players. Thus, there is a real need for efficient techniques to further process and analyse the data.

In this paper, we study the problem of finding similar movements of players in such positional data streams. The problem is challenging for two reasons. Firstly, it is not clear how to define an appropriate similarity measure on player trajectories and secondly, the sheer number of coordinates render exact approaches infeasible as we will show in the experiments. We first propose a translation-, rotation-, and scale-invariant representation of movements using Angle/Arc-Lengths [11]. Second, we investigate efficient near-neighbour routines based on dynamic time warping [9] and locality sensitive hashing [2]. Our empirical results on positional data recorded from a real soccer game show the efficiency and accuracy of our approach compared to exact baseline methods.

The remainder is structured as follows. Section 2 briefly reviews related work. Our main contribution is presented in Section 3 and we report on empirical results in Section 4. Section 5 concludes.

2 Related Work

Prior work on mining positional data streams mostly focuses on the performance of individual players. Kang et al. [4] present an approach that uses positional data to assess player positions in particular areas of the pitch, such as catchable, safe or competing zones. Grunz et al. [3] analyse groups of players and their behaviour using self organising maps on positional data. Every neuron of the network represents a certain area of the pitch. Thus, whenever a player moves into such an area, the respective neuron is activated. Perše at al. [8] use positional data of basketball games to compare movements of players with respect to tactical patterns, e.g., a player blocks space for his teammate. The presented approach however does not detect novel movements that deviate from the already known patterns. By contrast, we study a purely data-driven approach to find similar movements in positional data for a given query trajectory without making any assumptions on zones, tasks, or movements.

3 Contribution

3.1 Preliminaries

For each player, we are given a positional data stream $\mathcal{P} = \langle \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots \rangle$ where $\boldsymbol{x}_t = (x_1, x_2)^\top$ denotes the coordinates of the players position on the pitch at time t. A trajectory or movement of the player is a subset $\boldsymbol{p} \subset \mathcal{P}$ of the stream, e.g., $\boldsymbol{p} = \langle \boldsymbol{x}_t, \boldsymbol{x}_{t+1}, \ldots, \boldsymbol{x}_{t+m} \rangle$, where m is the length of the trajectory. A game

$\mathcal{D}$ is thus given by the union of all trajectories of length m of the two teams. For simplicity, we omit the time index t and simply index elements of a trajectory by their offset $1, \ldots, m$ in the remainder. The goal of this paper is to accurately and efficiently compute similarities between trajectories in $\mathcal{D}$. That is, given a query trajectory $\boldsymbol{q}$, we aim at finding the N most similar trajectories in $\mathcal{D}$.

3.2 Representation

We aim to exploiting the symmetries of the pitch and use Angle/Arc-Length (AAL) [11] transformations to guarantee *translation*, *rotation*, and *scale* invariant representations of trajectories. The main idea of AAL is to represent a movement $\boldsymbol{p} = \langle \boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \rangle$ in terms of distances and angles

$$\boldsymbol{p} \mapsto \bar{\boldsymbol{p}} = \langle (\alpha_1, \|\boldsymbol{v}_1\|), \ldots, (\alpha_m, \|\boldsymbol{v}_m\|) \rangle, \tag{1}$$

where $\boldsymbol{v}_i = \boldsymbol{x}_i - \boldsymbol{x}_{i-1}$. The difference $\boldsymbol{v}_i$ is called the *movement vector* at time i and the corresponding angle with respect to a reference vector $\boldsymbol{v}_{ref} = (1, 0)^\top$ is defined as

$$\alpha_i = sign(\boldsymbol{v}_i, \boldsymbol{v}_{ref}) \left[cos^{-1} \left(\frac{\boldsymbol{v}_i^\top \boldsymbol{v}_{ref}}{\|\boldsymbol{v}_i\| \, \|\boldsymbol{v}_{ref}\|} \right) \right],$$

where the $sign$ function computes the direction (clockwise ore counterclockwise) of the movement with respect to the reference. In the remainder, we discard the norms in Equation (1) and represent trajectories by their sequences of angles, $\boldsymbol{p} \mapsto \tilde{\boldsymbol{p}} = \langle \alpha_1, \ldots, \alpha_m \rangle$.

3.3 Dynamic Time Warping

In this section, we propose a distance measure for trajectories. The proposed representation of the previous section fulfils the required invariance in terms of translation, rotation and scaling [11]. However, some movements may start slow and end fast, while others start fast and then slow down at the end. Thus, we additionally need to compensate for phase shifts of trajectories. A remedy comes from the area of speech recognition and is called dynamic time warping (DTW) [9]. Given two sequences $\boldsymbol{s} = \langle s_1, \ldots, s_m \rangle$ and $\boldsymbol{q} = \langle q_1, \ldots, q_m \rangle$ and an element-wise distance function $dist : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ (e.g., Euclidean distance), we define the DTW function g recursively as follows

$$\begin{aligned}
g(\emptyset, \emptyset) &= 0 \\
g(\boldsymbol{s}, \emptyset) &= dist(\emptyset, \boldsymbol{q}) = \infty \\
g(\boldsymbol{s}, \boldsymbol{q}) &= dist(s_1, q_1) + min \left\{ \begin{array}{l} g(\boldsymbol{s}, \langle q_2, \ldots, q_m \rangle) \\ g(\langle s_2, \ldots, s_m \rangle, \boldsymbol{q}) \\ g(\langle s_2, \ldots, s_m \rangle, \langle q_2, \ldots, q_m \rangle) \end{array} \right\}
\end{aligned}$$

The time complexity of DTW is $\mathcal{O}(|\boldsymbol{s}||\boldsymbol{q}|)$ which is clearly intractable for computing similarities of thousands of trajectories. However, recall that we aim at finding the N best matches for a given query. This allows for pruning some DTW computations using lower bounds f, i.e., $f(\boldsymbol{s}, \boldsymbol{q}) \leq g(\boldsymbol{s}, \boldsymbol{q})$, with an appropriate function f that can be more efficiently computed than g [10]. We use two different lower bound functions, f_{kim} [6] and f_{keogh} [5], that are defined as follows: f_{kim} focuses on the first, last, greatest, and smallest values of two sequences [6]

$$f_{kim}(\boldsymbol{s}, \boldsymbol{q}) = \max\left\{|s_1 - q_1|, |s_m - q_m|, |\max(\boldsymbol{s}) - \max(\boldsymbol{q})|, |\min(\boldsymbol{s}) - \min(\boldsymbol{q})|\right\}$$

and can be computed in $\mathcal{O}(m)$. However, the greatest (or smallest) entry in the transformed paths is always close or identical to π (or $-\pi$) and can thus be ignored. Consequentially, the time complexity reduces to $\mathcal{O}(1)$ [10]. The second lower bound f_{keogh} [5] uses minimum ℓ_i and an maximum values u_i for subsequences of the query $\boldsymbol{q}$ given by

$$\ell_i = \min(q_{i-r}, \ldots, q_{i+r}) \quad \text{and} \quad u_i = \max(q_{i-r}, q_{i+r}),$$

where r is a user defined threshold. Trivially, $u_i \geq q_i \geq \ell_i$ holds for all i and the lower bound f_{keogh} is given by

$$f_{keogh}(\boldsymbol{q}, \boldsymbol{s}) = \sqrt{\sum_{i=1}^{m} c_i} \quad \text{with} \quad c_i = \begin{cases} (s_i - u_i)^2 & : if\ s_i > u_i \\ (s_i - \ell_i)^2 & : if\ s_i < \ell_i \\ 0 & : otherwise \end{cases}$$

which can also be computed in $\mathcal{O}(m)$ (see [7] for details).

Algorithm 1 extends [10] to compute the N most similar trajectories for a given query $\boldsymbol{q}$. Lines 2–9 compute the DTW distances of the first N entries in the database and store the entry with the highest distance to $\boldsymbol{q}$. Lines 10–21 loop over all subsequent trajectories in $\mathcal{D}$ by first applying the lower bound functions f_{kim} and f_{keogh} to efficiently filter irrelevant movements before using the exact DTW distance for the remaining candidates. Every trajectory, realising a smaller DTW distance than the current maximum, replaces its peer and the variables $maxdist$ and $maxind$ are updated accordingly. Note that the complexity of Algorithm 1 is linear in the number of trajectories in $\mathcal{D}$. In the worst case, the sequences are sorted in descending order by the DTW distance, which requires to compute all DTW distances. In practice we however observe much lower run-times.

An important factor is the tightness of the lower bound functions. The better the approximation of the DTW the better the pruning. The parameter N plays also a crucial part in the effectiveness of the algorithm. If we set $N = 1$ the

Algorithm 1 TOP_N(N,$\boldsymbol{q}$,$\mathcal{D}$)

Input: number of near-neighbour movements N, query trajectory $\boldsymbol{q}$, game $\mathcal{D}$
Output: The N most similar trajectories to $\boldsymbol{q}$ in $\mathcal{D}$

```
 1: $output = \varnothing \;\wedge\; maxdist = 0 \;\wedge\; maxind = -1$
 2: for $i = 1, \dots, N$ do
 3:     $dist = g(\boldsymbol{q}, \mathcal{D}[i])$
 4:     $output[i] = \mathcal{D}[i]$
 5:     if $dist > maxdist$ then
 6:         $maxdist = dist$
 7:         $maxind = i$
 8:     end if
 9: end for
10: for $i = N+1, \dots, |\mathcal{D}|$ do
11:     if $f_{kim}(\boldsymbol{q}, \mathcal{D}[i]) < maxdist$ then
12:         if $f_{keogh}(\boldsymbol{q}, \mathcal{D}[i]) < maxdist$ then
13:             $dist = g(\boldsymbol{q}, \mathcal{D}[i])$
14:             if $dist < maxdist$ then
15:                 $output[maxind] = \mathcal{D}[i]$
16:                 $maxdist = \max\{g(\boldsymbol{q}, output[j]) : 1 \le j \le N\}$
17:                 $maxind = \arg\max_{1 \le j \le N} g(\boldsymbol{q}, output[j])$
18:             end if
19:         end if
20:     end if
21: end for
```

maximum value will drop faster towards the lowest value in the dataset. By contrast, setting $N = |\mathcal{D}|$ requires to compute the DTW distances for all entries in the database. Hence, in most cases, $N \ll |\mathcal{D}|$ is an appropriate choice to reduce the overall computation time.

3.4 Locality Sensitive Hashing

To further improve the efficiency of our algorithm, we will use locality sensitive hashing (LSH) [2] to remove a great deal of trajectories before processing them with Algorithm 1. The idea of LSH is to hash similar objects to the same bucket, so that all objects of a bucket are considered candidates for being near-neighbours. An interesting equivalence class of LSH functions are distance based hashes (DBH) [1] that can be applied together with arbitrary (e.g., non-metric) distance measures.

To define a hash family for our purposes, we first need to define a function $h : \mathcal{D} \to \mathbb{R}$ that maps a trajectory $\boldsymbol{s} \in \mathcal{D}$ to the set of real numbers. Choosing two randomly drawn members $\boldsymbol{s}_1, \boldsymbol{s}_2 \in \mathcal{D}$ we define the function h as follows:

$$h_{\boldsymbol{s}_1,\boldsymbol{s}_2}(\boldsymbol{s}) = \frac{dist(\boldsymbol{s},\boldsymbol{s}_1)^2 + dist(\boldsymbol{s}_1,\boldsymbol{s}_2)^2 - dist(\boldsymbol{s},\boldsymbol{s}_2)^2}{2\,dist(\boldsymbol{s}_1,\boldsymbol{s}_2)}.$$

In the remainder, we will use the identity $dist(\boldsymbol{s}_1, \boldsymbol{s}_2) = f_{kim}(\boldsymbol{s}_1, \boldsymbol{s}_2)$ for simplicity. To compute a discrete hash value for $\boldsymbol{s}$ we verify whether $h(\boldsymbol{s})$ lies in a certain interval $[t_1, t_2]$,

$$h^{[t_1,t_2]}_{\boldsymbol{s}_1,\boldsymbol{s}_2}(\boldsymbol{s}) = \begin{cases} 1 : h_{\boldsymbol{s}_1,\boldsymbol{s}_2}(\boldsymbol{s}) \in [t_1, t_2] \\ 0 : \quad otherwise \end{cases}$$

Optimally, the interval boundaries t_1 and t_2 are chosen so that the probability that a randomly drawn $\boldsymbol{s} \in \mathcal{X}$ lies with 50% chance within and with 50% chance outside of the interval. The set $\mathcal{T}$ defines the set of admissible intervals,

$$\mathcal{T}(\boldsymbol{s}_1,\boldsymbol{s}_2) = \left\{[t_1,t_2] \;:\; Pr_{\mathcal{D}}(h^{[t_1,t_2]}_{\boldsymbol{s}_1,\boldsymbol{s}_2}(\boldsymbol{s})) = 0) = Pr_{\mathcal{D}}(h^{[t_1,t_2]}_{\boldsymbol{s}_1,\boldsymbol{s}_2}(\boldsymbol{s})) = 1)\right\}.$$

Given h and $\mathcal{T}$ we can now define the DBH hash family that can be directly integrated in standard LSH algorithms:

$$\mathcal{H}_{DBH} = \left\{h^{[t_1,t_2]}_{\boldsymbol{s}_1,\boldsymbol{s}_2} \;:\; \boldsymbol{s}_1,\boldsymbol{s}_2 \in \mathbb{R} \wedge [t_1,t_2] \in \mathcal{T}(\boldsymbol{s}_1,\boldsymbol{s}_2)\right\}$$

Using random draws from $\mathcal{H}_{DBH}$, we construct several hash functions by AND- and OR-concatenation [2]. Given a query trajectory $\boldsymbol{q} \in \mathcal{D}$, the retrieval process first identities candidate objects that are hashed to the same bucket for at least one of the hash functions and computes the exact distances of the remaining candidates using Algorithm 1.

4 Evaluation

In this section, we evaluate our approach in terms of run-time, pruning efficiency, and precision. For our experiments, we use positional data published by the DEBS Grand Challenge in 2013[1]. There are eight players in each team, where every player is equipped with two sensors, one for each shoe. We average these two values to obtain only a single measurement for every player at a time. Discarding additional data that is not useful in our context leaves us with a stream of

$$\text{(sensor/player id, timestamp, player coordinates)}$$

[1] http://www.orgs.ttu.edu/debs2013/index.php?goto=cfchallengedetails

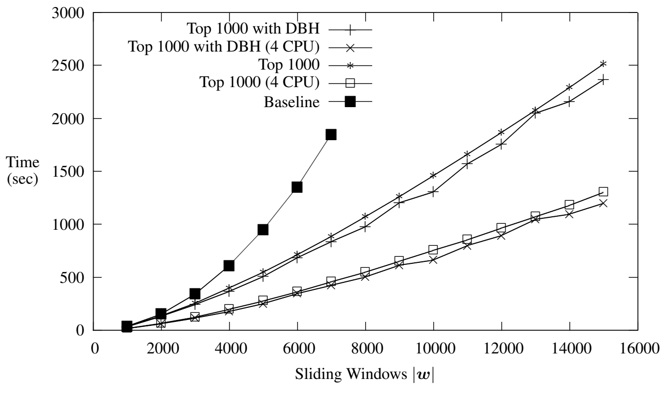


Fig. 1. Run-time.

triplets. Additionally, we discard all data points that have been recorded before or after the game as well as data points that occurred outside of the pitch. We also remove the effects of changing sides after half time by appropriate transformations. Finally, we average the positional information of each player over 100ms to reduce the overall amount of data and use trajectories of size 10.

