
Citation:

Adeyemo, V. (2023) Identification of Movement Patterns in Rugby Football League Using GPS Data: Algorithms and Applications [Online]. Leeds Beckett University. Available from:
<https://figshare.leedsbeckett.ac.uk/articles/thesis/Identification_of_Movement_Patterns_in_Rugby_Football_League_using_GPS_data_Algorithms_and_Applications/24306928/1> [Accessed 24 October 2023]. - [Link](#)

Link to Leeds Beckett University Research Data and Thesis Repository record:

[10.25448/lbu.24306928.v1](https://figshare.leedsbeckett.ac.uk/articles/thesis/Identification_of_Movement_Patterns_in_Rugby_Football_League_using_GPS_data_Algorithms_and_Applications/24306928/1)

Item Type:

Thesis

Licence:

CC BY 4.0

The aim of the Leeds Beckett University Research Data and Thesis Repository is to provide open access to the outputs and data from our research, as required by funder policies and permitted by publishers and copyright law.

The Leeds Beckett University Research Data and Thesis Repository holds a wide range of outputs and data, each of which has been checked for copyright, licenses and any relevant embargo periods have been applied by the Research Services team.

We operate on a standard take-down policy. If you are the author or publisher of an output and you would like it removed from the repository or believe there to be any issues with copyright please [contact us](#) and we will investigate on a case-by-case basis.

Identification of Movement Patterns in Rugby Football League using GPS data: Algorithms and Applications

Victor Elijah, ADEYEMO

A thesis submitted for the degree of

Doctor of Philosophy

School of Built Environment, Engineering and Computing

Leeds Beckett University

January 2023

Victor Elijah, ADEYEMO

Identification of Movement Patterns in Rugby Football League using GPS data: Algorithms and Applications

Keywords: Player Movement Profiling, Pattern Mining Algorithm, Rugby League Football and Sports Analytics.

Abstract

This PhD study is based on considering the sequential nature of match activity occurrences to understand the (physical) demand of games on players as well as uncover players' behavioural movement patterns during competitive matches. It involves quantifying players' external load (i.e., completed match activities) using movement patterns that provide information on how players accumulated external load in contrast to existing physical and technical-tactical performance indicators. The quantification of external load helps with team and player improvement, training specificity and even talent identification and recruitment. Elite players of rugby football league were considered as the participants of this PhD study because rugby league is a physically intense team sport where activities happen quickly and within a stipulated timeframe. Existing player movement profiling frameworks were investigated which revealed the use of a sequential pattern mining algorithm to extract movement patterns. However, the algorithm for extracting patterns in the existing player movement profiling frameworks is found to have limitations such as the identification of few movement patterns, providing only the longest common patterns within a cluster of movement sequences while it discards other interesting patterns. Furthermore, a review of sequential pattern mining algorithms revealed none of the existing algorithms is suitable for player movement profiling. The state-of-the-art algorithm for extracting closed contiguous patterns (i.e., CCSpan) does not scale well on large data as well as data

with long rows of sequences. Also, the CCSpan algorithm is without a parameter to define a maximal length of extract patterns which is useful as a sliding window. This PhD study's contributions to the body of knowledge include the development and optimisation of a novel pattern mining algorithm for extracting user-defined length of frequent closed contiguous patterns, called LCCspm (l-length closed contiguous sequential pattern mining), among others. An intrinsic evaluation of the LCCspm algorithm was considered in terms of speed and memory consumption performance measures. Results revealed it is four, seven or ten times faster than the state-of-the-art algorithm when tested on natural data in three different use cases. An extrinsic evaluation of the LCCspm algorithm reveals and validates that its movement patterns are best to profile players into playing positions when compared to other distinct types of obtainable movement patterns. Furthermore, this PhD study applied LCCspm and other artificial intelligence algorithms to discover signature movement patterns of elite rugby league players per playing position, identify (key) movement patterns as predictor variables for classifying players into playing positions, and later establish a set of movement performance indicator useful for assessing players' performance variability across playing positions. In the broader scope of computing, this PhD study contributes LCCspm algorithm as an advancement of pattern mining algorithms. The application of LCCspm can be extended to other sports domains and challenges, allowing for more accurate analysis and insights into player behaviour and team dynamics. Additionally, LCCspm can be applied in any field where the consecutiveness of items matters during the analysis of patterns.

Declaration

The candidate confirms that the work submitted is his own and that appropriate credit has been given where reference has been made to the work of others.

Victor Elijah, ADEYEMO

Acknowledgements

I acknowledge my God (Jehovah), my Lord and personal saviour (Jesus Christ) and the Holy Spirit for the privilege and all it entails to carry out this project.

Special thanks go to my supervisory team: Dr Anna Palczewska, Prof. Ben Jones and Dr Dan Weaving. I received immeasurably support and guidance in all aspects of the commencement and completion of this research.

I acknowledge the support of The Rugby Football League (RFL) towards data collection and availability. My sincere thanks to the staff members of the England Performance Unit and Leeds Rhino rugby football club for providing background knowledge of this project.

Special thanks must also go to my wife Oluwaseun, my daughter Joanne and the Adeyemo family. I received overwhelming encouragement and love throughout my time completing this project. I also thank my Mosaic church family as well as my friends for their tremendous encouragement and support.

Finally, I acknowledge my colleagues and administrative staff members at the Carnegie School of Sport, for sharing and providing feedback and support.

Publications and Presentations

Journals

- Victor Elijah Adeyemo, Anna Palczewska, Ben Jones, Dan Weaving and Sarah Whitehead. Optimising classification in sport: a replication study using physical and technical-tactical performance indicators to classify competitive levels in rugby league match-play, 2022, Science and Medicine in Football, DOI: 10.1080/24733938.2022.2146177
- Victor Elijah Adeyemo, Anna Palczewska, Ben Jones, and Dan Weaving. Identification of pattern mining algorithm for rugby league players positional groups separation based on movement patterns, 2023. arXiv preprint arXiv:2302.14058 *Submitted to Plos One Journal*
- Victor Elijah Adeyemo, Anna Palczewska, Ben Jones, and Dan Weaving. The use of match-based exact movement activities to classify elite rugby league players into positional groups, 2023, PREPRINT (Version 1) available at Research Square, <https://doi.org/10.21203/rs.3.rs-3424741/v1>. *Submitted to International Journal of Data Science and Analytics (Applications)*

Conference proceedings

- Victor Elijah Adeyemo, Anna Palczewska, and Ben Jones. “LCCspm: l-Length Closed Contiguous Sequential Patterns Mining Algorithm to Find Frequent Athlete Movement Patterns from GPS.” In 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 455-460. IEEE, 2021.
- Victor Elijah Adeyemo, Anna Palczewska, Ben Jones, and Dan Weaving. “Elite Rugby League Players’ Signature Movement Patterns and Position Prediction.”

In International Workshop on Machine Learning and Data Mining for Sports Analytics (MLSA). Cham: Springer Nature Switzerland, 2023.

Presentations

- LCCspm: LCCspm: l-Length Closed Contiguous Sequential Patterns Mining Algorithm to Find Frequent Athlete Movement Patterns from GPS - *presented at School of the Built Environment, Engineering and Computing Postgraduate Research Symposium, 2022.*
- The use of Movement Patterns to classify Elite Rugby League Players into Positional Groups - *presented at Carnegie School of Sport Postgraduate Research Symposium, 2022.*
- Identification of Movement Patterns in Rugby Football League using GPS data: Algorithms and Applications - *presented at School of the Built Environment, Engineering and Computing End of the Year Showcase, 2023.*
- Elite Rugby League Players' Signature Movement Patterns and Position Prediction - *presented at European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in 10th International Workshop on Machine Learning and Data Mining for Sports Analytics, Turin, Italy, 2023.*

Software and Source Code

- LFCS (l-Length Frequent Contiguous Sequence) Explorer [LFC \(2020\)](#)
- LCCspm (l-Length Closed Contiguous sequential pattern mining) Source code [Adeyemo \(2021a\)](#)
- Validation of the pattern mining algorithms for player movement profiling source code [Adeyemo \(2021b\)](#)

Contents

1	Introduction	1
1.1	Background and Motivation	1
1.2	Thesis Aim and Objectives	6
1.3	Data Collection and Ethical Approval	7
1.4	Main Contributions	7
1.5	Thesis Structure	8
2	Literature Review	11
2.1	Rugby League Football	11
2.2	Player Performance Analysis using Classification Approach	15
2.2.1	Identifying Key Variables in Classification Analyses	21
2.3	Frequent Pattern Mining in Sports	23
2.3.1	Athlete Monitoring	24
2.3.2	Players Movement Profiling	26
2.4	Sequential Pattern Mining Algorithms	29
2.4.1	Apriori-based Algorithms	32
2.4.2	Pattern Growth Algorithms	36
2.4.3	Constraints-Based Algorithms	38
2.4.4	Snippet Growth Algorithm	41
2.5	Chapter Summary	43
3	Methodology	45

3.1	Player Movement Profiling Frameworks	45
3.1.1	Sweeting Framework	47
3.1.2	Sequential Movement Pattern-mining (SMP) Framework . . .	49
3.2	Closed Contiguous Frequent Pattern Mining for Player Movement Analysis	53
3.2.1	Closed Contiguous Sequential Pattern Mining (CCspan) Algorithm	54
3.2.2	Movement Pattern Analysis	55
3.2.2.1	Classification Algorithms	56
3.2.2.2	Pattern Similarity Measurements	59
3.3	Player Movement Profiling	60
3.3.1	Signature Movement Patterns of RFL Playing Positions	61
3.3.2	RFL Playing Position Classification using Movement Patterns	61
3.3.2.1	Data Representation	62
3.3.2.2	Data Sampling	62
3.3.2.3	Classification Model Development and Evaluation .	63
3.3.2.4	Key Movement Patterns for Classification	65
3.3.3	Movement Performance Indicators (MPIs)	68
3.4	Chapter Summary	69
4	LCCspm: <i>l</i>-Length Closed Contiguous sequential pattern mining algorithm	70
4.1	<i>l</i> -Length Closed Contiguous Pattern Mining	70
4.1.1	Basic Concepts and Definitions	71
4.1.2	Candidate Generation and Search Space Pruning Approach . .	74
4.1.3	Pattern Closure Checking Approach	77
4.1.4	LCCspm Algorithm (Memory-Optimised)	78
4.1.5	LCCspm Algorithm (Time-Optimised)	83
4.1.6	LCCspm Complexity Analysis	86
4.2	Evaluating LCCspm Algorithm (Memory-Optimised)	87
4.2.1	Datasets and Experimental Setting	87
4.2.2	Results and Discussion	89
4.2.2.1	Small Sequential Data	89

4.2.2.2	Large Sequential Data	94
4.2.2.3	Men’s FIFA 2018 World Cup Match-event Data . .	99
4.3	Evaluating LCCspm Algorithm (Time-Optimised)	102
4.3.1	Dataset and Experimental Settings	102
4.3.2	Results and Discussion	103
4.4	Chapter Summary	106
5	Comparison of LCCspm with Existing Player Movement Profiling Frame-works	108
5.1	Experimental Method	108
5.1.1	Data and Processing	108
5.1.2	Pattern Mining Algorithms and Parameter Settings	110
5.1.3	Movement Pattern Validation	111
5.1.3.1	Jaccard Analysis	111
5.1.3.2	Overlap Movement Patterns	112
5.1.3.3	Separation of Players into Hookers and Wingers . .	112
5.1.3.4	Significant Movement Patterns for Hookers and Wingers Separation	113
5.2	Results	114
5.2.1	Jaccard Analysis	114
5.2.2	Overlapping Movement Patterns	115
5.2.2.1	Overlapping Movement Patterns Between Algorithms	115
5.2.2.2	Overlapped Frequent-50 Movement Patterns between Positions	119
5.2.2.3	Overlapped Movement Patterns between Positions per Algorithm	120
5.2.3	Separation of Players into Hookers and Wingers	122
5.2.3.1	Separation Analysis	122
5.2.3.2	Significant Movement Patterns for the Separation .	125
5.3	Discussion	128
5.4	Chapter Summary	130

6	Applications of LCCspm Movement Patterns in Rugby Football League	132
6.1	Experimental Method	132
6.1.1	Data and Processing	134
6.1.2	Signature Movement Patterns of RFL players	134
6.1.3	RFL Playing Position Classification using Movement Patterns	135
6.1.3.1	Key Movement Patterns for Classification	136
6.1.4	Assessment of Players' Performance Variability	138
6.2	Results and Discussion	139
6.2.1	Signature Movement Patterns of RFL Playing Positions	139
6.2.1.1	Centres	139
6.2.1.2	Five Eighths	141
6.2.1.3	Full Backs	141
6.2.1.4	Half Backs	142
6.2.1.5	Hookers	142
6.2.1.6	Loose Forwards	142
6.2.1.7	Prop Forwards	143
6.2.1.8	Second Rows	143
6.2.1.9	Wingers	144
6.2.2	RFL Playing Position Classification using Movement Patterns	144
6.2.2.1	Classification Based on Binary values	144
6.2.2.2	Classification Based on Relative Frequency values .	148
6.2.2.3	Key Movement Patterns for Classification	151
6.2.3	Assessment of Players' Performance Variability	162
6.3	Chapter Summary	168
7	Conclusions	171
7.1	Practical Implications	173
7.2	Summary of Contributions	174
7.3	Limitations and Future Works	175
	References	177
	Appendix A Ethical Approval and Informed Consent	193

Appendix B	Feature Selection Results	196
B.1	Correlation-based Subset Evaluator Technique on original relative frequency Data	196
B.2	Consistency-based Subset Evaluator Technique on original relative frequency Data	197
B.3	Correlation-based Subset Evaluator Technique on Oversampled Relative frequency Data	198
B.4	Consistency-based Subset Evaluator Technique on Oversampled Relative frequency Data	199

List of Figures

2.1	Rugby League Playing Positions (RFL)	12
2.2	Australian rugby league under-20 and senior NRL competition levels classification tree (Woods et al., 2018b)	18
2.3	England rugby league (forwards) academy and senior competition lev- els classification tree (Whitehead et al., 2021)	19
2.4	England rugby league (backs) academy and senior competition levels classification tree (Whitehead et al., 2021)	20
2.5	An example of significantly increasing duration pattern (Hrovat et al., 2015)	25
3.1	Outline of Research Method for this PhD study (Adeyemo, 2023). . .	46
3.2	Depiction of Sweeting et al. Movement Sequencing Framework (Adeyemo, 2023).	48
3.3	SMP Framework extracted from (White et al., 2021)	51
3.4	Example of 10Hz GPS Data (Adeyemo, 2023).	52
3.5	Random Forest Illustration RFI (2020)	57
3.6	MLP with one hidden layer (Pedregosa et al., 2011)	59
5.1	Workflow for Identifying the best sets of Movement Patterns (Adeyemo et al., 2023)	109
5.2	Overlapped movement patterns between the most frequent-50 LCC- spm and LCS patterns (Adeyemo et al., 2023)	116

LIST OF FIGURES

5.3	Overlapped movement patterns between the most frequent-50 AprioriClose and LCCspm patterns (Adeyemo et al., 2023)	118
5.4	Overlapped movement patterns between the most frequent-50 LCS and AprioriClose patterns (Adeyemo et al., 2023)	119
5.5	Overlapped movement patterns between most frequent-50 LCCsmp and LCS patterns per position (Adeyemo et al., 2023)	119
5.6	Overlapped movement patterns within positions as extracted by LCCspm (Adeyemo et al., 2023)	120
5.7	Overlapped movement patterns within positions as extracted by LCS (Adeyemo et al., 2023)	121
5.8	Overlapped movement patterns within positions as extracted by AprioriClose (Adeyemo et al., 2023)	122
6.1	Workflow for RFL Players' Position Profiling, Classification and Performance Variability Assessment (Adeyemo, 2023)	133
6.2	Signature Movement Patterns per Playing Positions (Adeyemo, 2023)	140
6.3	SHAP values of Top Fifty Movement Patterns Across and per Playing Positions (Adeyemo, 2023)	160
6.4	SHAP values of Top Fifty Movement Patterns per Playing Positions (Cont'd) (Adeyemo, 2023)	161
6.5	SHAP values of Top Fifty Movement Patterns per Playing Positions (Cont'd) (Adeyemo, 2023)	162
6.6	A Centre Player Performance Variability Assessment (Adeyemo, 2023)	165
6.7	A Winger Player Performance Variability Assessment (Adeyemo, 2023)	166
6.8	A Full-Back Player Performance Variability Assessment (Adeyemo, 2023)	166
6.9	A Half-Back Player Performance Variability Assessment (Adeyemo, 2023)	166
6.10	A Hooker Player Performance Variability Assessment (Adeyemo, 2023)	166
6.11	A Loose-Forward Player Performance Variability Assessment (Adeyemo, 2023)	167
6.12	A Prop-Forward Player Performance Variability Assessment (Adeyemo, 2023)	167

LIST OF FIGURES

6.13 A Second-Row Player Performance Variability Assessment (Adeyemo, 2023)	167
6.14 A Winger Player Performance Variability Assessment (Adeyemo, 2023)	167
A.1 University Ethical Approval	194
A.2 The Rugby Football League Inform Consent	195

List of Tables

2.1	Example of a set of Discrete Movement Sequences (Adeyemo, 2023) .	30
3.1	The movement descriptors and threshold assignment values (White et al., 2021).	52
3.2	Movement Unit and Character (White et al., 2021)	53
3.3	Confusion Matrix (Adeyemo et al., 2022)	64
4.1	Examples of types of frequent patterns extracted from Table 2.1 (Adeyemo, 2023)	73
4.2	FIFA 2018 Match Event Dictionary (Adeyemo, 2023)	88
4.3	Performance Evaluation Results on Small Player Movement Sequential Data (Adeyemo et al., 2021)	91
4.4	Small SDB Top-15 Patterns ($l = 7, \sigma = 5\%$) (Adeyemo et al., 2021) . .	92
4.5	Small SDB Bottom-15 Patterns ($l = 7, \sigma = 5\%$) (Adeyemo, 2023) . . .	93
4.6	Performance Evaluation Results on Large Player Movement Sequential Data (Adeyemo et al., 2021)	95
4.7	Large SDB Top-15 Patterns ($l = 7, \sigma = 5\%$) (Adeyemo et al., 2021) . .	96
4.8	Large SDB Bottom-15 Patterns ($l = 7, \sigma = 5\%$) (Adeyemo, 2023) . . .	97
4.9	Performance Evaluation Results on Men's FIFA 2018 World Cup (Adeyemo et al., 2021)	100
4.10	Men's FIFA 2018 World Cup Top-15 Patterns ($l = 29, \sigma = 50\%$) (Adeyemo et al., 2021)	102

LIST OF TABLES

4.11	Men's FIFA 2018 World Cup Bottom-15 Patterns ($l = 29$, $\sigma = 50\%$) (Adeyemo, 2023)	103
4.12	LCCspm Algorithm (Time-Optimised) Performance Evaluation Re- sults on Large Player Movement Sequential Data (Adeyemo, 2023) . .	105
5.1	Classification Algorithms Parameter settings (Adeyemo et al., 2023) .	113
5.2	Similarity of Movement Patterns per Algorithm (Adeyemo et al., 2023)	115
5.3	Overlapped patterns and support values between Top-50 LCCspm and LCS Frameworks (Adeyemo, 2023)	117
5.4	Classifiers' Separation Accuracies using various sets of Movement Se- quences (Adeyemo et al., 2023)	123
5.5	Logistic Regression Top 20 Important Patterns and Scores per Algo- rithm (Adeyemo et al., 2023)	126
5.6	Classification results on the Key Closed Contiguous Movement Pat- terns (Adeyemo, 2023)	127
6.1	Original and SMOTE Distribution of Players per Fixture per Playing Position (Adeyemo, 2023)	135
6.2	Classifiers' Separation Accuracies of RFL Playing Positions based on Binary Values (Adeyemo, 2023)	146
6.3	Classifiers' Separation Accuracies of RFL Playing Positions based on Relative Frequency Values (Adeyemo, 2023)	149
6.4	Classification results on original and its key movement patterns datasets based on Relative Frequency Values (Adeyemo, 2023)	153
6.5	Classification results on oversampled and its key movement patterns datasets based on Relative Frequency Values (Adeyemo, 2023)	156
6.6	Top Fifty Unique Movement Patterns and Classification Importance Scores (Adeyemo, 2023)	158
6.7	A set of Movement Performance Indicators and decoded Movement units (Adeyemo, 2023)	164

Chapter 1

Introduction

This chapter provides a brief overview of sports and sports analytics, it highlights the role of wearable devices and the use of performance indicators in a sporting context. It also introduces player movement profiling and its current challenges which motivate the research conducted in this PhD study.

1.1 Background and Motivation

The act of participating in sporting activities is carried out among billions of people across the globe (Perin et al., 2018). However, there is a category of people who participate in various sports as professionals. This set of people is elite within their respective sports, as they train and compete professionally among themselves with respect to the rules and regulations of their different sports. The sports industry (Goes et al., 2021; Morgulev et al., 2018) is neither immune nor retrogressive to the development and various applications of data, big data, and data science. Technology advancements had found their way into the sports industry in order to help stakeholders (e.g., players, coaches, managers and sports directors) optimally performed their respective responsibilities (e.g., injury prevention and performance improvement), thereby making sports analytics an interesting research area.

Sports analytics is a rapidly growing area under the broader scope of data science. It involves the development and or application of various computational, mathematical and/or statistical techniques, methods and algorithms (Goes et al., 2021; Morgulev

et al., 2018) to often analyse sport-data. In the field of sports science, researchers and practitioners are faced with several analytical problems including visualization (Hrovat et al., 2015), regression (Whiteside et al., 2016) and classification (Ayala et al., 2019; Van Eetvelde et al., 2021) among others. These analytical problems are primarily based on characterizing team-sport training and competitions (Weaving et al., 2019) towards understanding match demands on players (Colomer et al., 2020), detection of teams' tactics during competitive matches (Decroos et al., 2018), understanding the differences and similarities between player positional groups (Razali et al., 2017; Woods et al., 2018a) among others. Most of the research involving these analytical problems is conducted on players participating in team sports. Team sports (Garganta, 2009) refers to any game played by two teams who oppose each other, where players concurrently and directly interact among themselves to ensure the movement of a ball or similar object to score points and prevent the opposition from scoring, in accordance with the laid down rules of the game.

Regarding technological advancements, besides capturing match games using semi-automated cameras, micro-sensors (i.e., gyroscopes, accelerometers and magnetometers) containing global positioning systems (GPS) (Gabbett et al., 2011; Kupperman and Hertel, 2020) are now available and integrated to collect the sports activities performed by players during training or competitions. Through the use of these technologies, sporting activities are being recorded in different formats providing quantitative and qualitative data. These data exist in video file format, comma-separated values (.csv) files, text files and many other file formats. The existence of these data enabled analyses for the purpose of discovering hidden or unknown valuable information contained in the data (Mack et al., 2019; Sbrollini et al., 2019), which can be used in various decisive capacities. These decisive capacities assist in players' performance improvement (Hrovat et al., 2015), injury prevention (Emery and Pasanen, 2019) and talent identification and recruitment (Williams and Reilly, 2000) among others. It is noteworthy to emphasise at this point that the obtained sports data are from athletes considered to be experts and largely rewarded to perform at their respective individual optimal levels.

In sports, the use of data collected via portable GPS units has become pivotal to quantifying completed activities and movements (i.e., external load) performed by team sports players. Physical indicators (as variables) are derived from GPS data (Gab-

bett et al., 2011; Kupperman and Hertel, 2020) and used to quantify the external load completed by players especially during competitions as one of the means to characterize matches. These physical indicators of player movements (Weaving et al., 2019; Whitehead et al., 2019) are reported in volumes, duration-based volumes and or as discretized threshold values (e.g., average match speed ($\text{m} \cdot \text{min}^{-1}$), 60 seconds- average running speed ($\text{m} \cdot \text{min}^{-1}$), total distance covered (m)) and only account for the aggregated accumulation of external load. GPS data also provide captured trajectory data of players' positions (i.e., longitude and latitude). Similarly, data from semi-automated cameras and video are used to derive technical-tactical indicators (Kupperman and Hertel, 2020; Whitehead et al., 2018) (as variables) to quantify the tactical and technical events completed by players as another means to characterize matches. These type of indicators simply reports the counts of technical-tactical events e.g., pass, goal, kick, shot and reception.