In our first experiment, we focus on 15,000 consecutive positions of one player, so that we are still able to compare performances to the exact baseline using the DTW distance from Section 3.3. We compute the N-most similar trajectories using Algorithm 1, where $N = 1,000$ and study run-times of the different approaches. Figure 1 (left) shows the results. The computation time of the baseline grows exponentially in the size of the data $\mathcal{D}$. Algorithm 1 performs slightly super-linear and clearly outperforms the baseline. Pre-filtering trajectories using DBH results in only a small speed-up. Adding more CPUs further significantly improves the run-time of the algorithms and indicates that parallelisation of the approach allows for computing near-neighbours for large data sets in only a couple of minutes.

The observed improvements in run-time are the result of a highly efficient pruning strategy. Table 4 shows the amount of trajectories that are pruned for different amounts of data. Note that the DBH pruning depends on the data and not on the ratio $\frac{N}{|\mathcal{D}|}$. The effectiveness of pruning using f_{kim} and f_{keogh} increases with increasing amounts of data for constant N.

We now investigate the accuracy of the proposed approach. We compute the 1000 most similar trajectories for all 35,248 player movements and measure the effect of DBH in terms of the *precision@N*. For every query $\boldsymbol{q}$ we computed the performance for $N \in \{100, 200, \ldots 1000\}$ and averaged the results that are shown in Figure 2. For completeness we also included the worst cases. The

Table 1. Pruning efficiency

trajectories	f_{kim}	f_{keogh}	**DBH**	**Total**
1000	0%	0%	11.42%	11.42%
5000	0.28%	34.00%	16.33%	50.61%
10000	9.79%	41.51%	17.80%	60.10%
15000	17.50%	46.25%	11.82%	75.57%

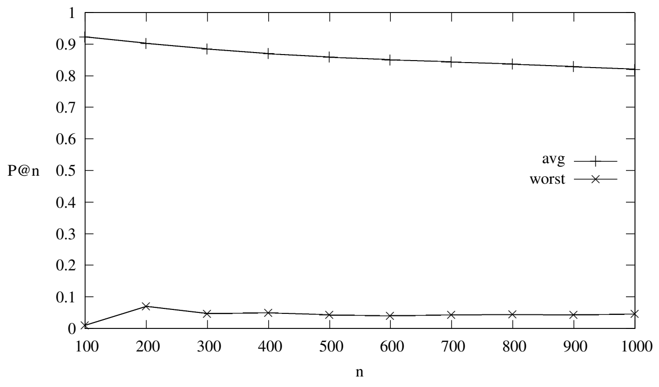


Fig. 2. Accuracy of DBH.

quality of the candidates covers a broad range and the worst cases are clearly inappropriate for accurate computations of similarity. Nevertheless, on average DBH performs well and only slightly decreases in the size of N. Figure 3 shows an exemplary query trajectory (top, left) as well as five similar trajectories found by DBH, where the axes denote the coordinates on the pitch of the respective movement. The retrieved near-duplicates are very close to the query and well suited for further processing.

5 Conclusion

In this paper, we presented an approach to efficiently compute similar movements in positional data streams. Our solution is based on dynamic time warping and distance based hashing. Empirically, we showed the efficiency and accuracy of our approaches. Future work will deal with detecting frequent movements across players.

References

1. V. Athitsos, M. Potamias, P. Papapetrou, and G. Kollios. Nearest neighbor retrieval using distance-based hashing. In *Proceedings of the 2008 IEEE 24th International Conference on*

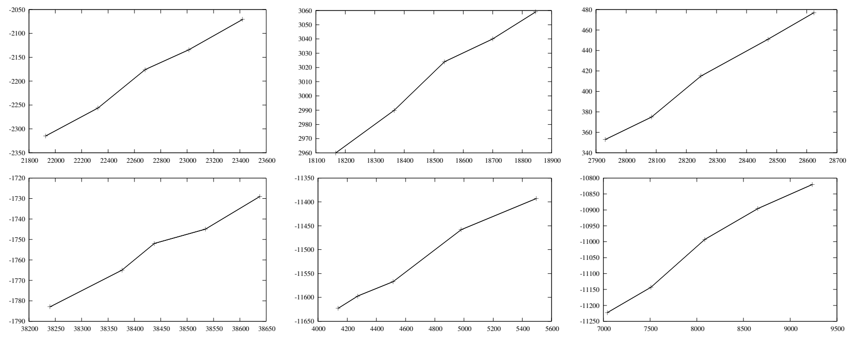


Fig. 3. Exemplary results: The query trajectory is shown in the top-left figure. The other figures show five similar trajectories found by our approach.

Data Engineering, pages 327–336, 2008.

2. A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proceedings of the 25th International Conference on Very Large Data Bases*, pages 518–529, 1999.
3. Andreas Grunz, Daniel Memmert, and Jrgen Perl. Tactical pattern recognition in soccer games by means of special self-organizing maps. *Human Movement Science*, 31(2):334 – 343, 2012. Special issue on Network approaches in complex environments.
4. C.-H. Kang, J.-R. Hwang, and K.-J. Li. Trajectory analysis for soccer players. In *Proceedings of the Sixth IEEE International Conference on Data Mining - Workshops*, pages 377–381, 2006.
5. Eamonn Keogh. Exact indexing of dynamic time warping. In *Proceedings of the 28th international conference on Very Large Data Bases*, pages 406–417, 2002.
6. S.-W. Kim, S. Park, and W. W. Chu. An index-based approach for similarity search supporting time warping in large sequence databases. In *Proceedings of the 17th International Conference on Data Engineering*, pages 607–614, 2001.
7. D. Lemire. Faster retrieval with a two-pass dynamic-time-warping lower bound. *Pattern Recogn.*, 42(9):2169–2180, September 2009.
8. M. Perše, M. Kristan, S. Kovačič, G. Vučkovič, and J. Perš. A trajectory-based analysis of coordinated team activity in a basketball game. *Computer Vision and Image Understanding*, 113(5):612 – 621, 2009.
9. Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.
10. T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh. Searching and mining trillions of time series subsequences under dynamic time warping. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 262–270, 2012.
11. Michail Vlachos, D. Gunopulos, and Gautam Das. Rotation invariant distance measures for trajectories. In *Proceedings of tInternational Conference on Knowledge Discovery and Data Mining*, KDD '04, pages 707–712, 2004.

Comparison of machine learning methods for predicting the recovery time of professional football players after an undiagnosed injury

Stylianos Kampakis, University College London, United Kingdom

stylianos.kampakis@gmail.com

Abstract. Injuries are a common problem in professional football. A challenge that the medical team faces is to successfully predict the recovery time of an injured player. Current medical standards can only give vague predictions as to when a player will return to play. Obviously, making an accurate prediction as soon as possible would be helpful to the coach. This research tries to answer the question of whether it is possible to predict when a player will return to play, based on information at the moment of injury, while also comparing three machine learning methods for this task: support vector machines, Gaussian processes and neural networks. The tests were conducted on data from the professional football club of Tottenham Hotspur. The results demonstrate that this task can be completed with a reasonable amount of accuracy, without any method performing significantly better than the rest. Future directions and possible improvements are discussed.

Keywords: injury prediction, football, support vector machine, neural network, Gaussian process, comparison

1 Introduction

Injuries are a common problem in every sport, including football. Professional football players get injured on average once per year [1] with 10-35 injuries occurring per 1000 game hours [2]. Injuries have been described as the main factor that prevents professional players from not being able to participate in training and playing activities [3].

The factors that cause injuries can vary. A significant percentage of injuries (9%-34%) happens due to overuse [4-5]. Most of the injuries are described as traumatic, with 29% of them being due to foul play [4]. The majority of injuries happen in play, and the most severe cases can be attributed to body contact [5].

As soon as an injury happens it is important to make an estimate of how long the player will need to recover from the injury and get back to play. This information can help the manager make appropriate changes in the squad or the tactical planning of the team. It can also help the director of the club, since new players might need to get signed in order to cover for players who are going to stay out of play for a long time. Additionally, managing the player’s expectations with respect to his injury is important, so that the player can prepare himself mentally and psychologically. Finally, it would help the medical team by providing additional certainty in the predictions of the experts.

Currently, there is no standard method to estimate the time a player will miss from play. The time is estimated based on the experience of the physician and on recommendations by various groups and studies. The suggestions can vary quite significantly with each other, and they can also have significant variance. For example, suggestions for return to play following anterior cruciate ligament reconstruction can range from 16 to 24 weeks [6]. Similar recommendations exist for hamstring injuries [7] and concussions [10-13].

Machine learning has been used in sports for various purposes (e.g. cycling [8] and swimming [9]) including football [16-17]. The complicated and multi-factorial nature of many sports makes machine learning a natural choice for predictive tasks.

The purpose of this study is to compare different machine learning methods on predicting the recovery time of professional football athletes. The goal is to make the prediction based on information available at the time of injury, before an official diagnosis has been conducted. There are three main reasons for which the final diagnosis was left out.

First, diagnoses, in some cases, can take some time, while ideally a coach would like to know as soon as possible how long a player will stay out of play. It would be interesting to see what is the best prediction that can be obtained before an official diagnosis.

Secondly, there are many different diagnoses and different levels of abstraction that can be used. For example, in this study's dataset there were some knee injuries that were described as “knee pain, unspecified”, “patellofemoral pain” and “Left knee medial meniscus”. These diagnoses could be elaborated even further, or they could be abstracted, by classifying them all as “knee injuries”. This is a medical problem that can influence the performance of any machine learning or statistical model that will use this information.

However, it is not entirely clear what degree of elaboration would actually help in the prediction of the response variable. For that reason it is important to know what degree of accuracy can be achieved in the prediction of the response variable before including the diagnosis, so that future research could actually tackle the problem of trying to identify the correct level of abstraction needed for this task.

Thirdly, as part of UEFA guidelines, teams in the premier league have to collect information on every injury that occurs. This information mainly concerns extrinsic factors of an injury (e.g., whether the player was running, whether there was a collision, etc.) and it is easy to collect. A proper model that tries to achieve maximum accuracy on the task of predicting the recovery time obviously requires as much information as possible, like a player's medical exams or training records. However, it would be interesting to see what is the maximum accuracy that can be achieved for this task based solely on extrinsic information. This result could be used to establish a baseline which future, more elaborate models, will improve.

The methods that were chosen for this research were Gaussian processes, support vector machines and neural networks. The reason behind these choices is that all these methods are popular for regression tasks. While there are many other choices for solving regression problems in machine learning, these three methods have been proven and tested in a variety of applications, so they provide sensible choices for approaching this task.

The primary goal of this study was to test the degree to which this task is possible in general by reaching a level of error in the predictions that can have practical applicability, at least in some cases. Once this was established, the next goal was to see whether one of these methods is more suited for this task compared to others. The study itself is part of a greater research project that has as a final goal a fully-working predictive system that can aid football teams. Therefore, future plans, directions and suggestions for research are discussed, as well.

2 Methods

2.1 The dataset

The dataset consists of a list of injuries at Tottenham Hotspur Football Club which were recorded according to the UEFA guidelines over the period 2006-2012. For every injury, a list of variables was collected. These are presented in table 1. Note that the variable "injury" included in the dataset is not a final diagnosis, but a first general estimate such as "muscle strain" or "bone injury".

Table 1. List of variables in the dataset

Parameter	Description
Age	The age of a player
Stage of season	The stage of season (e.g. mid-season or off-season) when the injury occurred
Where	Describes whether the injury took place in the training field or in the game
Phase of play	Describes the exact way that the injury happened (e.g. running or shooting)
Injury	Description of the injury without a specific diagnosis (e.g. bone injury or overuse)
Type	Describes whether the injury was due to overuse or it was an acute injury
Injured side	Describes whether the left or right side was injured
Position	The position of the player (e.g. forward)
Body part injured	Where the player was injured
Reoccurrence	Describes whether the same injury has happened to the same player in the past
Days unavailable	The main variable of interest in our model. It specifies how many days a player stayed out of play after his injury.

All variables, with the exception of “Age” and “Days unavailable” were categorical variables and they were converted to dummy variables in the statistical sense of the term. Therefore, for each value of a categorical variable, a binary variable was created. This gave rise to a dataset that contains 78 variables (including the response variable).

A histogram of the dataset is shown in figure 1. It is evident that most of the injuries are less than 25 days and the histogram is severely skewed. The total number of cases is 152. The mean is 15.5 and the standard deviation 36.039.

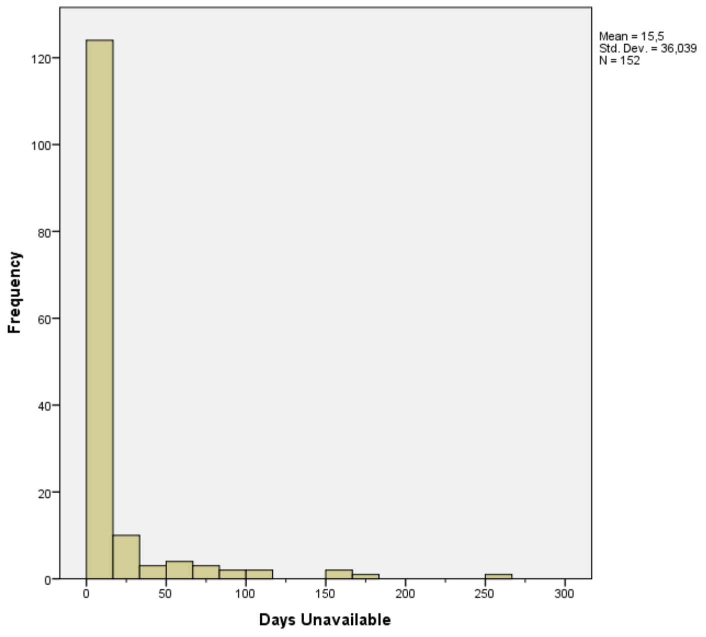


Fig. 1. Histogram of the response variable “Days unavailable”

2.2 Algorithms

Three different methods were used and evaluated: neural networks, support vector machines and Gaussian processes. Each method was executed with many different parameter sets. In order to find the best parameters, grid search was used with some additional manual tuning. Due to the number of tests (more than 50 tests for each method) conducted it is not practical to provide detailed tables and graphs for each parameter set and result.