Sports practitioners widely used physical and technical-tactical indicators as the means to ensure training prescription aspects (such as small-side games) replicate the characteristics of competitive matches. However, physical indicators (Weaving et al., 2019; Whitehead et al., 2019) are simply numeric descriptors of a single accumulated external load, often chosen subjectively, non-specific (i.e., non-individualized) and they typically aggregate the information across the whole match with little to no context on how players accumulated the quantified external load. For example, the “total distance covered” physical indicator can hypothetically provide information such as players A and B covered a total distance of 11km respectively, within a match or at two different matches. However, the physical indicator (i.e., total distance covered) does not provide information on how these players accumulated the quantified external load. Similarly, technical-tactical indicators count single technical-tactical (Adeyemo et al., 2022; Whitehead et al., 2021) events while they do not provide the activities that happened before (and after) the counted event. Although sports big data contain external load performed by players per fixture or during training that naturally happened in sequence, the majority of the existing methods for analysing sports data are based on physical and technical-tactical performance indicators that completely disregard the sequence of external load. It is important to consider the sequential order of quantified external loads because it explains how external loads were accumulated and also enables the derivation of exact match characteristics useful in training programme

designs. The analysis of sports big data with respect to the sequence of external load is important because it will also provide more granular information, for example, how players A and B covered a total distance of 11km respectively will be uncovered or explained. Hence, player movement profiling as an emerging and important area of research in sport science offered an alternative view to characterizing team sports competitions and training.

Player movement profiling (Sweeting et al., 2014, 2017) is now an important area in sports analytics as its application will expand the boundary of discovered knowledge. It is premised on the science of uncovering how players accumulate external loads and how technical-tactical events are completed through the identification of the behavioural movement patterns (i.e., group of movement activities) and identification of the series of activities leading to an event as performed by players. Basically, players complete and accumulate external load by performing some combinations of movement activities including walking, jogging, running or sprinting while they accelerate or decelerate and make some changes in their directions. Also, technical-tactical events are usually not performed by players in isolation. Uncovering how players accumulate external load during matches and identifying the group of frequent events leading to an interesting technical-tactical event(s) will certainly replicate the (exact) characteristics of competitive matches better than physical and technical-tactical indicators. At the time of this research commencement, only two studies (Sweeting et al., 2014, 2017) profiled the movement patterns of team sport players by finding frequently occurring movement patterns but there is a gap for further investigation (especially in regards to the algorithm for extracting patterns). Theoretically, player movement profiling will provide practitioners with information for conditioning elite players for match games, preparing junior-elite players for elite competitions, opposition analysis, profiling players toward talent identification and development and events or movement patterns leading to injury among others. In general, player movement profiling has the potential of improving all applications of characterizing competitive matches.

Data mining (Pascu, 2018) and pattern mining (Nijssen, 2013) have been successfully applied for analysing sequential data in different fields such as genome sequencing (Rashid et al., 2012) and large vehicle trajectory analysis (Bermingham and Lee, 2020) among others. The pattern mining area of data mining had recorded the development of various algorithms for finding patterns from a set of sequences. These

algorithms are broadly categorised (Mabroukeh and Ezeife, 2010) into frequent itemsets mining (for associative rule mining) and sequential pattern mining. The algorithms for mining frequent itemset and associative rules (Fournier-Viger et al., 2017) involve finding groups of items that share associations or correlations among themselves wherein item adjacency is not considered. Sequential pattern mining algorithms (Fournier-Viger et al., 2017) are focused on discovering recurring subsequences or patterns from very large sets of sequential data whose items must be ordered sequentially. The fundamental difference between both categories of pattern mining algorithms is that frequent itemsets mining and associative rule mining algorithms do not consider the order of events whereas sequential pattern mining algorithms maintain the order of events.

Due to the sequential nature of players' movement activities and technical-tactical events on the sporting field, the application of sequential pattern mining algorithms fits player movement profiling in team sports. It would avail the identification of players' behavioural movement patterns containing groups of activities that occurred in sequential order as well as sequential events within a match that led to a technical-tactical event. Concisely, the application of a sequential pattern mining algorithm can produce movement patterns that quantify players' external load sequentially and provide information on how external loads were accumulated. However, the generation of false candidate patterns (Han et al., 2001; Srikant and Agrawal, 1996) serves as one of the strongest limitations of existing sequential pattern mining algorithms. False candidate patterns are patterns generated by sequential pattern mining algorithms which do not exist in a given set of sequences. Other limitations (Fournier-Viger et al., 2017; Mabroukeh and Ezeife, 2010) include multiple scans of a set of sequences for mining patterns, large numbers of joins, large-sized index lists of generated candidate patterns which cost space and time, consideration of all possible occurrences of frequent subsequences, expensive recreation of sets of sequences projections, time and memory inefficiency and redundant frequent patterns among others. These highlighted limitations and the need for a quick turn-around time for the extraction movement patterns based on a predefined sliding window in a real-time application motivated this project.

This project is motivated to formulate, develop, validate and apply a novel computational method for player movement profiling by identifying movement patterns of a physically intense team sport (Gabbett et al., 2010; Glassbrook et al., 2019) - Rugby

Football League (RFL). It is an interdisciplinary research between Leeds Beckett University schools of “Built Environment, Engineering and Computing” and “Carnegie School of Sport”. Elite RFL players were considered as the participants of this PhD study because rugby league is a physically intense team sport where activities happen quickly and within a stipulated timeframe. Players participating in RFL matches often wear a portable GPS unit for collecting GPS data. The collected GPS data per fixture are mainly characterised by a succession of movement activities performed by players which are or can be recorded and or represented by sequence(s). The quantification of RFL players’ external loads by extracting sequential movement patterns seems plausible to take advantage of the sequential occurrences of movement activities. Therefore, this research explored player movement profiling by identifying movement patterns in rugby league from GPS data on the grounds that those patterns are consecutive, frequent but non-redundant and that the maximal length of extracted movement patterns can be specified.

1.2 Thesis Aim and Objectives

The aim of this thesis is to propose and enhance methods to identify movement patterns from rugby football league players’ GPS data. The required objectives to achieve this aim are to:

- Investigate (the application of) existing methods for profiling team sport players’ movements as well as existing sequential pattern mining algorithms to uncover their suitability for player movement profiling;
- Develop and optimise an algorithm for extracting frequent but non-redundant consecutive (i.e., closed contiguous) patterns and conduct a performance evaluation of the same with the state-of-the-art closed contiguous algorithm;
- Compare the proposed algorithm with existing player movement profiling frameworks to identify the best type of movement patterns for profiling player movement;
- Apply LCCspm algorithm to extract movement patterns to identify signature movement patterns of each RFL playing position, classify players into all RFL

playing positions while discovering sets of key movement patterns for such classification; and establish “Movement Performance Indicators” useful for players’ performance variability assessment and visualisation.

1.3 Data Collection and Ethical Approval

This study received the approval of the University Ethics Committee and obtained written informed consent (See Appendix A) from the organisation representing all participants (i.e. elite England rugby league players). The Global Positioning Systems (GPS) data were obtained (from clubs through the Rugby Football League) and are not available publicly. 10Hz GPS data of elite male Rugby Football League players were collected via wearable sensors (Catapult S5, Catapult Innovations, Melbourne, Australia) worn during matches during the 2019 and 2020 seasons. Other data were available and assessed from a public repository.

1.4 Main Contributions

This PhD study contributes to the field of computing and sports analytics by developing and introducing a novel sequential pattern-mining algorithm called LCCspm (*l*-Length Closed Contiguous sequential pattern mining algorithm). LCCspm addresses the limitations of existing methods and fulfils the criteria for a suitable algorithm in player movement profiling. The LCCspm algorithm utilizes efficient data structures, including dictionary and 2-tuple representations, along with a snippet-growth mechanism, to generate and store frequent closed contiguous patterns. A unique technique called “inverse characteristic” is employed to achieve lossless compression of frequent patterns.

Through extensive experiments on real-life datasets, the performance of LCCspm is evaluated in terms of memory consumption, execution runtime, and scalability. The results demonstrate that LCCspm outperforms existing algorithms by quickly identifying frequent closed contiguous patterns while consuming less computer memory. Furthermore, the scalability experiment reveals that LCCspm effectively handles small and large-sized sets of sequences, even with extremely long sequences, and accommodates low support thresholds set by the user.

The practical application of LCCspm for player movement profiling is validated in the context of player position classification in rugby league. LCCspm-based movement patterns were successfully utilized to classify players into tactically different positions, showcasing its efficacy and accuracy, when compared to all other obtainable movement patterns.

Additionally, various methods to apply LCCspm are demonstrated, including identifying signature movement patterns, extracting movement patterns and investigating movement pattern values for classification modelling while identifying key movement patterns, and developing Movement Performance Indicators (MPIs) to assess and visualize players' performance variability.

Overall, this study contributes novel advancements in algorithm design, efficient pattern generation and storage, lossless compression techniques, performance evaluation, and practical applications of LCCspm in player movement profiling. These findings significantly enhance the understanding and analysis of player behaviour in competitive matches, with potential implications for sports performance optimization and decision-making strategies.

1.5 Thesis Structure

This PhD thesis is arranged into seven chapters and two appendices:

- Chapter 2 fulfils the first objective of this PhD study. This chapter presents an overview of the data, its sources, and its availability in sports alongside the need for advanced analytics algorithms and methods to process such data. A survey of various applications of frequent pattern mining algorithms and machine learning algorithms in team sports is presented. A concise history and overview of rugby league football are presented as well as thoroughly identifying interesting research areas of the sport. The application of frequent sequential pattern mining algorithms for player movement profiling and the classification of players into positional groups in rugby league is identified as emerging research areas. A review of the existing player movement profiling methods was conducted with respect to the first objective of this PhD study. Also, four categories of sequential pattern mining algorithms are reviewed - which justified the need for developing

a novel algorithm for player movement profiling.

- Chapter 3 present the methodology of this PhD study. An overall experimental framework of the study was presented and discussed which detailed the methods implemented in subsequent chapters. Machine learning supervised algorithms that are implemented for classification modelling tasks in sports and categories of predictor variables are discussed. The methods and techniques involved in the processes of classification modelling are also discussed.
- Chapter 4 propose the novel l -length closed contiguous sequential pattern mining algorithm, with respect to four criteria of a suitable algorithm for player movement profiling, and to fulfil the second objective of this PhD study. Basic concepts and definitions of sequential pattern mining are presented. A novel sequential pattern mining algorithm was proposed and optimised. Also, evaluations of LCCspm algorithm variants were conducted to test their performance for extracting frequent closed contiguous patterns, in the bid to satisfy the fourth objective of this PhD study. The algorithm and these performance results were presented at International Conference on Machine Learning and Application (ICMLA) and published in an online proceeding by The Institute of Electrical and Electronics Engineers (IEEE) (Adeyemo et al., 2021).
- Chapter 5 focuses on identifying the best type of movement patterns for player movement profiling, in the bid to fulfil the third objective of this PhD study. A comparative analysis of patterns generated by LCCspm against patterns generated by other algorithms to profile the movement of rugby league players into two tactically different playing positions was conducted. The results are currently published in arxiv preprint (Adeyemo et al., 2023) and submitted to a Plos One journal.
- Chapter 6 focuses on the three applications of LCCspm movement patterns in RFL. This fulfils the fourth objective of this PhD study. LCCspm was applied to discover the signature movement patterns of each RFL playing position. Also, LCCspm was applied to extract movement patterns useful as predictor variables to classify rugby league players into nine playing positions. Investigation of

appropriate movement pattern values for classification modelling was also carried out. Sets of key and significant movement patterns useful for classifying rugby football league players into nine playing positions were also identified via further experiments. Additionally, a set of movement patterns was identified and established as a movement performance indicator to assess players' performance variability over participated matches. The results of this chapter are submitted to the 10th IEEE International Conference on Data Science and Advanced Analytics'2023 (Applications Track) and the 10th Workshop on Machine Learning and Data Mining for Sports Analytics, 2023.

- Chapter 7 presents a summary of this PhD study. It highlights the contributions of each chapter and shows how the objectives of the PhD study were met. The practical implications of this PhD study were also highlighted, limitations were presented and future directions for this body of work were discussed.