Tables 2-6 below show the parameters that each method used and their value ranges. For each min-max pair of values 5-25 equidistant steps were used. So for example, for the momentum of the neural network the steps were [0, 0.1, 0.2,…,1]. Once the grid search was done, then some additional manual tweaking was performed.

The neural network was trained using standard backpropagation with momentum.

Table 2. Neural network parameters

	Epochs	Learning Rate	Momentum	Hidden neurons
Min	1500	0.1	0	10
Max	3000	1	1	60

Table 3. SVM parameters, kernel=RBF

	C	Sigma	Epsilon
Min	0	1	0
Max	200	20	2

Table 4. SVM parameters, kernel=polynomial

	C	Degree	Epsilon
Min	0	2	0
Max	200	7	2

Table 5. Gaussian Process parameters, kernel=RBF

	Lengthscale
Min	1
Max	50

Table 6. Gaussian Process parameters, kernel=Laplace

	Lengthscale
Min	1
Max	50

2.3 Evaluation

All the tasks were evaluated using the root mean squared error (RMSE) from the 10-fold cross validation runs of the grid search procedure. The mean of the data was used as a naïve predictor in order to compare the error of the methods against it.

Along with the RMSE the correlation was recorded as well. In the pilot experiments it was observed that, because of the distribution of the data, the error might not always carry a clear picture. In some cases the RMSE would not seem to be significantly better than using the mean as a predictor.

However, careful inspection of individual predictions showed that the RMSE could be severely affected by a few errors. The correlation between the predicted and the actual values is able to provide a scale-free measure of error. The correlation of the naïve predictor with the data is 0. Values above that can provide an additional measure of whether an algorithm can make better predictions than the mean or not.

An example can clarify this a little bit further. An SVM was trained through 10-fold cross-validation on 80% of the data and an RMSE 37.026 was achieved on the test folds. Using the mean of the data as a predictor gives an RMSE of 39.255 which is very close to that achieved through the SVM. However the correlation of SVM's predictions and the true values for the test folds is 0.49, while for the mean it is 0.

The correlation between SVM's predictions on the 20% of the data (41 points) that were not used in the training and the true values is 0.592 and the RMSE 27.74. This relationship is depicted in the scatterplot in figure 2.

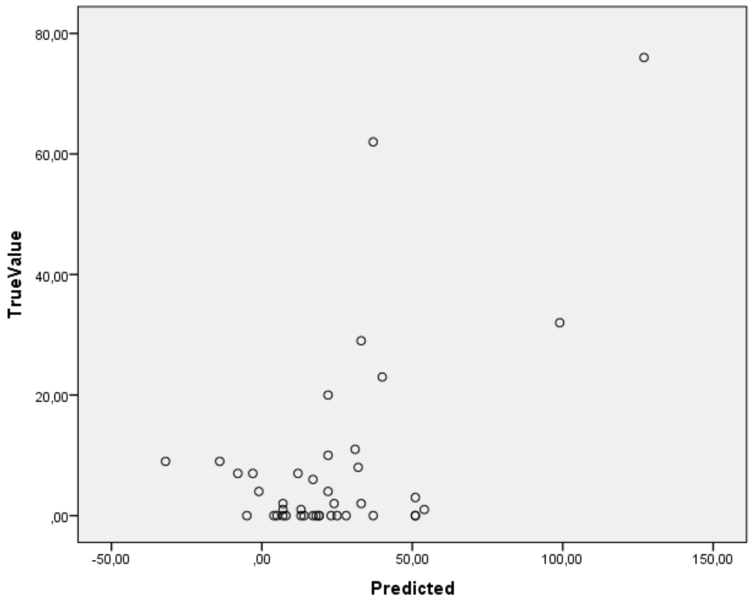


Fig. 2. Scatterplot of true values versus predicted values

For practical applications we can assume that all negative values are effectively 0. If we take that into account then the correlation rises to 0.638 and the RMSE drops to 25.6.

By using the mean as a predictor the correlation is obviously still 0 for this data, but the RMSE is 17.5. If we based our conclusions solely on the RMSE, then we'd assume that the SVM is not performing significantly better. However, the difference between the amended predictions for the SVM and the predictions used for the mean can be clearly seen in the scatterplot in figure 3.

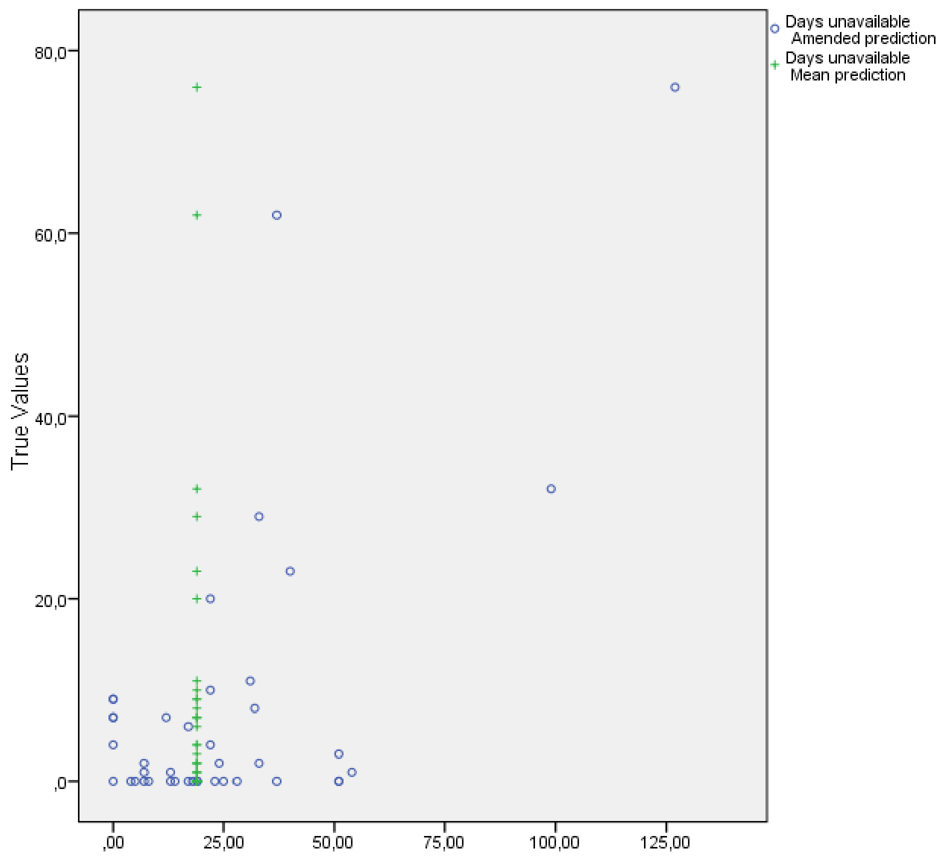


Fig. 3. Scatterplot of true values versus amended predicted values and predictions through the mean

All methods were evaluated using 10-fold cross validation and all tests were executed using RapidMiner version 5.3. The main criterion for choosing the best parameters for each algorithm was the RMSE achieved on the test folds of the cross-validation. The correlation was taken into account when some manual tweaking of the parameters was performed, once the grid search was over.

3 Results

The best results parameter settings achieved for each method are shown in table 7. These parameter settings where evaluated again for each classifier by running 20 rounds of 10-fold cross validation. The RMSEs and correlations for all runs were averaged and they are reported along with their corresponding standard deviations in table 7.

The total RMSE is the error on the whole dataset once the full model has been built.

Table 7. Results for each method

Method	Parameters	RMSE (test)	Correlation (test)	RMSE (total)
SVM	Polynomial kernel, degree=3, C=71, epsilon=1	31.8717 +/- 0.882	0.3565+/- 0.066	4.899
Gaussian Process	RBF kernel, lengthscale=7	32.098+/- 0.846	0.3934+/- 0.0411	5.795
Neural Network	Neurons=45, epochs=2500 learning rate=0.4, momentum=0.2	32.585 +/- 2.02	0.3607+/- 0.0672	1.303

Using the mean as a predictor the RMSE obtained was 35.92. The Wilcoxon signed rank test was conducted for the test RMSE of each classifier. The goal was to check if the error is significantly lower than the error achieved by just using the mean as a predictor. The p-value was less than 0.01 for all three classifiers.

The p-value for a test that the correlation of the classifiers is 0 had a p-value less than 0.01 for all three classifiers.

4 Discussion

It is evident that this task can be predicted with some degree of accuracy, albeit small. The means of the errors and their variances indicate that no single method seems to perform significantly better to others. However, the important point is that some estimate can be obtained. The extrinsic information that is collected for an injury seems to be useful, even to a small extent, when predicting the recovery time of players after an injury.

The results become more important when considering that the size of the dataset is small for this task and it concerns only a single football club. There are many types of injuries in football that can occur under different circumstances. Future research should use datasets from other football clubs in order to verify and expand the current results. Ideally, datasets from football clubs from different countries should be obtained, since the style of play in each country, along with other factors (e.g. a country's climate), could influence the response variable.

Obviously, the end goal is the practical applicability of the results. An issue with the evaluation of the results is the desired degree of accuracy that is required for a method in this task to be considered successful from the perspective of practical applicability. Football teams play a certain amount of games within a season. Usually this is 4 league games per month, and maybe some more cup games and games in European competitions. If a player is injured in a game, it might not matter so much whether he will be back in play in 3 or 5 days, as long as the coach knows that in 7 days, when the next game starts, he will be ready to play.

Furthermore, the dataset contains many transient injuries. The meaning of the word transient is vague and its interpretation is better left to a medical professional, but in general it describes injuries where the recovery time was 0 days or close to that value (e.g 2 days). Many of these cases do not require the execution of a predictive algorithm, because the medical professionals of the team can very quickly classify the injury as transient. Predictions are more helpful for injuries that have longer lasting effects, for example, more than a couple of weeks. This means, that the margin of error can be higher. If the medical staff's opinion is that a player will miss 5 to 10 weeks, then a prediction that manages to narrow down this margin to, for example, 6 to 7 weeks, can help the coach make better decisions and plan for the future.

An interesting feature of this task is that the models could be included in a diagnostic protocol. After each injury, the medical staff will conduct detailed medical tests in order to diagnose the injury. Models like the ones described in this paper could accompany a diagnosis, providing some additional support for the experts' estimates.

Furthermore, additional information that could be available at the moment of injury includes anthropometric and medical information such as the height, weight or medical blood tests of players. This information could improve the accuracy of the model, while also staying true to its original goal of making predictions right after an injury has occurred.

Finally, future research could also solve the problem of how additional official diagnostic information could be used alongside this model in order to make more accurate predictions.

5 Conclusion

This research dealt with the question of whether it is possible to predict the recovery time after an injury in professional football without an official diagnosis, while it also tests 3 methods against each other for this task. The results demonstrate that it is possible to reach some degree of accuracy in this task, but the size of the dataset, and maybe the variables themselves, limit the accuracy that can be reached. No single method was deemed to be significantly better than any of the other methods that were used.

However, this work paves the way for future research that can include bigger and more complicated datasets and can also be extended by protocols that can combine experts' opinions. Future research will built on top of the current results in order to provide a functional system for assessing injuries in professional football.

6 Acknowledgments

The author of this research would like to thank Wayne Diesel of Tottenham Hotspur Football Club for kindly sharing the data for conducting this research.

Bibliography

1. A. Junge and J. Dvorak, "Soccer injuries: a review on incidence and prevention," *Sports Medicine,* vol. 34, no. 13, pp. 929-938, 2004.
2. J. Dvorak and A. Junge, "Football injuries and physical symptoms: a review of the literature," *The American Journal of Sports and Medicine,* vol. 28, no. 5, 2000.
3. L. Parry and B. Drust, "Is injury the major cause of elite soccer players being unavailable to train and play during the competitive season?," *Physical Therapy in Sport,* vol. 7, no. 2, pp. 58-64, 2006.
4. R. D. Hawkins and C. W. Fuller, "Risk assessment in professional football: an examination of accidents and incidents in the 1994 World Cup finals," *British Journal of Sports Medicine,* vol. 30, no. 2, pp. 165-170, 1996.
5. L. Peterson, A. Junge, J. Chomiak, T. Graf-Baumann and J. Dvorak, "Incidence of football injuries and complaints in different age groups and skill-level groups," *The American Journal of Sports Medicine*, vol. 28, no. 5, 2000.
6. M. Bizzini, D. Hancock and F. Impellizzeri, "Suggestions from the field for return to sports participation following anterior cruciate ligament reconstruction: soccer," *The Journal of Orthepaedic and Sports Physical Therapy,* vol. 42, no. 4, 2012.
7. J. Mendiguchia and M. Brughelli, "A return-to-sport algorithm for acute hamstring injuries," *Physical Therapy in Sport,* vol. 12, no. 1, 2011.
8. B. Ofoghi, J. Zeleznikow, C. MacMahon and D. Dwyer, "Supporting athlete selection and strategic planning in track cycling omnium: A statistical and machine learning approach," *Information Sciences,* vol. 233, pp. 200-213, 2013.
9. E. Meżyk and O. Unold, "Machine learning approach to model sport training," *Computers in Human Behavior,* vol. 27, no. 5, p. 1499–1506, 2011.
10. A. Arnason, A. Gudmundsson, H. A. Dahl and E. Jóhannsson, "Soccer injuries in Iceland," *Scandinavian Journal of Sports & Medicine in Sports,* vol. 6, no. 1, pp. 40-45, 1996.
11. A. B. Nielsen and J. Yde, "Epidemiology and traumatology of injuries in soccer," *The American Journal of Sports Medicine,* vol. 17, no. 6, pp. 803-807, 1989.
12. R. C. Cantu, "Guidelines for return to contact sports after a cerebral concussion," *Physician and Sports Medicine,* vol. 14, no. 10, pp. 75-83, 1986.
13. G. Dicker, "A sports doctor's dilemma in concussion," *Sports Medicine Training and Rehabilitation,* vol. 2, pp. 203-209, 1991.
14. M. Lovell, M. Collins and J. Bradley, "Return to play following sports-related concussion," *Clinics in Sports Medicine,* vol. 23, pp. 421-441, 2004.
15. M. W. Collins, M. R. Lovell and D. B. McKeag, "Current issues in managing

sports concussion," *The Journal of the American Medical Association,* vol. 282, pp. 2283 - 2285, 1999.

16. A. Joseph, N. E. Fenton and M. Neil, "Predicting football results using Bayesian nets and other machine learning techniques," *Knowledge-Based Systems,* vol. 19, no. 7, pp. 544-553.

17. B. Min, J. Kim, C. Choe, H. Eom and B. R. I. McKay, "A compound framework for sports results prediction: A football case study," *Knowledge-Based Systems,* vol. 21, no. 7, 2008.

18. S. Marsland, Machine learning: an algorithmic perspective, CRC Press, 2009.

19. A. Junge and J. Dvorak, "Influece of definition and data collection on the incidence of injuries in in football," *The American Journal of Sports Medicine,* vol. 28, no. 5, 2000.

Predicting NCAAB match outcomes using ML techniques – some results and lessons learned

Zifan Shi, Sruthi Moorthy, Albrecht Zimmermann*

KU Leuven, Belgium

Abstract. Most existing work on predicting NCAAB matches has been developed in a statistical context. Trusting the capabilities of ML techniques, particularly classification learners, to uncover the importance of features and learn their relationships, we evaluated a number of different paradigms on this task. In this paper, we summarize our work, pointing out that attributes seem to be more important than models, and that there seems to be an upper limit to predictive quality.

1 Introduction

Predicting the outcome of contests in organized sports can be attractive for a number of reasons such as betting on those outcomes, whether in organized sports betting or informally with colleagues and friends, or simply to stimulate conversations about who "should have won". We would assume that this task is easier in professional leagues, such as Major League Baseball (MLB), the National Basketball Association (NBA), or the National Football Association (NFL), since there are only relatively few teams and their quality does not vary too widely. As an effect of this, match statistics should be meaningful early on since the competition is strong, and teams play the same opponents frequently. Additionally, professional leagues typically play more matches per team per season, e.g. 82 in the NBA or 162 in MLB, than in college or amateur leagues in which the sport in question is (supposed to be) only a side aspect of athletes' lives.

National College Athletics Association Basketball (NCAAB) matches therefore offer a challenging setting for predictive learning: more than 300 teams that have strongly diverging resource bases in terms of money, facilities, and national exposure and therefore attractiveness for high quality players, play about 30 games each per season, can choose many of their opponents themselves (another difference to professional teams), and often have little consistency in the composition of teams from one season to the next since especially star players will quickly move on to professional sports. Lopsided results and unrealistic match statistics will therefore not be uncommon, distorting the perception of teams' quality.

Most of the existing work in the field is more or less statistical in nature, with much of the work developed in blog posts or web columns. Many problems

* albrecht.zimmermann@cs.kuleuven.be

that can be addressed by statistical methods also offer themselves up as Machine Learning settings, with the expected advantage that the burden of specifying the particulars of the model shifts from a statistician to the algorithm. Yet so far there is relatively little such work in the ML literature. The main goal of the work reported in this paper was therefore to assess the usefulness of classifier learning for the purpose of predicting the outcome of individual NCAAB matches. Several results of this work were somewhat unexpected to us:

- Multi-layer perceptrons, an ML technique that is currently not seeing widespread use, proved to be most effective in the explored settings.
- Explicitly modeling the differences between teams' attributes *does not* improve predictive accuracy.
- Most interestingly, there seems to be a "glass ceiling" of about 74% predictive accuracy that cannot be exceeded by ML or statistical techniques.

2 Definitions

The most straight-forward way of describing basketball teams in such a way that success in a match can be predicted relate to scoring points – either scoring points offensively or preventing the opponent's scoring defensively. Relatively easy to measure offensive statistics include field goals made (FGM), three-point shots made (3FGM), free throws after fouls (FT), offensive rebounds that provide an additional attempt at scoring (OR), but also turnovers that deprive a team of an opportunity to score (TO). Defensively speaking, there are defensive rebounds that end the opponent's possession and give a team control of the ball (DR), steals that have the same effect and make up part of the opponent's turnovers (STL), and blocks, which prevent the opponent from scoring (BLK). And of course, there are points per game (PPG) and points allowed per game (PAG).

The problem with these statistics is that they are all raw numbers, which limits their expressiveness. If a team collects 30 rebounds in total during a game, we cannot know whether to consider this a good result unless we know how many rebounds were there to be had in the first place. 30 of 40 is obviously a better rebound rate than 30 of 60. Similar statements can be made for field goals and free throws, which is why statistics like offensive rebound rate (ORR), turnover rate (TOR), or field goals attempted (FGA) will paint a better picture. Even in that case, however, such statistics are not normalized: 40 rebounds in a game in which both teams combined to shoot 100 times at the basket is different from 40 rebounds when there were only 80 scoring attempts.

For normalization, one can calculate the number of possessions in a given game:

$$Possessions = 0.96 * (FGA - OR - TO + (0.475 * FTA))$$

and normalize teams' points scored and allowed per 100 possessions, deriving offensive and defensive *efficiencies*:

$$OE = \frac{Points\ scored * 100}{Possessions}, DE = \frac{Points\ allowed * 100}{Possessions}$$

It should be noted that the factor 0.475 is empirically estimated – when first introducing the above formulation for the NBA, Dean Oliver estimated the factor as 0.4 [6].

Dean Oliver has also singled out four statistics as being of particular relevance for a team's success, the so-called "Four Factors" (in order of importance, with their relative weight in parentheses):

1. Effective field goal percentage (0.4):
$$eFG\% = \frac{FGM + 0.5 \cdot 3FGM}{FGA}$$
2. Turnover percentage (0.25):
$$TO\% = \frac{TO}{Possessions}$$
3. Offensive Rebound Percentage (0.2):
$$OR\% = \frac{OR}{(OR + DR_{Opponent})}$$
4. Free throw rate (0.15):
$$FTR = \frac{FTA}{FGA}$$

While such statistics are normalized w.r.t. the "pace" of a game, they do not take the opponent's quality into account, which can be of particular importance in the college game: a team that puts up impressive offensive statistics against (an) opponent(s) that is (are) weak defensively, should be considered less good than a team that can deliver similar statistics against better-defending opponents. For best expected performance, one should therefore normalize w.r.t. pace, opponent's level, and national average, deriving *adjusted* efficiencies:

$$AdjOE = \frac{OE * avg_{all\ teams}(OE)}{AdjDE_{opponent}}, AdjDE = \frac{DE * avg_{all\ teams}(DE)}{AdjOE_{opponent}}$$

To gain a comprehensive picture of a team's performance during the season, such statistics would have to be averaged over all games (we describe two approaches for doing so in Section 4.2), and a state-of-the-art way of using the derived statistics in predicting match outcomes consists of using the so-called Pythagorean Expectation, e.g.:

$$Win\ Probability = \frac{((Adjusted)\ OE_{avg})^y}{((Adjusted)\ OE_{avg})^y + ((Adjusted)\ DE_{avg})^y}$$

to calculate each team's win probability and predicting that the team with the higher probability wins. More generally, *ranking systems* can by used by ranking the entire pool of teams and predicting for each match-up that the higher ranked team wins.

3 Related Work

The use of the Pythagorean Expectation actually goes back to Bill James' work on baseball. It was adapted for the use in basketball prediction by numerous analysts, including such luminaries as Daryl Morey, John Hollinger, Ken Pomeroy, and Dean Oliver. The difference between the different approaches comes down to which measures of offensive and defensive prowess are used and how the exponent has been estimated. Dean Oliver was also the one who first introduced possession-based analysis formally in his book "Basketball on Paper" [6], although he acknowledges that he had seen different coaches use such analysis in practice. In the same work, he introduced the "Four Factors".

The adjustment of efficiencies to the opponent's quality is due to Ken Pomeroy who uses them as input in his version of the Pythagorean Expectation to rank NCAAB teams and predict match outcomes. His is far from the only ranking system, however, with other analysts like Jeff Sagarin, Ken Massey or Raymond Cheung running their own web sites and giving their own predictions. Comparisons of the results of different ranking systems can for instance be found at `http://masseyratings.com/cb/compare.htm` or `http://www.raymondcheong.com/rankings/perf13.html`. The worst accuracy for those systems is in the $62\% - 64\%$ range, equivalent to predicting that the home team wins, the best ones achieve up to $74\% - 75\%$.

The NCAAB itself uses the so-called Ratings Percentage Index to rank teams, a linear weighted sum of a team's winning percentage, its opponents' winning percentage, and the winning percentage of those opponents' opponents.

As an alternative approach, Kvam *et al.* have proposed a logistic regression/Markov chain model [5]. In this method, each team is represented as a state in a Markov chain and state transitions occur if one team is considered better than its opponent. Logistic regression is used to estimate transition probability parameters from the data. The authors have proposed an updated version using Bayesian estimates [3], and recently published work in which they estimate their method's success in comparison to other ranking schemes [2].

4 Day-by-day predictions using ML

The approaches described in the preceding section are in many cases somewhat or even fully hand-crafted. This can be rather high-level, as in *defining* the transition probabilities in LRMC's Markov chain by hand, or it can go as far as Ken Pomeroy taking home court advantage into consideration by *multiplying* the home team's stats by 1.014. Furthermore, especially the Pythagorean Expectation seems to be a rather simple model.

Machine Learning promises to address both of these issues: we would expect to be able to *learn* the relative importance of different descriptive measures, in particular if this importance changes for different numerical ranges, and to be able to *learn* their relationships, automatically making the model as difficult (or simple) as needed. We therefore turned to classification learners representing several different paradigms and evaluated their performance.

In a reversal of current practice, explicit prediction of match outcomes could be used to rank teams by predicting the outcome of all hypothetical pairings and ranking teams by number of predicted wins.

The evaluated learners were:

- Decision trees, represented by C4.5.
- Rule learners, represented by Ripper.
- Artificial neural networks, represented by a Multi-layer Perceptron (MLP).
- Naïve Bayes
- Ensemble learners, by a random forest.

All algorithms were used in the form of their respective Weka implementations and run with default parameter settings, with the exception of Naïve Bayes, for which the "Kernel Estimator" option was activated to enable it to handle numerical attributes effectively, J48, whose pre-pruning threshold we set to 1% of the training data, and the Random Forest, which we set to consist of 20 trees. All data has been downloaded from Ken Pomeroy's web site, `kenpom.com`, and we limit ourselves to matches involving two Division I teams. Matches were encoded by location (home, away, neutral court), the chosen numerical statistics up to the day the match was played, and the outcome (win, loss) from the perspective of the first team. We always chose the team with the lexicographically smaller name as first team. For each experiment run, one season was used as test set and the preceding seasons from 2008 onward as training data, leading to the training and test set sizes shown in Table 1.

Season	2009	2010	2011	2012	2013
Train	5265	10601	15990	21373	26772
Test	5336	5389	5383	5399	5464

Table 1. Training and test set sizes per season

4.1 Seasonal Averaging

Ken Pomeroy's web site features only the most recent averaged adjusted efficiencies (and averaged Four Factors), i.e. from the end of the season for completed seasons, and for seasons in progress the efficiencies up to the current date. We therefore calculated the day-to-day averaged adjusted efficiencies ourselves, following Pomeroy's description. While that description is very precise for the most part, the averaging is summarized as averaging over the season with more weight given to recent games. We chose to average via two methods:

1. an adjustable weight parameter α:

$$AdjE_{avg,post-match} = (1-\alpha)AdjE_{avg,pre-match} + \alpha AdjE_{post-match}$$

and evaluated a number of different alpha values. Both averaged efficiencies and Four Factors stabilized for $\alpha = 0.2$. To have a pre-match value for the first game of the season, we used the preceding season's end-of-season efficiencies, and

2. explicitly:
   A side-effect of using an α-parameter less than 0.5 (e.g. 0.2) in averaging is that last season's end-of-season averaged adjusted efficiency is weighted rather highly since it is the only value whose weight is never multiplied with α itself but always with $(1-\alpha)$. We therefore evaluated a different weighting scheme in which each match's adjusted efficiency is weighted explicitly with the number of games played +1. This means that last season's end-of-season efficiency has weight one, the adjusted efficiency of the first game weight two etc. The sum is normalized with the total sum of weights up to the current date.

We have to admit that using either way, we did not manage to arrive at the same end-of-season efficiencies as Ken Pomeroy. Typically, our values are more extreme, with adjusted offensive efficiencies higher and adjusted defensive efficiencies lower than Pomeroy's values. Also, since α-weighting performed consistently worse, we will focus on the explicit averaging for the rest of the paper.

4.2 Using adjusted efficiencies

In the first set of experiments, we aimed to identify which attributes out of the full set of raw statistics, normalized statistics, Four Factors, and adjusted efficiencies were most useful in predicting match outcomes. We found the combinations of location and adjusted offensive and defensive efficiencies, and location and Four Factors to work best. This result is supported by the outcome of using Weka's feature selection methods to winnow the attribute set down, which select location first, followed by adjusted efficiencies, and the Four Factors.

A somewhat surprising result is the weak performance of the symbolic classifiers: MLP and Naïve Bayes give consistently best results (Table 2). We also see that more training data does not translate into better models, and that 2012 seems to have been an outlier season.

Season	J48	RF	NB	MLP
2009	0.6839	0.6885	0.7101	0.7077
2010	0.6899	0.6942	0.7172	0.7251
2011	0.6905	0.6779	0.7028	0.716
2012	0.7042	0.7137	0.7276	0.7446
2013	0.6898	0.6881	0.7193	0.7215

Table 2. Match outcome prediction accuracies using adjusted efficiencies

Season	J48	RF	NB	MLP
2009	0.6647	0.6801	0.7121	0.7011
2010	0.6645	0.6931	0.7202	0.7165
2011	0.6622	0.6983	0.7206	0.7121
2012	0.6788	0.702	0.7305	0.7311
2013	0.6508	0.6892	0.7081	0.7092

Table 3. Match outcome prediction accuracies using adjusted four factor

The accuracies for the different seasons are on par with those of the best-performing predictive systems, e.g. Ken Pomeroy's predictions and the LRMC, but unfortunately they are not better.

4.3 Using adjusted Four Factors

As mentioned in Section 3, Dean Oliver proposed the so-called "Four Factors" as being influential for a team's success. Since our experiments had indicated that the unadjusted Four Factors were already as useful in predicting match outcomes as adjusted efficiencies, we assumed that adjusted Four Factors should be more effective. We therefore performed adjusting in the same way as for efficiencies: multiplying with the national average and dividing by the opponent's counter-statistic, averaging using both methods. Averaging using α proved again to be worse, while explicitly averaging lead to similar yet slightly worse results compared to using adjusted efficiencies, as Table 3 shows.

In a bid to improve the performance of the symbolic classifiers, we also experimented with encoding the differences between adjusted Four Factors explicitly, hypothesizing that for instance C4.5's over-fitting had to do with inducing branches for many different combinations of values that could be summarized by their difference. We either subtracted a team's defensive factor from the opponent's corresponding offensive factor, or subtracted offensive from corresponding offensive, and defensive from corresponding defensive factors. The former scheme severely underperformed, while the latter scheme with explicit weights for averaging showed very similar results to Table 3.

Finally, we attempted to address our more extreme adjusted values by calculating each season from stretch, not using the preceding season's values as input for the first game. While the resulting adjusted efficiencies are closer to those reported on Pomeroy's web site, prediction accuracies also decrease slightly.