Literature Review

The aims of the literature review are, to: (1) provide a concise history and characteristics of rugby league, uncover its data sources and availability, critically evaluate the current knowledge base for characterising match demands on rugby league players (relating to training programmes for player development, talent identification, and performance variability assessment); (2) investigate the use of various machine learning frameworks in team sport towards identifying interesting and emerging classification problems in rugby league and common predictor variables; (3) investigate the current state of applying pattern mining algorithms in all sporting contexts; and (4) discuss and provide critical analysis of existing sequential pattern mining algorithms to justify the need for a new sequential pattern mining algorithm for the identification of player movement patterns.

2.1 Rugby League Football

Rugby league originated in the north of England during the late 19th century ([Gardner et al., 2015](#)) and is played across various competition levels. It is popular and more established in countries ([Brewer and Davis, 1995](#); [Meir et al., 2001](#)) such as (in no particular order) the United Kingdom, Australia, France and New Zealand. Rugby league is a team sport where each opposing team fields thirteen players (and four interchanges), broadly categorized ([Read et al., 2017](#); [Till et al., 2017](#)) into two positional groups (i.e., 'forwards' and 'backs') or four subgroups (of nine individual playing po-



Figure 2.1: Rugby League Playing Positions (RFL)

sitions) which are pivots (hooker, stand-off, scrum-half or half-back), outside backs (fullback, centre, winger), back row (loose or lock forward, second row) and props. Figure 2.1 illustrates rugby league players playing positions. The rugby league phase of play allowed no more than six tackles (prior to scoring a try) before handling the ball to the opposition. It is played over eighty minutes (forty-minute halves) at an elite level and involves basic movements such as passing, kicking and tackling. Importantly, the ball played cannot be passed forward but can either be kicked downfield or carried forward. Rugby league is played across amateur (Gabbett, 2000), semi-professional (Gabbett, 2002) and professional (Johnston et al., 2014) competition levels.

Rugby league as described by (Geeson-Brown et al., 2020; Till et al., 2017), is a collision team sport in which matches heavily involve a complex interaction of players' technical, tactical, cognitive and particularly physical qualities. Physically, a typ-

2.1 Rugby League Football

ical rugby league football match requires players to complete frequent bouts of high-intensity activities (Gabbett et al., 2010; Glassbrook et al., 2019) (e.g., jumping, side-stepping and sprinting) separately by lower intensity activities (Whitehead et al., 2018) such as standing, repositioning and walking. Based on competition levels (Gabbett et al., 2012) and playing positions (Austin and Kelly, 2013), players often cover a total distance of 4000m up to 8000m, where up to 1000m high-speed distance is reported by (Waldron et al., 2011) to be often covered. In the presence of enormous high-speed running demands (Johnston et al., 2014), collision and wrestling bouts (Geeson-Brown et al., 2020; Till et al., 2017), rugby league players are reported to always be in need of highly developed and conditioned muscular strength, speed and power to consistently repeat these actions across an 80-minutes match. As such, the published research conducted in the rugby league as reviewed by studies (Gabbett, 2005; Johnston et al., 2014) is focused on investigating the physical, technical and tactical demands (among others) of competitive matches to understand the physiological, injury epidemiology, strength and conditioning, psychological and performance challenges faced by rugby league players. To this end, data are collected to conduct research providing solutions to various challenges faced by rugby league players.

Data in rugby league are often collected by sports scientists in various files and formats and mostly based on different study designs. Majority of the research (Adeyemo et al., 2022; Dalton-Barron et al., 2021; Gabbett et al., 2011; Glassbrook et al., 2019; Kupperman and Hertel, 2020; Weaving et al., 2019; Whitehead et al., 2018, 2019, 2021) that quantified the characteristics of rugby league competitive match reported physical indicators extrapolated from wearable micro-sensors (Guo et al., 2018) (e.g., Catapult GPS) and technical-tactical indicators expertly coded by video analysts extracted from match videos. These indicators as variables are often reported per the whole (Whitehead et al., 2019), per half (Glassbrook et al., 2019), per peak period (Weaving et al., 2019) and even averages per minutes (Dalton-Barron et al., 2021). Physical, technical-tactical and anthropometric indicators as predictor variables are widely used in literature to prepare athletes for transition between levels (Bradley and Ade, 2018), identification of skills for talent recruitment and development (Hughes and Franks, 2008), injury prevention and recovery (Gabbett et al., 2010), opposition analysis (Colomer et al., 2020) and classification of players into competition levels within positional group (Whitehead et al., 2021). Largely, the essence of all aspects of

science in rugby league is the characterisation of players, teams, training and competitions through quantifiers (i.e., anthropometric, physical, technical, tactical) towards understanding match demands on players and teams, players' progress or fitness or performance monitoring, and classification of players into groups (e.g., positional group, playing positions, competition level and injury status).

Player movement profiling and the separation of players into playing positions are emerging and interesting research areas in rugby league. Player movement profiling (Sweeting et al., 2014) has become an emerging research area because it offers an alternative view to understanding match demands by concurrent evaluation of the speeds, changes in speeds and turning angles completed by players at any point in time. It provides movement patterns that explain how external loads are accumulated by providing exact completed movement activities sequentially performed by players. It can provide viable solutions to various research aspects of rugby league such as injury epidemiology (by uncovering movement patterns that often led to injury incidents), strength and conditioning (by enhancing training specificity) and performance analysis (through external load evaluation and performance variability assessment) among others. On the other hand, the separation of players into playing positions (Woods et al., 2018a) is another interesting area because it offers (among other benefits) the identification of the required physical, technical and or even tactical skills required of a player belonging to a particular group. This is extremely useful in talent identification (Williams and Reilly, 2000), as it assists in the early recognition and recruitment of young players to a particular position. It also offers sports organisations effective financial investment (Pion et al., 2017) by channelling available resources on the development of the identified talents having the skills required to perform in a particular playing position. As match demands on player is based on playing positions, this PhD study is motivated to explore rugby league players' movement profiling and positional group separation. This will ultimately identify sets of movement patterns that characterise competitive matches, separate rugby league players into playing positions and assess performance variability.

2.2 Player Performance Analysis using Classification Approach

Machine learning is an aspect of computational algorithms comprising methods that imitate human intelligence through learning from the surrounding environment. It broadly consists of unsupervised (useful for clustering analysis) and supervised (useful for classification modelling) algorithms. One of the sporting contexts of applying machine learning algorithms is classification modelling. The goal of applying machine learning classification algorithm (Collins et al., 2022; Woods et al., 2018a), is to learn from data to develop a model that can predict outcomes or status or groups or levels. Sports data are often analysed to explore the difference between sporting aspects but this differs from applying machine learning classification algorithms. The difference is mainly due to the development of the predictive model(s) that can classify previous unseen observations of sporting aspects into pre-defined target variables. One such prominent sport-related problem, according to Van Eetvelde et al. (2021), is to predict if a player is injured or not (Ayala et al., 2019) as well as predicting injury types (Whiteside et al., 2016) using classification algorithms. Other sport-related classification problems are to classify players into competition levels (Adeyemo et al., 2022; Whitehead et al., 2021) and player positions (Woods et al., 2018c).

The application of machine learning classification algorithms in the sporting context of separating or classifying or predicting the players' playing positions or playing positions is beginning to garner the interest of sports scientists. Williams and Reilly (2000) expressed that it assists in the early recognition and recruitment of young talents within a particular sport with the potential to excel. It also offers sports organisations effective financial investment (Pion et al., 2017) by channelling available resources on the development of the identified talents. Talent identification, according to Vaeyens et al. (2008), provides methods and is mostly focused on predicting the success of young players in adult competitions. However, machine learning classification algorithms are applied to develop a predictive model towards identifying talents (Vaeyens et al., 2008) as a means to improve traditional methods (Till et al., 2013) (e.g., computer-based match analysis, physical conditioning and cross-sectional approach). Nowadays, sports organisations are beginning to use talent identification means and methods to identify players who have unique attributes of a predefined positional group

2.2 Player Performance Analysis using Classification Approach

or playing positions. As such, machine learning classification algorithms are now applied to players' data to classify players into positions and analyse for positional group unique attributes toward talent identification.

One of the two examples of employing machine learning classification algorithms to classify players into positions was carried out by Woods et al. (2018c). Elite junior Australian footballers were classified into four common playing positions (i.e. forward, defence, midfield and ruck) based on technical skill indicators. Technical skill data of a final two hundred and eleven junior players that competed in national U18 championships, representing teams from each eight state academies that participated in sixteen championship games were acquired by the study. Classification data was developed which was made up of six hundred and eighty observations (i.e., three hundred midfield observations; forty-one ruck observations; one hundred and sixty-eight defence observations; and one hundred and seventy-one forward observations) and twelve technical skill indicators. The Linear Discriminant analysis (LDA), random forest and PART decision list machine learning classification algorithms were implemented for the multi-class predictive modelling. Classification accuracy and confusion matrix were used to evaluate the performance of each multi-class predictive model. Homogeneity in players across playing positions was revealed as two of the machine learning classification algorithms (random forest = 52.61% accuracy and LDA = 56.8% accuracy) fared poorly in the classification of players. Nonetheless, the PART decision list accuracy produced an accuracy of 70.1%. This indicates that the given set of discrete technical skill indicators can partially separate players into playing positions and should not be used in isolation. Also, produced the sets of rules produced by the PART decision list algorithm for classifying players into the common four playing positions can be used alongside subjective evaluations in practice.

Razali et al. (2017) proposed a talent identification framework for assigning players to different playing positions in football. The core part of the framework focused on predicting football players' positions based on twenty-four technical, mental and physical skills. The *sweeper*, *wing backs*, *right/left backs*, *defensive midfielders*, *centre midfielders*, *wingers*, *attacking midfielders*, *wide midfielders*, *secondary strikers* and *forwards* positions were considered to cover the varieties of football playing positions. Data of one hundred players between the age of fifteen to seventeen were collected from Football Player Information System managed by Bukti Jalil Sports School (BJSS)

2.2 Player Performance Analysis using Classification Approach

and used for the classification analysis. Three classification algorithms (i.e., Decision tree, Bayesian networks and nearest neighbour) were utilized to predict the appropriate playing positions of the football players. The leave-one-out validation technique was used to fit the classification models whose performances were measured using the accuracy metric. The study was able to establish that the combination of technical, mental and physical skills as independent variables was able to predict the football players' positions at Bayesian network accuracy of 99%, Decision tree accuracy of 98% and Nearest Neighbor accuracy of 97%. However, the (type of) skills that were significant to the excellent prediction of the predictive models were not identified.

The classification of players into competitive levels is prominent in rugby league studies as it also helps with talent development and enhancement of training specificity. In rugby (Woods et al., 2018b), playing pathways are developed offering longitudinal player development opportunities to identified junior talents. The pathway helps with the development of junior talents' multidimensional (i.e., physical, technical, cognitive and tactical) qualities and it is designed to help them to become elite players. To bring out success, junior talents participate in competitive matches within specific playing pathways and there is a need to understand the differences between competition levels. Understanding the differences in competition levels is important (Whitehead et al., 2021) because young players can participate in a higher competition level while training at their contracted level. Also, young players are ultimately required to progress to senior competition (Adeyemo et al., 2022) and can sometimes replace injured senior players.