4.4 Development of predictive accuracy as the season progresses

Figure 1 shows how predictive accuracy develop as the season progresses. We chose MLP with adjusted efficiencies for this plot but the general trend is representative for other settings.

With the exception for 2009, when only training data from 2008 was available, predictive accuracy is 100% or close to it for the first few days of the season and then experiences a dip before it recovers, and shows only slight deterioration for the rest of the season. Interesting, but hard to spot in the plot, is that there are small up-and-downs in the playoffs, particularly in the last rounds, for instance predicting the semi-finals and final correctly after getting the quarter-finals wrong.

5 Lessons learned and open questions

In this work, we have explored the use of ML techniques, specifically classification learners, for making NCAAB match outcome predictions. These are just

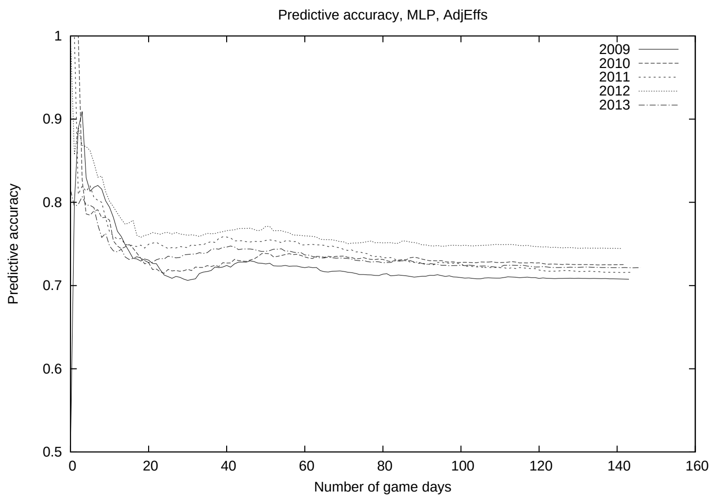


Fig. 1. Development of predictive accuracy over the course of a season (MLP, AdjEff)

preliminary steps and the exploration is obviously far from complete. While the results were somewhat disappointing, we want to stress that they were not bad per se – being on par with the state-of-the-art is only disappointing since we aimed to improve on it. Given our results, however, we believe that there are two first lessons that can be learned and that should guide our next steps.

5.1 It's in the attributes, not in the models

As stated above, one of our expectations was that more complex models could tease out relationships that simpler models would miss. Instead, we found that Naïve Bayes, arguably the simplest of the classifiers, performs remarkably well. Similar observations can actually be made about existing techniques, since Ken Pomeroy's straight-forward Pythagorean Expectation performs as well as, or even better than, the much more complex LRMC model, Brown *et al.*'s claims notwithstanding.

Instead, whatever differences in performance we have observed essentially came down to the used attributes and how they were calculated: adjusted efficiencies and (adjusted) Four Factors are validated both by feature selection techniques and by the success of the classifiers trained on those representations but different ways of averaging over the season have an effect on the quality. Using other or additional features, on the other hand, leads to worse results. In a sense, this should not be surprising: any given match will be won by the team that scored more points than the other one, which is the information encoded in the adjusted efficiencies, for instance.

Of course there is also ML/DM conventional wisdom that the main aspect of using such techniques effectively consists of constructing the right representation. Still, we found it surprising how stark the influence of choosing the right attributes was on achieving best results.

5.2 There seems to be a glass ceiling

Which brings us to the second lesson: the other invariant that we saw in our experiments is that there seems to be an upper limit to predictive accuracy for match outcomes, at around 74% − 75%. This holds not only for Naïve Bayes and the MLP, but when one considers comparisons of non-ML methods, e.g. `http://www.raymondcheong.com/rankings/perf13.html` or [2], one finds similar results. Additionally there are works in fields such a soccer [4] (76.9%), American Football [8] (78.6%), NCAA Football [7] (76.2%), and the NBA [1] (74.33%) that show best results in a similar region.

It is difficult to determine why this is the case. If the claim made in the preceding section holds and the performance of predictors comes down to attribute construction, then maybe this glass ceiling is an artifact of the attributes we and others use. It is also possible, however, that there is simply a relatively large residue of college basketball matches that is in the truest sense of the world unpredictable.

5.3 Where to next?

First off, there is need to verify that our first lesson is correct and attributes are indeed what make or break success. To this end, different feature selection and modeling techniques need to be contrasted to get a clear understanding of attributes' effects, and how to best aggregate them over the course of a season. Following (or parallel to) that, both of the possible explanations for the glass ceiling given above offer themselves up for exploration that we intend to pursue in the near future:

1) Most existing attributes do not encode so-called "intangibles" such as experience, leadership, or luck. Attempts have been made to construct objective indicators, as in `http://harvardsportsanalysis.wordpress.com/2012/03/14/survival-of-the-fittest-a-new-model-for-ncaa-tournament-prediction/`, whose author proposes a "Returning Minutes Percentage", Dean Oliver's attempts to measure positional stability, or Ken Pomeroy's work that takes the luck of teams into account. Pomeroy incidentally credits Dean Oliver (once again) with having introduced this into basketball analysis. Hence, constructing new attributes that include additional information could improve predictive power.

2) A better understanding of incorrectly predicted matches is necessary. The weak performance of ensembles indicates that misclassified matches are not easily modeled. However, identifying similarities of misclassified matches or learning a model that can discriminate correctly and incorrectly classified instances, would help in gaining an understanding whether those matches are different or simply

unpredictable. At this point, we would also finally come back to whether we can determine which team "should have won".

Finally, somewhat unrelated, it could be interesting to separate training data by conference and learn models particular to the involvement of certain conferences teams. The most challenging question would probably have to do with how to decide which model's prediction to use if the two models disagree.

References

1. Loeffelholz Bernard, Bednar Earl, and Bauer Kenneth W. Predicting nba games using neural networks. *Journal of Quantitative Analysis in Sports*, 5(1):1–17, January 2009.
2. Mark Brown, Paul Kvam, George Nemhauser, and Joel Sokol. Insights from the LRMC method for NCAA tournament predictions. In *MIT Sloan Sports Conference*, March 2012.
3. Mark Brown and Joel Sokol. An improved LRMC method for NCAA basketball predictions. *Journal of Quantitative Analysis in Sports*, 6(3), 2010.
4. Kou-Yuan Huang and Wen-Lung Chang. A neural network method for prediction of 2006 world cup football game. In *IJCNN*, pages 1–8. IEEE, 2010.
5. S. P. Kvam and J. S. Sokol. A logistic regression/Markov chain model for ncaa basketball. *Naval Research Logistics*, 53:788–803, 2006.
6. Dean Oliver. *Basketball on Paper.* Brassey's, Inc., 2002.
7. Michael Pardee. An artificial neural network approach to college football prediction and ranking. Technical report.
8. M.C. Purucker. Neural network quarterbacking. *Potentials, IEEE*, 15(3):9–15, 1996.

Inverse Reinforcement Learning for Strategy Extraction

Katharina Muelling[1,3], Abdeslam Boularias[1], Betty Mohler[2], Bernhard Schölkopf[1], and Jan Peters[1,3]

[1]Max Planck Institute for Intelligent Systems, Tübingen, Germany
{muelling,boularias,bs,jrpeters}@tuebingen.mpg.de

[2]Max Planck Insitute for Biological Cybernetics, Tübingen, Germany
mohler@tuebingen.mpg.de

[3]Technische Universität Darstadt, FG IAS, Darmstadt, Germany
{muelling,peters}@ias.tu-darmstadt.de

Abstract. In competitive motor tasks such as table tennis, mastering the task is not merely a matter of perfect execution of a specific movement pattern. Here, a higher-level strategy is required in order to win the game. The data-driven identification of basic strategies in interactive tasks, such as table tennis is a largely unexplored problem. In order to automatically extract expert knowledge on effective strategic elements from table tennis data, we model the game as a Markov decision problem, where the reward function models the goal of the task as well as all strategic information. We collect data from players with different playing skills and styles using a motion capture system and infer the reward function using inverse reinforcement learning. We show that the resulting reward functions are able to distinguish the expert among players with different skill levels as well as different playing styles.

Keywords: Computational models of decision processes, Table tennis, Inverse reinforcement learning

1 Introduction

Understanding the complex interplay between learning, decision making and motion generation is crucial both for creating versatile, intelligent robot systems as well as for understanding human motor control. For example, in table tennis, a player usually cannot win the game by always returning the ball safely to the same position. Instead, players need a good strategy that defines where and how to return the ball to the opponent's court. An action should always be chosen to have a high probability of successfully returning the ball as well as to make the task of the opponent harder, i.e., it should improve the chance of winning the game. In this paper, we want to infer strategic information from a game of table tennis. Rather than identifying the frequencies and effectiveness of specific movement patterns [1–4], we want to model the decision process for choosing actions by players in a match of table tennis from a computational point of view. Thus, we are not only able to use the learned model for artificial systems, such as table tennis robots [5], but also yield a better insight into the reasons for choosing a given action in a specific state. Therefore, we only consider basic features available to the player.

A common way to model decision processes in artificial systems is to use a Markov Decision Problem (MDP [6]). Here, an agent interacts with a dynamic environment. It chooses and executes an action that will change the state of the player and its environment. The agent can observe this state change and may receive a reward for its action. A strategy defines the general plan of choosing actions in specific states in order to achieve a goal. A strategy in the MDP framework is usually called a *policy*. The expert knowledge used to win the game can be captured in the reward function that defines the reward the agent will receive in a specific situation when executing an action.

The process of determining the reward function from an expert demonstration is referred to as *Inverse Reinforcement Learning* (IRL [7, 8]). IRL has been applied to many problems such as helicopter control [9], routing preferences of drivers [10] and, user simulation in spoken dialog management systems [11]. In most of these approaches, the underlying dynamics of the system is assumed to be known. However, the dynamics of human behavior is usually difficult to model. We avoid modeling these complex dynamics by learning the strategies directly from human demonstration.

In the remainder of this paper, we will proceed as follows. In Section 2, we present the theoretical background for modeling decision processes, including MDPs and IRL techniques. We present the experimental setup and evaluations in Section 3. In Section 4, we summarize our approach and the results.

2 Modeling Human Strategies

In this section, we will first introduce the notation and basic elements necessary for the table tennis model. Subsequently, we will discuss different model-free Inverse Reinforcement Learning (IRL) approaches and show how the states, actions and reward features in the table tennis task can be represented.

2.1 Preliminaries

To employ IRL, the problem at hand needs to be modeled as a Markov Decision Problem (MDP). Formally, a MDP is a tuple (S, A, T, R), where S is the state space, A is the action space, and $\mathcal{T}$ is a transition function $\mathcal{T}(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) = Pr(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$, with states $\mathbf{s}_t, \mathbf{s}_{t+1} \in S$ and actions $\mathbf{a}_t \in A$. The function $R(\mathbf{s}, \mathbf{a})$ defines the reward for executing action $\mathbf{a}$ in state $\mathbf{s}$.

A deterministic policy π is a mapping: $S \mapsto A$ and defines which action is chosen in a state $\mathbf{s} \in S$. A stochastic policy is a probability distribution over actions in a given state $\mathbf{s}$ and is defined as $\pi(\mathbf{s}|\mathbf{a}) = Pr(\mathbf{a}|\mathbf{s})$. The performance of a policy is measured with the so-called *value function* $V^{\pi}(\mathbf{s})$. The value function of a policy π evaluated at state $\mathbf{s}$ for a finite horizon H is given by $V^{\pi}(\mathbf{s}) = \frac{1}{H}\mathbb{E}[\sum_{t=0}^{H-1} R(\mathbf{s}_t, \mathbf{a}_t)|\pi, \mathcal{T}, \mathbf{s}_0 = \mathbf{s}]$, and corresponds to the expected reward following policy πstarting from state $\mathbf{s}$. The optimal value function is defined by $V^*(\mathbf{s}) = \max_\pi V^{\pi}(\mathbf{s})\ \forall \mathbf{s} \in S$. The goal of an agent is to find the optimal policy π^*, i.e., a policy that maximizes the expected return for every $\mathbf{s} \in S$.

We assume that the reward function R is given by a linear combination of m feature functions f_i with weights w_i. The reward function is therefore defined by $R(\mathbf{s},\mathbf{a}) = \sum_{i=1}^{m} w_i f_i(\mathbf{s},\mathbf{a}) = \mathbf{w}^{\mathrm{T}}\mathbf{f}(\mathbf{s},\mathbf{a})$, where $\mathbf{w} \in \mathbb{R}^m$ and $\mathbf{f}(\mathbf{s},\mathbf{a}) \in \mathbb{R}^m$. The features f_i are fixed, known, bounded basis functions mapping from $S \times A$ into $\mathbb{R}$. Similarly to the value function, we can define the feature count f_i^{π} under policy π by $f_i^{\pi}(\mathbf{s}) = \frac{1}{H}\mathbb{E}[\sum_{t=0}^{H-1} f_i(s_t,a_t)|\pi,\mathcal{T},\mathbf{s}_0 = \mathbf{s}]$ as the expected features observed when following policy π. As a result, V^{π} can be written as $V_{\mathbf{w}}^{\pi}(\mathbf{s}) = \sum_{i=1}^{m} w_i f_i^{\pi}(\mathbf{s}) = \mathbf{w}^{\mathrm{T}}\mathbf{f}^{\pi}(\mathbf{s})$, where $\mathbf{f}^{\pi} \in \mathbb{R}^m$ is a vector containing the single feature counts $f_i^{\pi}(\mathbf{s})$ as entries.

2.2 Learning the Reward Function

The reward function is a crucial part of the MDP as it defines the goal of the task and shapes the policy optimization process. The problem of designing the right reward function led to the development of IRL methods. Given the actions of an agent that is assumed to behave in an optimal manner, the available sensory information about the environment and, if possible, a model of the environment, the goal of IRL is to determine a reward function that can (mostly) justify the demonstrated behavior. A recent review of IRL algorithms can be found in [12].

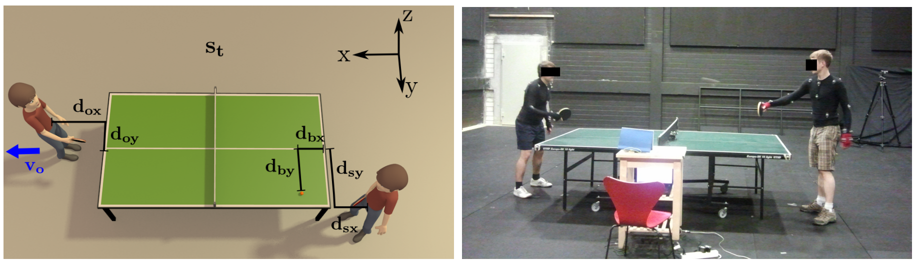
Most IRL approaches rely on a given model of the environment $\mathcal{T}$ or assume that it can be accurately learned from the demonstrations. In this paper, we want to estimate the underlying reward function for playing table tennis based on demonstrations *without having to model the correct dynamics model.* Only few model-free IRL methods have been suggested [13, 14].