Woods et al. (2018b) uncovered the game-play characteristics between two rugby league competitive levels. By adopting a longitudinal observational study design, twenty-six rounds of elite youth (i.e. under-20) and senior Australian National Rugby League (NRL) data were respectively obtained from a publicly available source. The under-20 competition played one hundred and eighty-six games while the NRL competition had one hundred and eighty-nine games recorded. For under-20 players, three hundred and seventy-two observations were generated and labelled. For senior NRL players, three hundred and seventy-eight observations were generated and labelled. Both competition-level observations were combined to form the input data for classification modelling. The observations were described based on twelve technical indicators used as predictor variables. A conditional interference classification tree algorithm

2.2 Player Performance Analysis using Classification Approach

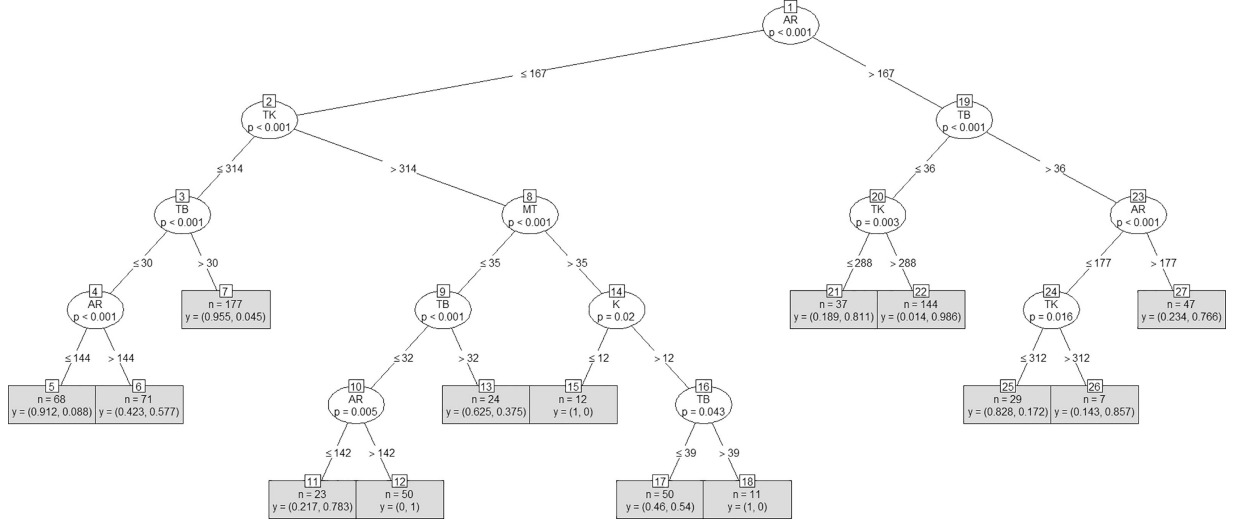


Figure 2.2: Australian rugby league under-20 and senior NRL competition levels classification tree (Woods et al., 2018b)

was utilized to develop a model that classified players' observations into competition levels. The model was further analysed to reveal the predictor variables (i.e., technical indicators) that are most capable of explaining both competition levels. Meanwhile, collinearity between two predictor variables (i.e. all runs and all run metres) was detected and the latter was excluded from the classification modelling. The classification model was able to correctly classify two hundred and ninety-three under-20 observations (i.e., 79% accuracy) and three hundred and fifty senior NRL observations (i.e., 93% accuracy). Five of the final eleven predictor variables were used by the classification tree model (Figure 2.2), where “all runs”, “tackles” and lowered number of “tackle breaks” helped in the correct classification of senior NRL observations relative to under-20.

Whitehead et al. (2021) suggest that Woods et al. (2018b) are limited in their classification of rugby league players into playing level because only technical indicators were considered as predictor variables. Hence, the study included both physical and technical-tactical performance indicators to classify rugby league players into two levels (i.e., academy and senior). GPS data of a total of thirty-one male senior players and another total of forty-one male academy players of an England rugby league club that participated in the European Super League across two competitive seasons were collected. Three hundred and twenty-five senior players and two hundred and fifty-

2.2 Player Performance Analysis using Classification Approach

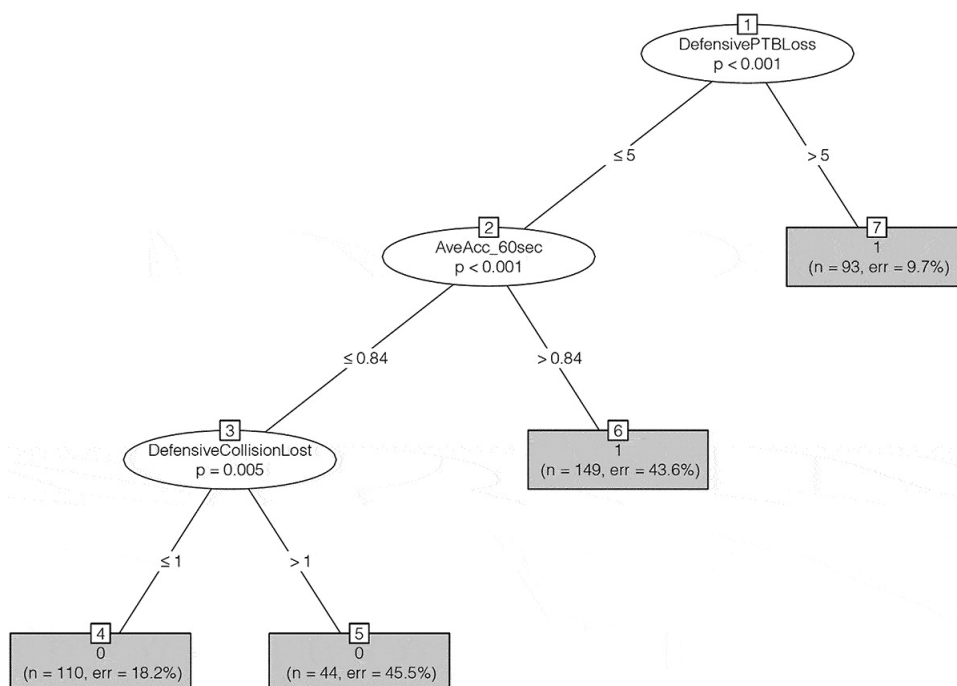


Figure 2.3: England rugby league (forwards) academy and senior competition levels classification tree (Whitehead et al., 2021)

nine academy players' observations were separated into "forwards" data. Another two hundred and forty senior players and two hundred and twenty-one academy players' observations were separated in "backs" data respectively. Random forest classification algorithm and conditional interference tree were used for classification modelling and a train-test split (70:30) technique was utilized for model development and testing. Random forest was first used to identify key predictor variables (based on its inbuilt feature importance method), the results of which were used to generate new data subsets for each positional group. Another random forest model was developed on the newly generated data (for each positional group) to classify players' match observation into playing levels. Afterwards, the conditional interference tree was applied to provide information on how the key predictor variables interact at each competition level. From "forwards" data, four predictor variables were identified as key variables and academy players were correctly classified at 62% accuracy while senior players were correctly classified at 72% accuracy (i.e., 68% overall accuracy).

From "backs" data, nine key performance indicators were identified which pro-

2.2 Player Performance Analysis using Classification Approach

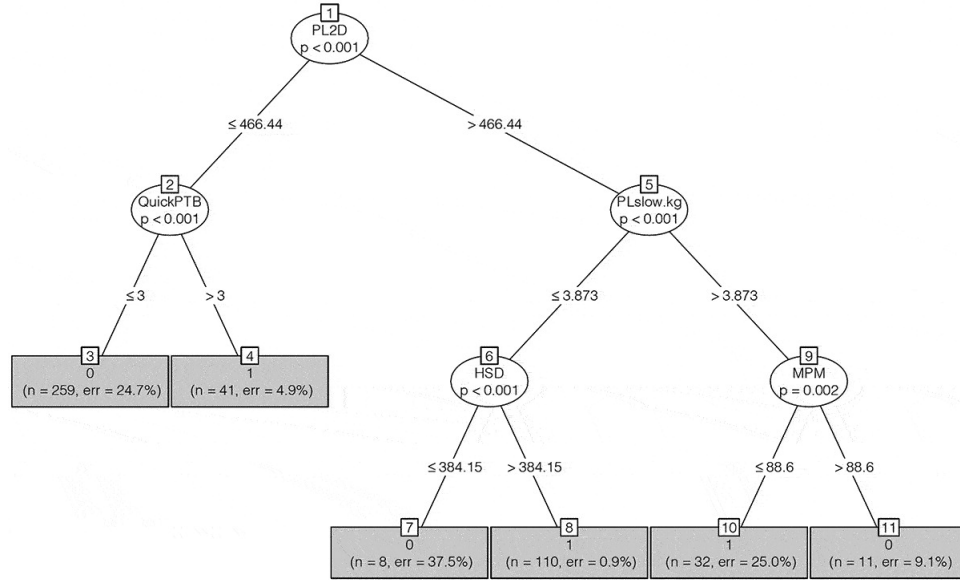


Figure 2.4: England rugby league (backs) academy and senior competition levels classification tree (Whitehead et al., 2021)

duced the random forest model that correctly classified 79% of the academy observation and 86% of the senior observations (i.e., 83% overall classification accuracy). A conditional interference model was grown on the “forwards” key predictor variables data which produced an overall accuracy of 64% (similar to random forest’s overall accuracy). However, three of the four variables were present in the tree (Figure 2.3). Likewise, another conditional interference model was grown on the “backs” key predictor variables data which produced an overall accuracy of 78% (slightly lower than random forest’s overall accuracy. Five out of the nine “backs” key predictor variables were present in the conditional interference tree (Figure 2.4).

Overall, the application of machine learning classification algorithms to predict players’ positions is in its infancy but it is greatly useful for uncovering qualities required of a player belonging to a specific positional group. Such qualities, if and when identified, can further help in talent identification and recruitment as well as enhance training programmes for each group. Studies had implemented various classification algorithms to predict players’ positions in different team sports with excellent accuracy, however, the identification of significant variables for predicting players’ positions is currently unexplored. In rugby league literature, playing-level classification

2.2 Player Performance Analysis using Classification Approach

is well-researched but the classification of players into playing positions is yet to be explored despite its immense benefits. Physical, technical and tactical performance indicators, as well as problem-specific variables (e.g., injury incidence), are derived and used as predictor variables for classification analyses. Therefore, the classification of rugby league players into positions requires exploration based on its immense usefulness in practice. More so, the review of related studies revealed the various types of classification algorithms with different learning methods for developing classification and or prediction models. Various strategies (such as repeated leave-one-out, train-test split and cross-validation) are employed to develop classification models. Accuracy (overall), AUC, confusion matrix and other measures are used to evaluate the performance(s) of classification models. Also, some (tree-based and or Random Forest) classification algorithms are preferred because of their inbuilt method to visualise key predictor variables.

2.2.1 Identifying Key Variables in Classification Analyses

In sports science, it is common for research designs that aim to address a classification problem to include multiple predictor variables. In sports literature, the predictor variables are often derived from demographic, psychological, neuromuscular measurements, training load variables, player perceived performance, genetic markers, anthropometric measurement, physical fitness, workload features from GPS, training intensity, technical skills, physical skills and mental skills data. Therefore, it becomes important to evaluate the construct validity and reliability of each predictor variable included before analysis. Often, researchers and practitioners are still left with high dimensional and colinear variables following this process.

To overcome the multidimensional and multicollinearity of predictor variables (i.e., identify key or significant predictor variables), studies typically conduct multiple univariate analyses by investigating each predictor and target variable values separately. For example, [Gabbett \(2013\)](#) investigated the difference in external loads among rugby leagues players across two different competitive levels (i.e., the National Youth Competition and National Rugby League) using a repeated-measures analysis of variance on physical performance indicators. However, such an approach is limited as it does not consider the covariance of the data and the multiple models produced could in-

2.2 Player Performance Analysis using Classification Approach

crease classification models' error rates.

Alternatively, machine learning feature importance methods can be used to identify key predictor variables (Thornton et al., 2017) by selecting only the variables which are relatively important to the target variable values. This approach has been implemented when establishing the important training load indicators to predict injury status (Thornton et al., 2017) and in establishing the importance of seven sleep components to the Pittsburgh Sleep Quality Index score (Halsen et al., 2022). However, using machine learning variable importance methods is reported to be suboptimal in identifying key predictors to the target variable values (Williamson et al., 2021) and it affects classification accuracy.

For example, Whitehead et al. (2021) identified key predictor variables by using a single random forest model to establish variable importance of technical-tactical and physical performance indicators to classify rugby league players into two competitive levels (i.e., senior and academy) based on their playing positions (i.e., backs and forwards) using another Random Forest classification model. Whitehead et al. (2021) reported 83% accuracy for backs and 68% for forwards. Predictor variables that are not model-agnostic can be identified and the accuracy of classification models can be improved using the feature selection technique. A more reliable and robust method to Whitehead et al. (2021) method is aggregating repeated random forest variable importance results (Calhoun et al., 2021) which will involve using a different number of variables per attempt. However, this method is computationally expensive, may still produce a suboptimal classification model and the identified key predictor variables are still model-based. Alternatively, key predictor variables can be identified by applying a feature selection method (Mabayoje et al., 2016) that possesses no bias to a specific classification algorithm.