Instead of collecting only demonstrations from an expert we use also demonstrations from less skilled players for finding the reward function. To compute the reward weights, we compared two different methods. The first evaluated method is based on the max-margin algorithm of Abbeel and Ng [15], while the second is the model-free relative entropy IRL algorithm [13]. In the following, we assume that we are given a set of expert demonstrations $D^E = \{\tau_p\}_{p=1}^{P}$, where $\tau_p = \mathbf{s}_1^p\mathbf{a}_1^p, ..., \mathbf{s}_{T_p}^p\mathbf{a}_{T_p}^p$ corresponds to one rally (i.e., state-action trajectory), as well as a set of non-optimal demonstrations $D^N = \{\tau_l\}_{l=1}^{L}$. Here, T_p is the number of volleys (i.e., state-action pairs) in the observed rally τ_p.

Model Free Maximum Margin. The max-margin method of Abbeel and Ng [15] aims at finding a policy π that has feature counts close to that of the expert, i.e., $\|\mathbf{f}^{\pi} - \mathbf{f}^{\pi_E}\|_2 \leq \epsilon$. Using the max-margin algorithm [15] in a model-free setup in a straight forward manner is not suited as it is unlikely that any player plans the strokes for more than only a few steps ahead. Therefore, we need to compare the values of the expert in every state in the recorded trajectories to the ones of the non-experts in the same state. We can find the weight vector $\mathbf{w}$ by solving the quadratic optimization problem

$$\max_{\mathbf{w}} \sum_{p=1}^{P}\sum_{t=0}^{T_p}\left(V_{\mathbf{w}}^{\pi_E}(\mathbf{s}_t^p) - \hat{V}_{\mathbf{w}}^{\pi_N}(\mathbf{s}_t^p)\right) - \lambda||\mathbf{w}||_2,$$

where $\hat{V}_{\mathbf{w}}^{\pi_N}(\mathbf{s}_t^p)$ is an estimated value of the non-expert players in the current state $\mathbf{s}_t^p$ of the expert. Estimating the value $\hat{V}^{\pi_N}$ in a given state $\mathbf{s}$ is a regression problem that we propose to solve by using the k-nearest neighbors method,



(a) State (b) Experimental Setup

Fig. 1. Fig. a illustrates the state of the system, defined by the relative position of the agent (d_{sx}, d_{sy}) and the the relative position (d_{ox}, d_{oy}) and velocity ($\mathbf{v}_o$) of the opponent towards the table, as well as the the position (d_{bx}, d_{by}) and velocity ($\mathbf{v}_b$) of the ball. Fig. b shows the experimental setup. A naive player (right side) plays against an skilled opponent (left side).

$\hat{V}^{\pi_N}_{\mathbf{w}}(\mathbf{s}) = \frac{1}{k}\sum_{\mathbf{s}'\in\mathcal{N}_k(\mathbf{s})} V^{\pi_N}_{\mathbf{w}}(\mathbf{s}')$, where $\mathcal{N}_k(\mathbf{s})$ is the set of k-nearest neighbors of $\mathbf{s}$ among all the states that have been observed in the non-optimal trajectories. We use a Gaussian kernel to define a similarity measure between states.

The value functions V^{π_E} and V^{π_N} of the expert's policy π_E and non-optimal policies π_N are computed as

$$V^{\pi}_{\mathbf{w}}(\mathbf{s}^p_t) = \frac{1}{H^p_t - t + 1}\sum_{i=t}^{H^p_t} \mathbf{w}^{\mathrm{T}}\mathbf{f}^{\pi}(\mathbf{s}^p_i, \mathbf{a}^p_i),$$

where $H^p_t = \min\{t + H - 1, T_p\}$ and H is the planning horizon, i.e., the number of steps we look into the future. In the following, we will refer to this algorithm as MM (Maximum Margin).

Relative Entropy Method. The relative entropy IRL method [13] finds a distribution $\mathcal{P}$ over trajectories that minimizes the KL-divergence to a reference distribution Q, while ensuring that the feature counts under $\mathcal{P}$ are similar to the feature counts in the expert trajectories. The solution to this problem takes the following form

$$\mathcal{P}(\tau|\mathbf{w}) = \frac{1}{Z(\mathbf{w})}Q(\tau)\exp\left(\mathbf{w}^T f^{\tau}_i\right),$$

where $Z(\mathbf{w}) = \sum_\tau Q(\tau)\exp\left(\mathbf{w}^T f^{\tau}_i\right)$. The reward weight vector $\mathbf{w}$ is found by solving the optimization problem $\max_{\mathbf{w}} \mathbf{w}^T\mathbf{f}^{\pi_E} - \ln Z(\mathbf{w}) - \lambda\|\mathbf{w}\|_1$. The gradient of this objective function is calculated by re-using the expert and non-optimal trajectories with importance sampling. For our experiments, we choose the reference distribution Q to be uniform. In the following, we will refer to this algorithm as RE (Relative Entropy).

2.3 Computational Model for Representing Strategies in Table Tennis

As a next step, we need to specify the states, actions and reward features of the table tennis task. The state of the system consist of all sensory information experienced by the agent. However, learning in such high-dimensional continuous state domains is likely to be intractable. Therefore, we assume that the player has to decide where and how to hit the ball when the hitting movement is initiated. Furthermore, we assume that the decision depends on the following information: the planar Cartesian position of the agent $\mathbf{d}_s = [d_{sx}, d_{sy}]$, the opponent's position $\mathbf{d}_o = [d_{ox}, d_{oy}]$ and velocity $\mathbf{v}_o$, the state of the rally $\mathbf{g} \in \{\text{player serve, opponent serve, not served}\}$ as well as the ball position $\mathbf{d}_b = [d_{bx}, d_{by}]$, velocity $|\mathbf{v}_b|$ and direction given by the angles θ_{py} and θ_{pz} (see Fig. 1). The variables θ_{py} and θ_{pz} are defined as the horizontal and vertical bouncing angles of the ball at the moment of impact on the player's side of the table, respectively. θ_{pz} defines the bouncing angle in the xz-plane and corresponds to how flat the ball was played. θ_{py} defines the bouncing angle in the xy-plane. Additionally, we define a set of terminal states $s_T \in \{W, L\}$ for winning and loosing the rally respectively.

The action defines where and how to return the ball to the opponent's court. This decision includes the desired bouncing point $\mathbf{p}_b$ of the ball on the opponent's court, the corresponding bouncing angles θ_{oy} and θ_{oz}, the velocity $||\mathbf{v}_b||$ and the spin of the ball. Since the different kinds of spin are hard to capture without an expert classifying the sampled data, we discard the spin and use only basic strategic elements.

The reward features $f_i(\mathbf{s}, \mathbf{a})$ for each state-action pair are defined by: (i) the goal of the ball on the opponent's court, (ii) the proximity of the ball to the edge of the table $\boldsymbol{\delta}_{\text{t}}$, (iii) the distance of the bouncing point of the ball on the opponent's court and the right hand of the opponent $\boldsymbol{\delta}_{\text{o}}$, (iv) the proximity of the ball to the elbow δ_{elbow}, (v) the velocity of the ball $\|\mathbf{v}_b\|$, (vi) the velocity of the opponent v_o relative to the ball in y-direction, (vii) the bouncing angles θ_{oz} and θ_{oy} of the ball when bouncing on the opponent's side of the court, and (viii) whether the ball was a smash or not. All features are scaled to lie in an interval of [0 1], except for the direction sensitive features θ_{oy} and v_o, which lie in an interval of [-1 1].

3 Experiments and Evaluations

In this section we describe the experiments for the data collection and the results of the evaluation of the presented approach.

3.1 Experimental Setup and Data Collection

We recorded table tennis players with various skill levels. Therefore, we used eight right-handed subjects of all genders which could be grouped into naive and skilled players. The group of naive players consisted of five subjects and fulfilled the following requirements: (i) never played in a table tennis club, (ii) did not train any racket sports on a regular basis in the last five years, and (iii) did not participate in table tennis tournaments. The group of skilled players consisted of three subjects and fulfilled the following requirements: (i) played for at least eight years in a table tennis club, (ii) trained at least twice a week and (iii) participate regularly in table tennis competitions.

One of the skilled players were used as a permanent *opponent* and, therefore, was not considered part of the subject set. Each subject played a game of table tennis under the following three conditions. In Condition 1, the subjects played a cooperative game of table tennis for a ten minute period. In Condition 2, the subjects were told to perform a competitive game of table tennis, while the opponent was instructed to return the ball "nicely" (i.e., the opponent was instructed to play towards the subject in a cooperative way). In Condition 3, both the subject and the opponent were instructed to play a competitive game of table tennis.

In order to collect information about the position of the participants, the table and the ball during the game, we used a VICON motion capture system. With this setup a 3D kinematic model of the upper body of each individual was captured during the game. The experimental setup is also shown in Fig. 1b.

3.2 Results and Discussion

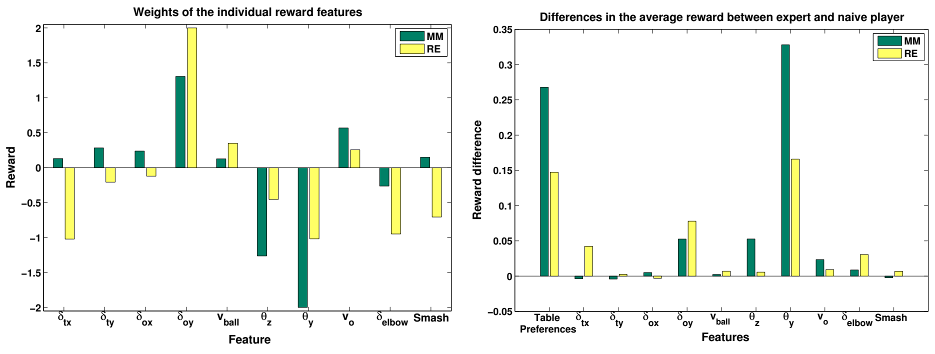
Only one of the skilled subjects was able to win against the opponent under Condition 3. All other games were won by the fixed opponent. The scoring results of the subjects that lost the game can be found in Table 1. Based on these results, the data was divided into two subsets: (1) a non-expert data set and (2) an expert data set. The non-expert data set included all games of the subjects who lost against the fixed opponent, i.e., all naive subjects and one of the skilled players, as well as all cooperative games. We will refer to the players that lost as Naive 1 to 5 and Skilled 1. The expert data set consisted of all rallies in the competitive game (Condition 3) of the skilled player that won against the opponent. We will refer to this player as Expert. When asked which player performed worst, the opponent stated that Naive 3 was the worst.

To evaluate the potential reward functions, we performed a leave-one-subject-out testing scheme. We computed the reward feature weights for each of the two methods as described in Section 2.2 seven times. Every time leaving out all rallies (i.e., state-action trajectories) of one of the subjects that lost or the rallies of the cooperative game of the Expert respectively. We also excluded 20 rallies of the Expert for the validations. For the MM algorithm we determined empirically an optimal planning horizon of three, which is used throughout the evaluations.

Classifying the skill levels of the players. We computed the differences in the average reward for a state-action pair of the spared expert and non-expert

Table 1. Summary of the results of the evaluations for the different methods. The differences in the average rewards with respect to the expert, define the differences between the reward of the expert and the spared test subject of the non-expert data set.

	Method	Naive 1	Naive 2	Naive 3	Naive 4	Naive 5	Skilled 1	Coop.
Average reward	MM	1.16	0.07	1.24	0.86	0.71	0.33	0.50
differences	RE	0.70	0.11	0.60	0.80	0.42	0.31	0.55
Scores in Condition 2		5:33	12:33	2:33	5:33	2:33	21:34	
Scores in Condition 3		13:33	17:33	10:33	5:33	17:33	20:33	



(a) Reward feature weights (b) Average reward differences

Fig. 2. Fig. (a) shows the weights of all other features for the MM algorithm and the RE algorithm, respectively. Fig. (b) shows the differences of the average reward of the expert and the naive player for each feature seperately.

data for the obtaind reward functions (see Table 1). All reward functions were able to distinguish between the non-expert games and the expert game, as well as between the different playing styles of the expert (competitive vs cooperative). In general the average reward for each player reflected the skill level of the players with the exception of Naive 2.

All reward functions obtained in the evaluation resulted in a very small difference in the average reward of the Expert and Naive 2, followed by Skilled 1 and Naive 5. Furthermore, both methods showed relatively large differences between the Expert and Naive 1, Naive 3 and Naive 4. However, they disagree in the ranking of these players. While the reward function obtained by the RE algorithm shows the highest difference for the Expert and Naive 4, the reward function obtained by the MM algorithm yields the highest difference between the Expert and Naive 3. Naive 4 being the worst player is in compliance with the scoring results for Condition 3, while Naive 3 being the worst player is in compliance with the statement of the opponent. Analyzing player Naive 2, we can conclude that the player chooses his actions based on the same principles as both skilled players, but lost against the opponent due to his inaccurate movement execution.

Individual reward features. Analyzing the reward weights individually, the different methods showed similar weights for the most important features (i.e., the features with the highest weights). The reward weights and differences for the individual features are displayed in Fig. 2a and b. The largest influence resulted from the bouncing angles θ_y and θ_z, the table preferences and the distance between the desired bouncing point and the racket of the opponent. We will discuss these features in the following. Other features as playing against the moving direction and the velocity of the ball were also positive correlated.

Goal preferences on the table. The resulting reward functions of the different algorithms showed a preference for the areas where the opponent would have to return the ball using the backhand, while the areas that are suited for returning the ball with the forehand and the areas directly after the net are rather avoided.

Distance to the opponent. Maximizing the distance in y-direction (i.e., along the width of the table) between the bouncing point and the racket of the opponent resulted in a high reward in both reward functions. This feature also influenced the differences in the reward yield by the naive and expert table tennis player. The overall performance on average only increased slightly. The differences in the average reward for the features before a terminal state, increased dramatically and became a dominant factor in the reward function (see Fig. 2b). This observation suggests that the chance of winning a point increases with an increasing distance between the bouncing point and the racket between the player.

Bouncing Angles. The horizontal angle θ_z had a high negative reward value, i.e., playing the ball flat was preferred. The angle θ_y also had a high negative weight, i.e., playing the ball cross to the backhand area was preferred opposed to playing the ball cross towards the forehand area. These results are conform with the table preferences. This feature was one of the dominating factors in the reward function and in the evaluations of the excluded subjects. However, the average difference between expert and naive players for the state right before the terminal state was only decreased slightly. The average reward two states before the terminal state on the other side became one of the dominant factors.

This observation together with the results of the distance of the bouncing point and the racket, suggests the following strategy successfully applied by the Expert only. When playing the ball very cross to the outer backhand area of the opponent, the opponent was forced to move to his left. The expert used this opportunity to play the ball to the other side of the table in order to increase the distance between the ball and the opponent.

4 Conclusion

In this paper, we modeled table tennis as a MDP. We have shown that it is possible to automatically extract expert knowledge on effective elements of basic strategy in the form of a reward function using model-free IRL. To accomplish this step, we collected data from humans playing table tennis using a motion capture system. Participants with different skill levels played in both a competitive and a cooperative game during this study. We divided the data into an expert and a non-optimal data set and used them to infer and evaluate the reward functions.

We have tested two different model-free inverse reinforcement learning methods. One was derived from the model-based IRL method of Abeel and Ng [15]. The second algorithm was model-free relative entropy [13]. The resulting reward functions were evaluated successfully in a leave-one-subject-out testing scheme. All learned reward functions were able to distinguish strategic information of players with different playing skills and styles. The key elements revealed by the model were (i) playing cross to the backhand area of the opponent, (ii) maximizing the distance of the bouncing point of the ball and the opponent, and (iii) playing the ball in a flat manner. Other elements as playing against the moving direction and the velocity of the ball were also positively correlated.

References

1. P. Wang, R. Cai, and S. Yang, “A tennis video indexing approach through pattern discovery in interactive process,” *Advances in Multimedia Information Processing*, vol. 3331, pp. 59 – 56, 2004.
2. J. Wang and N. Parameswaran, “Analyzing tennis tactics from broadcsting tennis video clips,” in *Proceedings of the 11th International Multimedia Modelling Conference*, pp. 102 – 106, 2005.
3. J. Vis, W. Kosters, and A. Terroba, “Tennis patterns: Player, match and beyond,” in *22nd Benelux Conference on Artificial Intelligence*, 2010.
4. A. Hohmann, H. Zhang, and A. Koth, “Performance diagnosis through mathematical simultion in table tennis,” in *Science and Racket Sports III* (A. Lees, J.-F. Kahn, and I. Maynard, eds.), pp. 220 – 226, London: Routledge, 2004.
5. K. Muelling, J. Kober, O. Kroemer, and J. Peters, “Learning to select and generalize striking movements in robot table tennis,” *The International Journal of Robotics Research*, vol. 32, no. 3, pp. 263 – 279, 2013.
6. M. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* New York, NY, USA: John Wiley & Sons, Inc., 1st ed., 1994.
7. S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, vol. 15 of *Studies in Applied Mathematics.* Philadelphia, PA: SIAM, June 1994.
8. A. Ng and X. Russel, “Algorithms for inverse reinforcement learning,” in *Proceedings of the 17th International Conference of Machine Learning*, pp. 663 – 670, 2000.
9. P. Abbeel, A. Coates, and A. Ng, “Autonomous helicopter aerobatics through apprenticeship learning,” *The International Journal of Robotics Research*, vol. 29, pp. 1608 – 1679, 2010.
10. B. Ziebart, A. Maas, A. Bagnell, and A. Dey, “Maximum entropy inverse reinforcement learning,” in *Proceedings of the 23th National Conference of Articiial Intelligence (AAAI)*, pp. 1433 – 1438, 2008.
11. S. Chandramohan, M. Geist, F. Lefevre, and O. Pietquin, “User simulation in dialogue systems using inverse reinforcement learning,” in *Proceedings of the 12th Annual Conference of the International Speech Communication Association*, 2011.
12. S. Zhifei and E. Joo, “A survey of inverse reinforcement learning techniques,” *International Journal of Intelligent Computing and Cybernetics*, vol. 5, no. 3, pp. 293 – 311, 2012.
13. A. Boularias, J. Kober, and J. Peters, “Relative entropy inverse reinforcement learning,” in *Proceedings of the Artificial Intelligences and Statistics (AISTATS)*, pp. 20 – 27, 2011.
14. T. Mori, M. Howard, and S. Vijayakumar, “Model-free apprenticeship learning for transfer of human impedance behaviour,” in *Proceedings of the 11th IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS)*, pp. 239 – 246, 2011.
15. P. Abbeel and A. Ng, “Apprenticeship learning via inverse reinforcement learning,” in *Proceedings of the 21st International Conference of Machine Learning (ICML)*, 2004.

Key point selection and clustering of swimmer coordination through Sparse Fisher-EM

John Komar[1], Romain Herault[2], and Ludovic Seifert[1]

[1] CETAPS EA-3832 Université de Rouen,
Boulevard Siegfried, 76821 Mont Saint Aignan, France
`firstname.lastname@univ-rouen.fr`
[2] LITIS EA-4108, INSA de Rouen,
Avenue de l'Université - BP 8,
76801 Saint-Étienne-du-Rouvray Cedex, France
`firstname.lastname@insa-rouen.fr` [⋆]

Abstract. To answer the existence of optimal swimmer learning/teaching strategies, this work introduces a two-level clustering in order to analyze temporal dynamics of motor learning in breaststroke swimming. Each level have been performed through Sparse Fisher-EM, a unsupervised framework which can be applied efficiently on large and correlated datasets. The induced sparsity selects key points of the coordination phase without any prior knowledge.

Keywords: Clustering, Variable selection, Temporal dynamics of motor learning, Sparse Fisher-EM

1 Introduction

The development of Dynamical Systems Theory [1] in understanding motor learning has increased the interest of sports scientists in focusing on temporal dynamics of human motor behavior. Broadly speaking, the investigation of motor learning traditionally implied the assessment of both a pre-learning behavior and a post-learning behavior [2], but the deep understanding of the process of motor learning requires a continuous and long term assessment of the behavior rather than previous traditional discrete assessments. Indeed, such a continuous assessment of behavioral data enables to investigate the nature of the learning process and might highlight the paramount role played by motor variability in optimizing learning [2].

From a theoretical point of view, motor learning is viewed as a process involving active exploration of a so-called perceptual-motor workspace which is learner dependent and defines all the motor possibilities available to him. Few studies have already highlighted this exploratory behavior during learning a ski

[⋆] Authors would like to thank the *Agence Nationale de la Recherche* (ANR) for its financial support to§ the project LeMOn (Learning with Multi-objective Optimization, ANR-11-JS02-10).

simulator task [3] or a soccer kicking task [4]. These authors showed that learners exhibited different qualitative motor organizations during skill acquisition. Nevertheless, these princeps studies mainly focused on a static analysis, defining the different behaviors exhibited during learning. As a matter of fact, a major interest in the field of motor learning resides in the definition of different pathways of learning, namely different possible learning strategies [5]. Such an interest in investigating the existence of different "routes of learning" needs to focus on a dynamical analysis, namely the analysis of the successions of different behaviors. An unanswered question to date concerns the existence of optimal learning strategies (i.e. strategies that would appear more effective). Thus, the discovery of optimal learning strategies could have a huge impact on the pedagogical approach of practitioners.

The article will describe at first the context of the research insisting on the way data have been collected, what are the long-term expectations in sport science field and what are the short term locks in machine learning field. Then we will give a brief view of the Fisher-EM algorithm [6] which is an unsupervised learning method used in this work. In the end, preliminary results of the data clustering will be analyzed.

2 Context of the Research

2.1 Previous work

In breaststroke swimming, achieving high performance requires a particular management of both arm and leg movements, in order to maximize propulsive effectiveness and optimize the glide and recovery times [7]. Therefore, expertise in breaststroke is defined by adopting a precise coordination pattern between arms and legs (i.e. a specific spatial and temporal relationship between elbow and knee oscillations). Indeed, when knees are flexing, elbows should be fully extended (180°), whereas knees should be fully extended (180°) when elbows are flexing, in order to ensure a hydrodynamic position of the non-propulsive limbs when the first pair of limbs is actually propulsive [8, 9].

Based on this context, the breaststroke swimming task was deemed as suitable in investigating the dynamics of learning, mainly as it implies at a macroscopic scale the acquisition of an expert arm-leg coordination that can be easily assessed. however, the investigation of potential differences in learning strategies required a continuous movement assessment. In that sense, the use of motion sensors allowed a fast, accurate and cycle per cycle movement assessment.

Previously, two analysis methods were used in the cycle per cycle study of motor learning. A previous study [3] highlighted the unstable character of the transition between novice and expert, but not really an exploration as experimental setup assumes that novices left their initial behavior to adopt the expert one. Therefore, no search strategies were really investigated. In order to overcome this issue, [4] used a cluster analysis (Hierarchical Cluster Analysis) in their experiment on football kicking and highlighted different behaviors used by each participant during learning to kick a ball. The authors therefore linked these different behaviors to a search strategy. However, the cluster analysis was performed individually and there was no comparison done between the learners (e.g. did they use identical behaviors?), it implied only few participants (i.e. four learners), it was performed only with 120 kicks per learner (i.e. 10 kicks per session during 12 sessions) and like the previous study of [3] it only defined the behavior from a static point of view (i.e. defining what behavior was adopted).

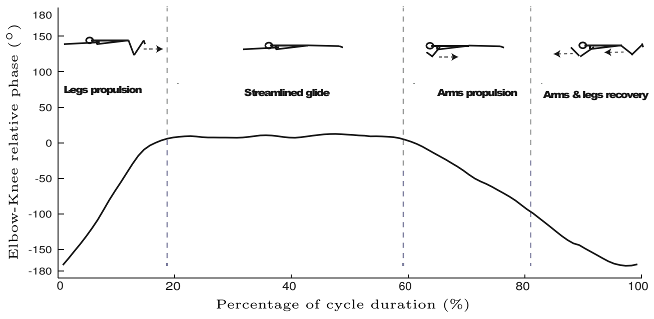


Fig. 1: A typical continuous relative phase between the knee and the elbow

2.2 Data collection

For this study, 26 novices were involved in 16 lessons of breaststroke swimming, with two sessions per week for a total duration of two months. The general goal of learning for all the 26 swimmers was to increase the distance per stroke, while maintaining the speed stable. Then the 26 learners were divided into four different groups, each group receiving a different instruction during the learning process: 1) Control group (N=7): This group received only the general goal of learning, increase the distance per stroke 2) Analogy group (N=7): In addition to the general goal of learning, this group received a single additional instruction: "glide two seconds with your arms outstretched" 3) Pacer group (N=6): In addition to the general goal of learning, this group had to follow an auditory metronome trying to perform one cycle every single auditory signal. The frequency of the metronome was decreased every two sessions, in order to promote a decrease in the stroke frequency of the learners that should lead to an increase in the distance per stroke 4) Prescription group (N=6): In addition to the general goal of learning, this group received multiple additional instructions: "keep your arms outstretched forward when you extend your legs; then glide with your arms and legs outstretched; then keep your legs outstretched when you flex your arms; recover both arms and legs together". These different instructions were supposed to have a specific impact on the learning strategies of the learners.

Each learner performed 10 trials of 25-m swim during each session, with 1 x 25-m consisting approximatively in 8 recorded cycles (one cycle correspond to the period between two successive maximal knee flexion). During every learning session, all learners were equipped with small motion sensors on both arms and legs (3-D gyroscopes, 3-D magnetometers, 3-D accelerometers) including a data logger and recording elbow and knee angles at a frequency of 200 Hz. Following the literature in coordination dynamics [1], the coordination between elbow and knee was defined by the continuous relative phase between these two oscillators [10], considering elbows and knees as acting like individual pendulums [7]. A value of relative phase close to -180° or 180° defined an anti-phase relationship (i.e. opposite movements of knee and elbow) while a value close to 0° defined an in-phase mode of coordination (i.e. identical movements of knee and elbow); here, each cycle will be described by a time series of 100 normalized values of continuous relative phase between the knee and the elbow (Fig. 1).

To sum-up, we have recorded 4160 trials (26 swimmers $\times$ 16 sessions $\times$ 10 trials) and there is an average of 8 cycles per trials. Thus, the dataset is composed by 33280 cycles, each cycle is represented by 100 continuous relative phase samples.

2.3 Study expectations

From a sport sciences point of view, the specific aims of the study were twofold: – Assessing the dynamics of learning: In other words, the aim was to assess not only the different behaviors used during learning but also the transitions between these behaviors, that is the potential search strategy exhibited by learners (e.g. they used preferably behavior n^o 1 then n^o 4, then n^o 3 . . .). – Assessing the impact of different learning conditions on the dynamics of learning: In other words, the aim was to investigate the possible existence of different behaviors exhibited by the learners regarding their learning condition, as well as the possible existence of different search strategy exhibited by the different groups.

A last point in this experiment was the possibility to transfer the results of the analysis towards practical application or guidelines for teachers. From a pedagogical point of view, it appeared difficult to teach novice swimmers by giving instruction on the arm-leg coordination during all the cycle and the definition of key points within the entire cycle reflects a paramount aspect for teaching. Indeed, a strong literature in sports pedagogy highlights the role played by attentional focalization during motor learning, as a focalization on a key point of the swimming cycle may be highly beneficial in seeking to reorganize the entire arm-leg coordination [11]. A third aim of this study was then to define highly discriminative key points within the swimming cycle and that might be the target of the instruction in order to orient the attention of learners.

From a machine learning point of view, there are two locks to tackle: 1) Each cycle is described by 100 features which are highly correlated due to the fact that they are samples of the relative phase which is a continuous time signal. Nevertheless, we don't want to bias the study by preprocessing the data, a transformation like filters, wavelet transform or sample selection that will embedded our a priori knowledge. 2) The number of cycles are not equal on all the trials, that is why a trial can not be directly described by a fixed number of features.

Those two problems were address by 1) using a clustering by Fisher-EM [6] that also performs dimension reduction and features selection, 2) doing a two stage clustering: on cycles then on trials; a procedure similar to *Bags of words* to have fixed size features on trial.

3 Fisher-EM Algorithm

A clustering can be derived from a mixture of Gaussians generative model. A Gaussian, which is parameterized by a covariance matrix and a mean in the observation space, represents a cluster. An observation is labeled according to its ownership (likelihood ratio) to each Gaussian. Knowing the number of clusters, the mixture and Gaussian parameters are learned from the observation data trough an Expectation-Maximization (EM) algorithm.

The Fisher-EM algorithm [6] is based on the same principles but the mixture of Gaussians does not lie directly on the observation space but on a lower dimension latent space. This latent space is chosen to maximize the Fisher criterion between clusters and thus be discriminative and its dimension is bounded by the number of clusters. This reduction of dimension leads to more efficient computation on medium to large datasets (here 33280 examples by 100 features) as operations can be held in the smaller latent space.

3.1 Generative Model

We consider that the n observations $y_1, y_2, \ldots, y_n$ are realizations of a random vector $Y \in \mathbb{R}^p$. We want to cluster these observations into K groups. For each observation y_i, a variable $z_i \in Z = \{1, \ldots, K\}$ indicates which cluster its belong to. This clustering will be decided upon a generative model, namely a mixture of K Gaussians which lies in a discriminative latent space $X \in \mathbb{R}^d$ where $d \leq K-1$.