All feature selection methods exist as (Balogun et al., 2020; Chen et al., 2020a) Filter feature ranking, Filter feature subset selection, Wrapper-based and Embedded methods. Filter feature ranking methods generate a ranked score for every variable based on statistical properties found in the data as computed by the method. The filter feature ranking method requires a user to set a threshold or the desired number of variables before identifying key predictor variables. This is a limitation when the importance or weight of variables is unknown or when the total number of important variables is unknown. Filter feature subset-selection implements heuristic and search

methods to evaluate multiple subsets of variables to produce the best subset of key predictor variables as related to target variable values (Balogun et al., 2020; Chen et al., 2020a). Filter feature subset-selection methods resolved the limitations of the filter feature ranking methods. Importantly, the sets of variables produced by filter feature ranking and filter feature subset-selection methods do not have a bias towards any classification method.

On the other hand, wrapper-based methods (Balogun et al., 2020; Chen et al., 2020a) are based on a computational greedy search method of the variable space for finding variables that improve the predictive performance of a particular classification algorithm. Similarly, embedded methods (Balogun et al., 2020; Chen et al., 2020a) are intrinsic to machine learning algorithms that find the best features for split decisions while fitting a predictive model. Both wrapper-based and embedded feature selection methods are for improving the performance of a specific machine learning algorithm, as partially implemented by Whitehead et al. (Whitehead et al., 2021) to optimize the Random Forest model for rugby league players' competition-level prediction.

Adeyemo et al. (2022) utilized a filter feature subset-selection method to improve the accuracies reported in Whitehead et al. (2021) study of classifying rugby league players to competition levels within positional groups. The Correlation-based feature subset technique was considered because it outputs the best subset of key predictor variables without bias to any classification algorithm. Adeyemo et al. (2022) reported an improved 84.55% accuracy for backs and 77.42% accuracy for forwards in contrast with Whitehead et al. (2021) reported 83% accuracy for backs and 68% for forwards. Adeyemo et al. (2022) thus established that the filter feature subset-selection method (as a feature selection technique) will identify key predictor variables that are not based on the classification model, without the user specifying the number of key predictor variables to identify as well as capable of improving classification models' accuracy.

2.3 Frequent Pattern Mining in Sports

Pattern mining (Nijssen, 2013) is a type of data mining setting aimed at finding frequently recurring structures in data. Through pattern mining, interesting, usable and unexpected substructures (Fournier-Viger et al., 2017) are discovered within the data. Across the years, pattern mining as a setting of data mining has been extended to sat-

isfy other types of requirements and thus led to the development of various categories of pattern mining algorithms that are applicable in different use cases.

In non-sporting contexts, a graph-based pattern mining algorithm was successfully applied to mine cloned codes in a software system data (Qu et al., 2014) where it outperformed a traditional token-based method in terms of reduction in computational complexity and quick execution runtime. Rashid et al. (2012) applied a sequential pattern mining algorithm to efficiently analyse genome sequence data where interesting contiguous patterns were discovered. Further, a pattern mining algorithm (Pascu, 2018) was utilized to profile the debt default behaviour of bank customers to prevent future risk, where discovered patterns assisted in preventing future risks. In sports, technological advancements have assisted in data collection and management (Goes et al., 2021) and enabled the efficient and consistent collection and storage of sport-related big data. The collected data made historical and large volumes of data available for different advanced analyses in sports. One such advanced analysis is the application of pattern mining algorithms to analyse sport-related (non-)sequential external load for various purposes. Examples of the relevant applications of frequent pattern mining application in sports are discussed in the subsections 2.3.1 and 2.3.2 below.

2.3.1 Athlete Monitoring

Sports trainers are tasked with analyzing athletes' performance during training, which is a complex process. This is because athletes complete different forms of training (e.g., field vs. resistance training) across varying schedules and complete different amounts of training at different intensities. This can lead to varied stresses placed on athletes, influencing the extent that they either adapt and improve or heighten their risk of fatigue and injury. Managing the training process by collecting wearable data is an important method to inform decision-making. Although, based on a large amount of data obtainable through wearable devices, trainers are challenged with manually analyzing the data to monitor athletes' progression or form and thus turned towards applying advanced data analytics techniques.

Hrovat et al. (2015) applied a pattern mining algorithm to analyse an athlete's performance during sports training as a form of player progress monitoring. The analysis focused on providing a means to either identify the current form of a cyclist or serve as

2.3 Frequent Pattern Mining in Sports

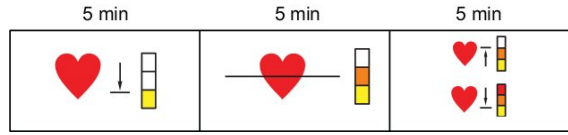


Figure 2.5: An example of significantly increasing duration pattern (Hrovat et al., 2015)

the cyclist's progress monitor through the discovery of interesting heart-rate sequential patterns from discretized time series data. Training data (gathered in TCX file format) was collected from a tracker device (i.e., Garmin Forerunner watch) worn by a cyclist each day. The tracking data was transformed into sequential data by splitting each daily data into some time intervals (e.g., 5 and 10 minutes), after which cycling speed, altitude and heart-rate features were identified and discretized into three levels (i.e., low, medium and high levels) based on equal thresholds. The Sequential Pattern Discovery using Equivalence classes (SPADE) pattern mining algorithm (Zaki, 2001) was used to identify and extract frequent sequential patterns. Afterwards, the interestingness (e.g., “more significantly increasing duration trends” and “more significantly decreasing duration trends”) of extracted frequent sequential patterns was calculated and visualized. Sequential patterns of daily heart rates that were difficult to achieve in practice, as well as those that were difficult to keep at the same level over time periods, were extracted by the SPADE algorithm. These patterns are characterized by a high level of maximal speed, a medium level of average heart rate and a maximal heart rate. An example of the identified top three “more significantly increasing duration trends” pattern over a fifteen minutes (i.e. 3 x 5-minute) time epoch cyclist ride is depicted in Figure 2.5), indicating low-level minimum heart rate for five minutes followed by medium-level average heart rate, and two medium-level maximum and minimum heart rates. Although, data from only one cyclist was analysed and the results cannot be generalised to all (professional) cyclists, Hrovat et al. (2015) established the use of a sequential pattern mining algorithm to find patterns that can assess the performance of an athlete and equally monitor an athlete's form. This demonstrates that movement patterns of rugby league players can be identified to assess their performance variability as it will be a useful solution for player monitoring.

2.3.2 Players Movement Profiling

Players participate in training programmes and competitive matches throughout competition seasons. Hierarchically, training programmes are broken into training weeks, sessions and modes (Weaving et al., 2022) while being carefully planned by a multidisciplinary team in sports for the concurrent development of players' multiple (technical, tactical and physical) qualities to ultimately improve players' performance during competitive match games. The aim of training programmes (Impellizzeri et al., 2005) is to individualise and manipulate the intensity, frequency and duration of training hierarchy to optimise the biomechanical and psycho-physiological responses, which assist in maximising the training-induced adaptation of players. Small-sided game (SSG) training (Pizarro et al., 2019), especially its constrained-led approach, is an important training method for the simultaneous development of players' physical and technical-tactical skills in team sports. Small-sided games aim to replicate the characteristics of competitive matches by enabling considerable unpredictability and decision-making demands on players during training.

Through its constrained-led approach (Machado et al., 2019), SSG training provides (Renshaw and Chow, 2019) an ecologically valid learning environment for players by reproducing situations that happened within competitive matches and offering a high degree of similarity between practice and competition. However, small-sided games as well as other training aspects are designed based on physical, technical-tactical and anthropometric indicators used to characterise competitive matches. These indicators (as predictor variables) are currently based on aggregated physical, technical and tactical demands. More so, there is no understanding of how players accumulated external load, no knowledge of players' frequently occurring group of locomotive activities nor what series of consecutive technical-tactical activities led to a desired technical or tactical event. Making inferences from visible locomotive activities to provide this knowledge or explain the accumulation of external load and or estimate short, high-intensity movement will be prone to error. As such, exploring the means to accurately provide this information led to the science of player movement profiling.

Player movement profiling is an emerging and interesting research area because it offers an alternative view to characterize team sport competitive matches. It is aimed at identifying team sport players' behavioural movement patterns during matches (Bunker

et al., 2021) (as certain ordered events have a strong influence on outcomes) to advance training programmes by providing information that imitates players' exact activities during matches. In the context of physical activities, it helps to identify frequent groups of movement activities (i.e., movement patterns) performed by players and uncover how often those groups of movements were performed. In the context of technical-tactical activities, player movement profiling helps to discover groups of events that led to a specific technical-tactical event. This can be extremely useful for players' strength and condition, players' performance monitoring and assessment, injury detection and or prediction, and players' positional group profiling for talent identification and recruitment, among many others.

At the commencement of this project, only two studies (Sweeting et al., 2014, 2017) were published that focused on the movement profiling of team sports players. Sweeting et al. (2014) embarked on the quantification of the physical movement (i.e., player movement profiling) of six female junior-elite netball athletes during the first quarter of match-play (i.e., 15 minutes duration) to understand performance and provide information for “next fixture” conditioning and any other specific preparation. The study was the first attempt to develop a movement sequencing technique that provides temporal sequences of movement characteristics and discrete movement sequences in sports. The study later applied a Longest Common Subsequence (LCS) sequential pattern mining algorithm to identify movement patterns from eighteen clustered discrete movement sequences over a 0.5-second epoch in comparison to three clusters of discrete movement sequences over a 1.5 seconds epoch. The longest common patterns extracted from the three clusters identified over the 1.5 seconds epoch are characterized by movement patterns involving sprinting, acceleration and deceleration in a straight direction. In later years, Sweeting et al. (2017) further developed the framework for player movement profiling. The study focused on the discovery of frequently occurring discrete movement sequences among twelve elite international-level female netball players, across all seven netball playing positions, that participated in four competitive national-level matches as a means to address the limitations of study (Sweeting et al., 2014). Data was collected via RF tracking devices and the longest common movement patterns were discovered using the existing player movement profiling framework. Frequently recurring movement sequences performed by players of each playing position were uncovered by conducting a frequency distribution. The fre-

quency distribution helped identified the movement signature of each netball playing position. Also, the study quantified the similarities among the movement signatures of each netball playing position by implementing Minkowski distance. A total of ten (10) frequently recurring movement sequences were identified across all netball playing positions and matches while three playing positions showed closely related movement signatures.

White et al. (2021) argued that the existing framework for player movement profiling (Sweeting et al., 2017) (Figure 3.2) is not stable while producing frequently recurring movement patterns because it produces different movement patterns for the same set of movement strings in consecutive runs. Thus, White et al. (2021) addressed this vital limitation by developing the Sequential Movement Pattern mining (SMP) framework and ran stability tests of both methods on the same set of rugby league elite players' movement strings. Both frameworks for player movement profiling presented by studies (Sweeting et al., 2017) and (White et al., 2021) were twice used to analyse identical datasets of thirteen individual rugby league players. The SMP framework produced stable movement patterns of the two frameworks for profiling players' movement patterns. Despite the robustness and stability of the SMP framework for discovering movement sequences, the total number of obtainable extracted patterns is limited to the number of clusters. Another limitation of both frameworks is that only the longest common pattern per cluster is outputted while other interesting patterns are discarded.

The main algorithm for finding the frequent sequential movement patterns in both frameworks (Sweeting et al., 2017; White et al., 2021) for player movement profiling was the "Longest Common Subsequence (LCS)" algorithm. LCS (Kuo and Cross, 1989) is a variant of sequential pattern mining algorithms that accepts two sequences at a time and outputs only the longest common subsequence of the pair. The outputted subsequence retained the order of item occurrence but does not account for item consecutiveness (i.e., missing movement activities within the movement pattern). Additionally, the LCS algorithm has no parameter that allows a user to specify a threshold of the least frequency movement patterns (to be identified) must satisfy. Ideally, the consecutiveness of movements or events in sports is important and there should be no omission of activities within profiled movement patterns because of replicating exact match characteristics as well as repeatability during strength and conditioning training

programmes. Therefore, the LCS algorithm is not too suitable for player movement profiling.