This latent space is linked to the observation space through a linear transformation,

$$Y = UX + \epsilon \ , \tag{1}$$

where $U \in \mathbb{R}^{p \times d}$ and $U^t U = Id(d)$ where $Id(d)$ is the identity matrix of size d, i.e. U is an orthogonal matrix and ϵ non-discriminative noise.

Let be $W = [U, V] \in \mathbb{R}^{p \times p}$ such that $W^t W = Id(p)$. V is the orthogonal complement of U. Thus, a projection $U^t y$ of an observation y from space Y of dimension p, lies on the latent discriminative subspace X of dimension d and the projection $V^t y_i$ lies on the non-discriminative complement subspace of dimension $p - d$.

Conditionally to $Z = k$, random variables X and Y are assumed to be Gaussian, $X_{|Z=k} \sim \mathcal{N}(\mu_k, \Sigma_k)$, and $Y_{|Z=k} \sim \mathcal{N}(m_k, S_k)$, where $\mu_k \in \mathbb{R}^d$, $\Sigma_k \in \mathbb{R}^{d \times d}$, $m_k \in \mathbb{R}^p$ and $S_k \in \mathbb{R}^{p \times p}$.

With the help of equation 1, we can deduce parameters of the distribution $Y_{|Z=k}$ in the observation space from the parameters of the distribution $X_{|Z=k}$

in the latent space, $m_k = U\mu_k$ and $S_k = U\Sigma_k U^t + \Psi$, where $\Psi \in \mathbb{R}^{p\times p}$ is the covariance matrix of ϵ which is assumed to follow a 0-centered Gaussian distribution. To ensure that ϵ represents non-discriminative noise, we will impose that the covariance of ϵ, Ψ, projected into the discriminative space is null, i.e. $U\Psi U^t = \mathbf{0}(d)$, and that Ψ projected into the non-discriminative subspace is diagonal, i.e. $V\Psi V^t = \beta Id(p-d)$. Thus,

$$W^t S_k W = \begin{pmatrix} \Sigma_k & \mathbf{0} \\ \mathbf{0} & \beta_k Id(p-d) \end{pmatrix} . \tag{2}$$

All the Gaussian distributions are mixed together, the density of the generative model is given by $f(y) = \sum_{k=1}^{K} \pi_k \phi(y; m_k, S_k)$ where π_k are mixing proportion and m_k, S_k are deduced from $\{U, \beta, \mu_k, \Sigma_k\}$.

Finally, the model is parameterized by: – U the projection from discriminative subspace to observation space, – β_k variance of ϵ in the non-discriminative subspace, – π_k the mixing parameter, – and Gaussian parameter $\{\mu_k, \Sigma_k\}$, where the 3 last parameters are repeated by the number of Gaussians.

Model variations, that lead to reduced numbers of parameters, can be achieved by enforcing shared covariances β and/or Σ between Gaussians, diagonalization of the covariance Σ without or with constant diagonal, and combination of these enforcements.

3.2 Parameter estimation

The iterative Expectation-Maximization (EM) algorithm can be extended by a Fisher Step (*F-Step*) in-between the *E-Step* and the *M-Step* where the latent discriminative subspace is computed [6]. The Fisher criterion computed at the *F-Step* is used as a stopping criterion. Convergences properties can be found in [12].

E-Step In this step, for each observation i, its posterior probability to each cluster k is computed by

$$o_{ik} \leftarrow \frac{\pi_k \phi(y_i, \hat{\theta}_k)}{\sum_{l=1}^{K} \pi_l \phi(y_i, \hat{\theta}_l)} ,$$

where $\hat{\theta}_k = \{U, \beta, \mu_k, \Sigma_k\}$. From these probabilities, each observation can be given to a cluster by $z_i = \underset{k}{\arg\max}\ o_{ik}$.

F-Step The projection matrix U is computed such that Fisher's criterion is maximized in the latent space,

$$U \leftarrow \begin{array}{ll} \underset{U}{\arg\max}\, trace & \left(\left(U^t S U\right)^{-1} U^t S_B U \right) \\ w.r.t. & U^t U = Id(d) \end{array} ,$$

where S is the variance of the whole dataset and $S_B = \frac{1}{n}\sum_{k=1}^{K} n_k (m_k - \bar{y})(m_k - \bar{y})^t$ where $n_k = \sum_i o_{ik}$ and $\bar{y}$ the mean of the dataset.

M-Step Knowing the posterior probabilities o_{ik} and the projection matrix U, we compute the new Gaussian parameters by maximizing the likelihood of the observations,

$$\hat{\pi}_k \leftarrow \frac{n_k}{n},\ \hat{\mu}_k \leftarrow \frac{1}{n_k}\sum_{i=1}^{n} o_{ik}U^t y_i,\ \hat{\Sigma}_k \leftarrow U^t C_k U,\ \hat{\beta}_k \leftarrow \frac{trace(C_k) - \sum_{j=1}^{d} u_j^t C_k u_j}{p-d},$$

where u_j is the j-th column of U and $C_k = \frac{1}{n_k}\sum_{i=1}^{n} o_{ik}(y_i - m_k)(y_i - m_k)^t$ the empirical covariance matrix of the cluster k.

3.3 Sparse version

Yet, the use of latent space introduces dimension reduction and computation efficiency. Nevertheless the back-projection from the latent space to the observation space can involve all the original features. To do feature selection, the projection matrix U has to be sparse. [13] proposed 3 methods to enforce sparsity: 1) After a standard F-step, compute an sparse approximation of U independently of the Fisher criterion, 2) Compute the projection with a modified Fisher criterion with a L_1 penalty on U, 3) Compute U from the Fisher criterion using a penalized SVD algorithm.

4 Application to swimmer coordination

The clustering is done in two steps: 1) A clustering on cycle data. Here an observation is just one swimming cycle. This clustering has two purposes, a) give a label to each cycle b) select which phase samples over the 100 are informative through sparsity. 2) A clustering on trials. Each trial can be described now by a sequence of cycle labels learned at the first step. Features for this clustering consist in the transition matrix of the sequence with its diagonal put to zero. The number of cluster is chosen by analysis of the Bayesian information criterion (BIC).

For the first clustering level, analysis of the BIC (Tab. 1) highlights the existence of 11 clusters within the whole set of data. The mean coordination of these clusters are represented at Figure 2a. This result advocates for qualitative reorganizations of motor behavior during motor learning, as each learner visited between 9 and 11 different clusters during their sessions. For instance, the mean and standard deviation of one cluster (n^{o}8) is presented in Figure 2b.

In order to differentiate the effect of the different instructions on the learning process, Table 2 shows the distribution of each emerging cluster across the different learning conditions. Interestingly, the use of different additional instructions led to the exhibition of different preferred patterns of coordination. For instance, the group who received an analogy exhibited preferably clusters 3, 7, 8 and 9, whereas clusters 2, 4 and 10 were inhibited. In the meantime, the use of the prescriptive instruction preferably led to the use of cluster 5 and inhibited the use of clusters 2, 6 and 10. This result is a key point of the experiment, validating the possibility of guiding the exploration during learning and by extension the result of the learning process with using different types of instructions during the practice.

On Figure 2c, we have superimposed a typical coordination curve and, in gray bars, the back-projection of latent space into observation space to see induced sparsity from the first level. The height of a bar at a feature $i \in [1 \dots p]$ is proportional to $\sum_{j=1}^{d} |U_{ij}|$. A null value shows that the corresponding feature is not involved in the projection to the latent space, i.e. it is not selected by the F-Step or it is squeezed by the sparsity; therefore it can be considered not relevant to build the clusters. Interestingly, only key points of the movement have high values, thus the Fisher-Em algorithm is able to select key points without any prior knowledge.

The second level of cluster analysis, based on the transition matrix during each trial showed the existence of six different clusters. More specifically, Figure 3 highlights the preferred transitions exhibited by each emerging cluster. Interestingly, the group who showed the highest number of preferred transition (i.e. cluster 6) was associated with the learning group that did not receive any instruction. In that sense, this second level of cluster analysis allowed to highlight the use of temporary additional information during learning in order to modify the learning search strategy, namely by impacting the preferred transitions.

5 Perspectives

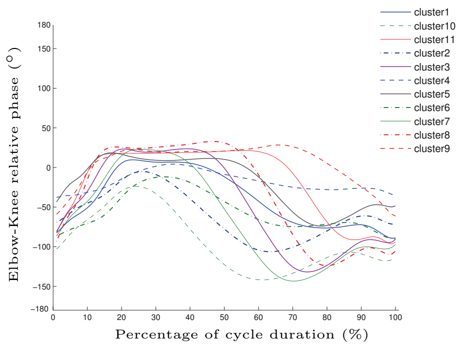
These preliminary experiments show that we can apply efficiently the Fisher-EM clustering on highly correlated features. Interestingly, the induced sparsity corresponds to key points of the coordination phase. Now, a qualitative work needs to be undertaken to qualify clusters of trials in term of learning condition and learning dynamics.

Table 1: Analysis of the BIC for the first level showing a plateau at 11 clusters

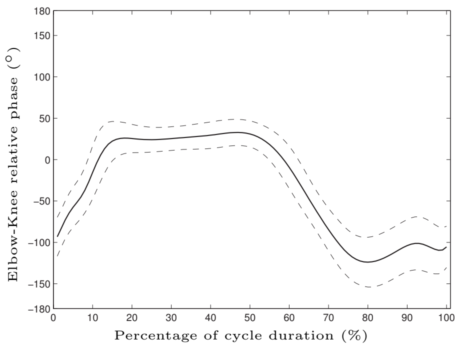
Number of clusters	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
BIC value ($\times 10^7$)	-1.23	-1.21	-1.18	-1.18	-1.15	-1.14	-1.13	-1.11	-1.08	-1.04	-1.05	-1.05	-1.07	-1.04	-1.04	-1.05

Table 2: Distribution (in %) of each cluster according to learning conditions

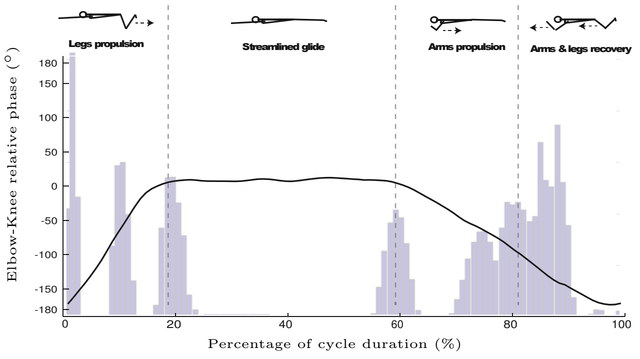
Cluster	Control	Analogy	Pacer	Prescription	Total	Cluster	Control	Analogy	Pacer	Prescription	Total
1	24.62	35.15	14.39	25.84	100	7	23.12	39.03	17.25	20.60	100
2	47.85	7.16	28.77	16.22	100	8	16.72	46.56	17.41	19.31	100
3	17.60	45.59	12.07	24.74	100	9	14.69	41.91	18.04	25.36	100
4	61.18	4.59	10.98	23.26	100	10	27.81	5.95	64.36	1.87	100
5	28.73	25.73	1.86	43.69	100	11	19.46	26.18	26.34	28.01	100
6	44.25	16.70	23.95	15.09	100						



a) Mean patterns of coordination for each cluster



b) Mean pattern for cluster 8 (black line), standard deviation (dotted line)



c) A typical coordination and superimposed induced sparsity

Fig. 2: First clustering level

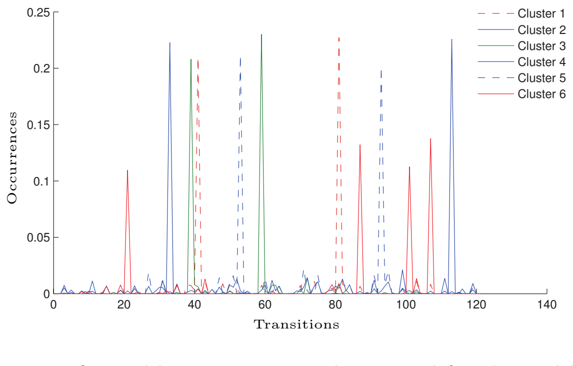


Fig. 3: Mean patterns of possible transitions within a trial for the 2nd level clustering, please note that there are $121 = 11 \times 11$ possible transitions as there is 11 clusters at first level

References

1. Scott Kelso, J.: Dynamic Patterns: the self-organization of brain and behavior. MIT Press (1995)
2. Müller, H., Sternad, D.: Decomposition of variability in the execution of goal-oriented tasks: Three components of skill improvement. Journal of Experimental Psychology: Human Perception and Performance **30**(1) (2004) 212–233
3. Nourrit, D., Delignières, D., Caillou, N., Deschamps, T., Lauriot, B.: On discontinuities in motor learning: A longitudinal study of complex skill acquisition on a ski-simulator. Journal of Motor Behavior **35**(2) (2003) 151–170
4. Chow, J.Y., Davids, K., Button, C., Rein, R.: Dynamics of movement patterning in learning a discrete multiarticular action. Motor Control **12**(3) (2008) 219–240
5. Gel'fand, I.M., Tsetlin, M.L.: Some methods of control for complex systems. Russian Mathematical Survey **17** (1962) 95–116
6. Bouveyron, C., Brunet, C.: Simultaneous model-based clustering and visualization in the fisher discriminative subspace. Statistics and Computing **22**(1) (2012) 301–324
7. Seifert, L., Leblanc, H., Chollet, D., Delignières, D.: Inter-limb coordination in swimming: Effect of speed and skill level. Human Movement Science **29**(1) (2010) 103–113
8. Seifert, L., Chollet, D.: A new index of flat breaststroke propulsion: A comparison of elite men and women. Journal of Sports Sciences **23** (March 2005) 309–320
9. Tagaki, H.: Differences in stroke phases, arm-leg coordination and velocity fluctuation due to event, gender and performance level in breaststroke. Sports Biomechanics **3** (2004) 15–27
10. Hamill, J., Haddad, J.M., McDermott, W.J.: Issues in quantifying variability from a dynamical systems perspective. Journal of Applied Biomechanics **16** (2000) 407–418
11. Komar, J., Chow, J.Y., Chollet, D., Seifert, L.: Effect of analogy instruction on learning a complex motor skill. Journal of Applied Sport Psychology (in press)
12. Bouveyron, C., Brunet, C.: Theoretical and practical considerations on the convergence properties of the Fisher-EM algorithm. J. Multivariate Analysis **109** (2012) 29–41
13. Bouveyron, C., Brunet, C.: Discriminative variable selection for clustering with the sparse Fisher-EM algorithm. Technical report, http://hal.archives-ouvertes.fr/hal-00685183