Overall, the extraction, interpretation and visualization of frequent sequential patterns were the core objectives of studies that applied pattern mining algorithms in a sporting context. Several existing pattern mining algorithms were useful and have been implemented to extract different types of frequent sequential patterns from ranges of sport-related data for various purposes. Pattern mining algorithms played a pivotal role in extracting patterns that detected tactics in football, identifying groups of patterns used by a cyclist to monitor player progress, used to discriminate between rugby union scoring and conceding outcomes as well as used for the extraction of movement patterns for player movement profiling. Especially for player movement profiling, the application of the LCS sequential pattern mining algorithm in this area leaves room for the development and application of a better pattern mining algorithm.

2.4 Sequential Pattern Mining Algorithms

In the search for a suitable sequential pattern mining algorithm to extract players' movement patterns, various sequential pattern mining algorithms were studied. There are two main categories of pattern mining algorithm, which is dependent on the nature of discovered patterns. The first category deals with mining *non-sequential* patterns from a large set of sequences to discover patterns. The algorithms under this category [Agrawal et al. \(1994\)](#) produce patterns that are without sequential order of items but maintain a *lexicographical* order of items. The second category deals with mining patterns where the order of items or events occurrences are considered when analysing a large set of sequences for frequently recurring patterns. This category of pattern mining involves the development of sequential pattern mining algorithms which was also pioneered by [Agrawal and Srikant \(1995\)](#). This PhD study focused on mining *sequential* patterns because it applies to identifying player movement patterns.

Before exploring the existing sequential pattern mining algorithms, basic definitions and concepts in sequential pattern mining are discussed below. Considering sequences of players' movements, sequences represent the ordered list of movement activity strings associated with a player. Each movement activity string consists of a set of movement units (i.e., encoded locomotive activities) performed by a player (per

2.4 Sequential Pattern Mining Algorithms

fixture). Each sequence is uniquely identified by a sequence identifier (sequence-id or SID), The size of the set of sequences ($|SDB|$) corresponds to the total number of sequences (i.e., the number of players (per fixture)) in the set.

Table 2.1: Example of a set of Discrete Movement Sequences (Adeyemo, 2023)

Sequence ID	Sequence
1	aabbcddefccdcgc
2	aabbcddefghcedh
3	aabbcddefcdcdf
4	aabbcddefceghceij
5	aabbcddefcdgc
6	aabbcddefcfdec

Table 2.1 is an example of a set of movement sequences with six sequences (i.e., $|SDB| = 6$): the first sequence contains fifteen movement units, the second contains fifteen movement units, the third sequence contains fourteen movement units, the fourth sequence contains sixteen movement units, the fifth sequence contains twelve movement units, and the sixth sequence contains thirteen movement units.

Formally, an item is a basic unit or the lowest granularity in data mining. An *itemset* is a set of characters or integers that forms up the sequences in a set of movement sequences. Let $I = \{i_1, i_2, i_3, \dots, i_k\}$ be a set of distinct items. A *sequence* $S = (j_1, j_2, j_3, \dots, j_m)$ is any series of ordered items in a set of movement sequences (SDB), where $j_1 \in I$ is an item for $1 \leq i \leq m$. Concisely, a sequence can be written as $j_1j_2j_3\dots j_m$. It is important to mention that an item j_1 can appear more than once in a sequence S . The size of the sequence S , denoted as $|S|$, is the number of distinct items in the sequence and the length of the same, denoted as $l(S)$, is the total number of items contained in the sequence regardless of the repeated item(s). In other words, the length of a sequence $l(S)$ is the count of all ordered itemset of any given sequence. Based on sequence s above, the length of the sequence i.e. $l(S) = m$. A set of movement sequences (SDB) is a list of sequences i.e. $SDB = \{S_1, S_2, \dots, S_n\}$ where each sequence has a unique identifier (i.e. ID).

Definition 2.4.1. Considering two sequences, $S_1 = (x_1x_2x_3 \dots x_m)$ and $S_2 = (y_1y_2y_3 \dots y_n)$, S_1 is contained or is a sub-sequence of S_2 (denoted as $S_1 \sqsubseteq S_2$ or $S_1 \sqsubset S_2$ if $S_1 \neq S_2$), if and only if there exist integers $1 \leq k_1 < k_2 < k_3 < \dots k_m \leq n$ and such that $x_1 = y_{k_1}, x_2 = y_{k_2}, x_3 = y_{k_3}, \dots, x_m = y_{k_m}$. S_1 can be referred to as a snippet

or sub-sequence of S_2 while S_2 can be referred to as a super-sequence of S_1 or said to contain S_1 .

Definition 2.4.2. A sub-sequence s^* is said to be contiguously contained in another (sub) sequence S , if and only if all the ordered items of s^* are exactly contained as the starting itemset of S , whose $l(S)$ must be higher between the two sequences.

Definition 2.4.3. Given a set of movement sequences G and a sub-sequence s^* , the **absolute support** of the sub-sequence s^* in a set of movement sequences G is the total count of sequences in a set of movement sequences G that contains s^* . This is denoted as $Sup_G^a(s^*) = |\{S | S \in G \wedge s^* \sqsubseteq S\}|$. Meanwhile, the percentage share of sequences in G that contains s^* is referred to as its **relative support**. This is also denoted as $Sup_G^r(s^*) = (|\{S | S \in G \wedge s^* \sqsubseteq S\}| / |G|) * 100$.

Definition 2.4.4. Given a set of movement sequences G and a sub-sequence s^* , the sub-sequence s^* is frequent if and only if its support value equals or is greater than the specified **threshold** = “ σ ”.

Definition 2.4.5. A contiguous sequential pattern s in a set of movement sequences G is referred to as a closed contiguous pattern if there exist no other contiguous sequential patterns s' such that s' are super-sequences of s and that the $Sup_G^a(s) == Sup_G^a(s')$.

Sequential pattern mining as a category of pattern mining is extremely important for analysing data collected and stored in sequential order. The importance of sequential pattern mining is premised on its many real-life applications because many data belonging to various disciplines are often encoded as sequences of alphanumeric such as in genome sequence analysis (Rashid et al., 2012), bioinformatics (Karim et al., 2012), sequence prediction and classification (Fradkin and Mörchen, 2015), software code bugs discovery (Qu et al., 2014), bank customer risk profiling (Pascu, 2018), Web-page click-stream analysis (Li, 2009), Web-logs access patterns (Jin and Lin, 2022) and large vehicle trajectory analysis (Bermingham and Lee, 2020). Also, time-series (Sim et al., 2009) (such as stock and athlete Global Positioning Systems [GPS]) data can be discretized, ordered based on timestamps and converted into sequences for the application of sequential pattern mining algorithms. The goal of all sequential pattern mining algorithms is to find and extract interesting subsequences from sets of sequences, where the sequential relationship between items is of special interest to the

user. The interestingness of patterns is often determined (Fournier-Viger et al., 2017) by frequency, length and or profit criteria.

Given the same interestingness parameter condition(s) and a set of sequences, sequential pattern mining algorithms will produce the same patterns as a result. Therefore, the difference(s) of sequential pattern mining algorithms does not lie in the outputted patterns but in the way each algorithm discovered the required sequential patterns. Sequential pattern mining algorithms differs from one another (Fournier-Viger et al., 2017; Mabroukeh and Ezeife, 2010) based on (1) how they represent the set of sequences for internal or external use, (2) how the search space of sequential patterns are explored to generate and store candidate patterns, (3) how each candidate pattern support is counted and tested to satisfy the frequency constraint, and (4) whether a breadth-first or depth-first search method was utilized.

There are two major categorisations and another three taxonomies of sequential pattern mining algorithms. The categorisation of sequential pattern mining algorithms according to (Fournier-Viger et al., 2017) depends on whether an algorithm is an (i) Depth-First search or (ii) Breadth-First search algorithm. The taxonomy of sequential pattern mining algorithms according to (Mabroukeh and Ezeife, 2010) was separated into three based on the technique for pruning the search space and candidate patterns generation, namely (i) Apriori-Based (AB) or Candidate Generation (ii) Pattern-Growth (PG) and (iii) Early-pruning. The early-pruning-based algorithms are offset of pattern-growth sequential pattern mining algorithms (Mabroukeh and Ezeife, 2010) and these algorithms often utilized the depth-first search method (Fournier-Viger et al., 2017). Meanwhile, most Apriori-based algorithms utilise a breadth-first search method. Hence, this thesis explored the taxonomy of sequential pattern mining algorithms as apriori-based and pattern-growth and a new snippet-growth category. All three categories are discussed in subsections 2.4.1, 2.4.2 and 2.4.4 (with popular examples) to uncover and understand the algorithmic characteristics of sequential pattern mining algorithms towards identification of player movement patterns.

2.4.1 Apriori-based Algorithms

The Apriori-based algorithms are based on the principles of *Apriori* for pruning a set of sequences search space. The *Apriori property* is stated as “All nonempty subse-

quences of a frequent itemset must also be frequent”. This *Apriori property* can be formally defined as, for any sequences j_a and j_b , if j_a is a subsequence of j_b ($j_a \sqsubset j_b$), then the support of j_b must be equal or lower to the support of j_a . For example, given two sequences $\{x\}$ and $\{x,y\}$, the occurrence frequency of $\{x,y\}$ will definitely be lower or at most equal to the occurrence frequency of $\{x\}$ because $\{x,y\}$ is more specific than $\{x\}$. The occurrence frequency is thus said to be *monotonic*. This property can also be described as *anti-monotonicity* or *downward-closed* because if a subsequence does not pass the minimum support test then all its super sequences will fail the test. Apriori-based algorithms utilize the downward-closure property to drastically reduce the search space of sequential patterns when it executes its Apriori-generate join procedure for candidate pattern generation.

All Apriori-based sequential pattern mining algorithms share three key characteristics, namely: generate and test, apriori-based pruning and multiple scans of the set of sequences. The Apriori-All Agrawal and Srikant (1995) and Generalized Sequential Patterns (GSP) algorithm (Srikant and Agrawal, 1996) algorithms both utilized the breadth-first search method to generate frequent sequential patterns starting with large 1-sequences. The candidate generation process of both algorithms is one of their limitations as both algorithms complete a lot of computationally expensive joins to generate large patterns. Besides, both algorithms are known to generate patterns that do not exist (Fournier-Viger et al., 2017) in the given set of sequences. This is possible because both algorithms generate longer candidate patterns by joining two smaller candidate patterns without checking the given set of sequences for its existence (e.g., the generation of large 3-sequences by joining two or more large 2-sequences). Apriori-All and GSP algorithms both waste a considerable amount of time by considering non-existing patterns in the given set of sequences. This will not be good behaviour for an algorithm for profiling player movement patterns in practice.

Most earlier Apriori-based algorithms are characterized by maintaining candidates in memory and performing multiple scans (Fournier-Viger et al., 2017) on the given set of sequences. Although the GSP algorithm improves its performance over Apriori-All by holding only frequent sequential patterns and the k-candidates in memory (Mabroukeh and Ezeife, 2010), both algorithm does consume a huge great amount of memory for maintaining candidate patterns. Similarly, both Apriori-All and GSP algorithms performed repeated scans of the set of sequences to compute the support value of candi-

date patterns to ascertain patterns frequency (Fournier-Viger et al., 2017). Meanwhile, it is easier for Apriori-All to count candidate patterns supports than GSP because of the maximal gap, minimal gap and taxonomy parameters introduced by GSP to formulated sequences (Zhao and Bhowmick, 2003). As the *frequency* of movement patterns is one of the reasons for identifying player movement patterns, the easiness to ascertain the frequency of movement patterns is an important feature of suitable sequential pattern mining which GSP lacks.

The characteristics (i.e. strength and weakness) of both algorithms led to the development of other efficient apriori-based sequential pattern mining algorithms such as Sequential Pattern Discovery using Equivalence classes (SPADE) (Zaki, 2001), Sequential Pattern Mining (SPAM) algorithm (Ayres et al., 2002) and Co-occurrence MAP based SPADE and SPAM algorithm (Fournier-Viger et al., 2014a). SPADE (Zaki, 2001) is characterized by its use of a vertical id-list format to represent a given set of sequences, unlike Apriori-All and GSP algorithms that used a set of sequences in its original form (i.e. horizontal). The vertical id-list is insensitive to data skewness and assists in calculating candidate pattern frequency without re-accessing the set of sequences (Mooney and Roddick, 2013). SPADE treats the original search space for candidate generation and pruning as a lattice and breaks it down into small pieces (sub-lattices) during its operation. It only requires three scans of a set of sequences to efficiently discover all frequent sequential patterns using both depth-first and breadth-first search for enumerating candidate patterns and both temporal and equality joins for generating large candidate patterns. SPADE performed better than the GSP algorithm by a factor of two while it is better by an order of magnitude if the support of 2-sequences is precomputed. However, it is limited by its use of vertical representation of a set of sequences for counting support (i.e., large-sized ID-list that cost time and space), generation of false candidate patterns as well as its large-numbered joins for generating patterns.

The SPAM algorithm (Ayres et al., 2002) is characterized by its use of a vertical bitmap representation of a given set of sequences, which enabled efficient counting of candidate patterns' support, to address the limitations of the SPADE algorithm. It also implemented a lexicographic tree or lattice for storing sequences items, this enables SPAM to specify the order in which each item can appear (e.g, $a \Rightarrow b \Rightarrow c, \dots, \Rightarrow z$). SPAM also implemented a novel depth-first strategy with effective pruning methods to

transverse the candidate pattern search space (Ayres et al., 2002; Mooney and Roddick, 2013). Although SPAM generates and tests candidate patterns, it was efficiently carried out via bitmaps ANDs (Mooney and Roddick, 2013). SPAM reportedly outperformed SPADE (Ayres et al., 2002) by about a factor of 2.5 on a small set of sequences while it outperformed both SPADE and Prefixspan (which is discussed in Section 2.4.2) by at least an order of magnitude on a large set of sequences. SPAM shares some similarities and limitations with SPADE, although they differ in their methods for representing data and generating candidate patterns. The limitations include false candidate pattern generation and large-sized ID-list among others. SPAM and SPADE algorithms' performance varied between space and time trade-off, as SPAM is faster but SPADE is space efficient (Mabroukeh and Ezeife, 2010). Both algorithms offer alternative methods for finding frequent sequential patterns and avoided the multiple scans of a set of sequences limitation of GSP and Apriori-All algorithms. Besides, Fournier-Viger et al. (2014a) proposed a candidate pruning mechanism to solve the generate and test property of algorithms by representing a set of sequences in vertical form. By introducing a new data structure called *Co-occurrence MAP*, a compact structure for storing item co-occurrence information from one single scan of the data, both SPADE and SPAM algorithms were further enhanced. Co-occurrence MAP is an optimisation method targeted to the candidate generation process. The optimised CM-SPADE and CM-SPAM algorithms greatly reduced generated candidate patterns in comparison to their basic forms and were up to eight times faster than SPADE and SPAM algorithms respectively. The limitation of the Co-occurrence MAP data structure for optimizing candidate pattern pruning is seen in a large amount of memory used in storing empty entries of item co-occurrence in a $n \times n$ matrix. Also, CM-SPADE and CM-SPAM algorithms generate false candidate patterns. Both SPADE and SPAM algorithms with the optimised CM-SPADE and CM-SPAM algorithms can only be suitable for identifying players' movement patterns if false candidate patterns are not generated, efficient in identifying frequent patterns, produced patterns without omission of items and based on a user-specified maximal length.

2.4.2 Pattern Growth Algorithms

Pattern growth algorithms were developed as a solution to the generate-and-test problem that plagued the apriori-based algorithms. The candidate generation step of the Apriori-based algorithm produces false candidate patterns that are not present in the set of sequences which also increases the computational cost (i.e., storing false patterns in memory and checking their support before exclusion). Pattern growth sequential pattern mining algorithm solution is based on recursive scans of a set of sequences to find larger patterns and as such consider only patterns that appear in the set of sequences. However, the original search space for finding (large) candidate patterns is partitioned into smaller sizes (i.e., projected set of sequences) to reduce computational cost while being transverse through the depth-first search method.

Examples of pattern growth algorithms are (1) Frequent pattern-project Sequential Pattern mining (FreeSpan) (Han et al., 2000) and (2) Prefix-project Sequential pattern mining (PrefixSpan) (Han et al., 2001) - the most influential pattern growth algorithm. FreeSpan (Han et al., 2000) is based on the property that claims *if an itemset X is infrequent, any sequence whose projected itemset is a superset of X cannot be a sequential pattern*. It considers a set of sequences as a set of tuples containing pairs of sequence identifiers and sequence, where a sequence comprises of series of elements which can only contain unique item(s). A sequential pattern is frequent if it is contained in at least n tuples in the set of sequences, where n is a positive integer defined by a user. The first scan of Freespan algorithm on the set of sequences is to find all frequent length-1 sequential patterns and sorts the items in support descending order into a frequent item list (i.e., f_list). Then, it implements a divide-and-conquer method to extract the complete set of sequential patterns of f_list into a frequent item matrix F . The frequent item matrix F is later used to generate the length-2 sequential patterns. It further scans the set of sequences to generate annotations on item-repeating patterns and a projected set of sequences which enable the further generation of 3 and longer frequent sequential patterns through alternative-level projection. Concisely, frequent sequential patterns are mined through original search space partitioning and recursive projection of sequence subsets of sequences based on the projected itemsets. When tested and evaluated on a set of sequences, Freespan outperformed GSP and naive FreeSpan algorithms, especially when the set of sequences grows large and the minimum support

2.4 Sequential Pattern Mining Algorithms

threshold reduces. The algorithm scales linearly as it examines fewer combinations of subsequences to identify frequent sequential patterns. However, it incurred memory and computation costs to hold the recursive projected set of sequences during analysis.

PrefixSpan (Han et al., 2001) was developed as a more effective and efficient pattern-growth sequential pattern mining algorithm. It was inspired by the Frequent Pattern Growth (FPGrowth) itemset mining algorithm (Han et al., 2004). It has one parameter that accepts the minimum support threshold besides requiring as input a set of sequences of sequences. PrefixSpan mainly utilises the set of sequences projection method to partition a set of sequences into smaller parts and does not generate candidate patterns based on joining two patterns, rather it recursively projects the set of sequences based on prefixes. Similarly to the FreeSpan algorithm, PrefixSpan utilized a depth-first search method to transverse the search space of sequential patterns. Beginning from length-1 frequent sequential patterns, it explores large patterns through a recursive appending of items to patterns to create larger ones. A lexicographical (or any other user-specified) order is put in place to ensure that there are no duplicates within the generated larger patterns. Prefix examines only the prefix subsequences and projects only their corresponding postfix subsequences into a projected set of sequences. Thus, it grows sequential patterns only within the projected set of sequences (and not the original search space) by exploring local frequent sequences. The recursive process terminates when the projected set of sequences contains no sequences or infrequent length-k sequential patterns. PrefixSpan was linearly scalable when evaluated. Its performance was compared against GSP and Freespan algorithms on synthetic sets of sequences and it was reported to be more efficient and scalable than both algorithms. It was however limited by the cost and creation of a projected set of sequences, although both bi-level projection and pseudo-projection were implemented to reduce the cost.

Pattern growth sequential pattern mining algorithms are characterised by predominantly using the depth-first search method and also implementing projection to partition the original candidate patterns search space into smaller units. This enables the reduction of time and memory consumption in identifying frequent sequential patterns. The major benefit of the pattern-growth projected set of sequences is that candidate generation through joining two lesser length patterns to form longer length patterns and subsequent testing for candidate patterns supports are no longer required (Mabroukeh

and Ezeife, 2010). Although the projection set of sequences can be computationally expensive, there are some existing optimisations such as bi-level and pseudo-projections.

As the number of patterns contained within the search space and the cost of operating for generating, processing and testing each itemset determine the time complexity of all sequential pattern mining, pattern growth algorithms have the advantage over Apriori-based algorithms because they only consider the patterns that are present in the given set of sequences. However, the application of pattern growth algorithm in practice revealed some algorithms such as SPAM and CM-Spade outperformed PrefixSpan (Ayres et al., 2002; Fournier-Viger et al., 2014a, 2017). This indicates the cost of scanning a set of sequences and projections (especially on a large set of sequences) can lower the performance of pattern-growth algorithms. In practice, pattern-growth sequential pattern mining algorithms offer more suitable algorithms for identifying player movement patterns because they do not consider nor generate false candidate patterns. However, none of the pattern-growth algorithms produced patterns without omission of items as well as based on a user-specified maximal length which are criteria of suitable movement patterns to profile rugby league players in practice.

2.4.3 Constraints-Based Algorithms

Both apriori-based and pattern growth sequential pattern mining algorithms aimed to identify frequent sequential patterns from a given set of sequences. These sequential patterns are often identified based on their frequency (i.e., the satisfaction of minimum support threshold) and generally with the omission of items within the identified frequent patterns. In practice, the task of sequential pattern mining differs for each application. Hence, multiple extensions or variations of the problems of sequential pattern mining are addressed. We refer to these problems as **constraints** and examples of constrained-based solutions include closed sequential patterns (Wang and Han, 2004), maximal sequential patterns (Fournier-Viger et al., 2013b, 2014b), top-k sequential patterns (Fournier-Viger et al., 2013a), compressed sequential patterns (Lam et al., 2014), contiguous pattern mining (Farzana Zerín and Jeong, 2011) and closed contiguous sequential patterns (Bermingham and Lee, 2020; Zhang et al., 2015).

The prominent limitation of traditional sequential pattern mining algorithms is that large numbers of frequent patterns are identified by algorithms, especially when the

minimum support threshold is set to a low positive integer value. Likewise, the characteristic (i.e., the total number of sequences and length of sequences) of a set of sequences also plays an important part in the number of extracted frequent patterns. The identification of a large number of patterns (based on frequent criteria) became a problem because users do not have enormous time to analyse a large amount of extracted frequent patterns. Therefore, extraction of *concise representations of frequent sequential patterns* instead of the all sequential pattern became the desired solution. A concise representation of frequent patterns is a subset of patterns that meaningfully and adequately summarise the whole set of all frequent patterns. These concise representations of frequent patterns are reportedly better and provided higher classification accuracy in practice (Gao et al., 2008; Pham et al., 2012), in comparison to using all frequent sequential patterns. Overall, three different forms of concise representation of frequent patterns exist, namely: maximal, generator and closed sequential patterns.

1. **Maximal sequential patterns** are set of frequent sequential patterns that have no sub-sequence(s) and supersequence(s). A sequential pattern s_a can be described as *maximal* if there is no other pattern s_b , where s_b is a superpattern of s_a (i.e., $s_a \sqsubseteq s_b$). Several studies (Fournier-Viger et al., 2013b, 2014b; Lu and Li, 2004) had developed algorithms for extracting maximal sequential patterns, some of the algorithms are based on pattern-growth (Fournier-Viger et al., 2013b), breadth-first search (Lu and Li, 2004) and even vertical representation of a set of sequences (Fournier-Viger et al., 2014b). Generally, the limitation of maximal sequential patterns is that despite being able to reproduce all frequent sequential patterns, it requires an additional scan(s) of a given set of sequences to recover the support value of extracted frequent patterns.
2. **Generator sequential patterns** on the other hand, is a set of frequent sequential patterns with no subsequence(s) sharing equal support. A sequential pattern s_a can be described as *generator* if there is no other pattern s_b , where s_b is a sub-pattern of s_a (i.e., $s_b \sqsubseteq s_a$) and both s_a and s_b shares the same support. Various algorithms (Gao et al., 2008; Lo et al., 2008; Yi et al., 2011) for extracting generator sequential patterns were developed, such as Frequent sequence generator miner (FEAT) (Gao et al., 2008), GenMiner (Lo et al., 2008) and Frequent Sequential Generators Patterns (FSGP) (Yi et al., 2011) which are based on pattern

growth and are reported to provide the smallest representation of subsequences that concisely represent a set of extracted frequent patterns.

3. **Closed sequential patterns** are the set of frequent sequential patterns not contained in other frequent patterns with the same support value. A sequential pattern s_a can be described as *closed* if there is no other pattern s_b , where s_b is a super-pattern of s_a and both s_a and s_b share the same support (i.e., $s_a \sqsubseteq s_b$ and $\text{support of } s_a == \text{support of } s_b$). Various algorithms for extracting closed sequential patterns were developed (Fournier-Viger et al., 2014a; Gomariz et al., 2013; Wang et al., 2007; Yan et al., 2003), such as BI-Directional Extension (BIDE) (Wang et al., 2007) and Closed Sequential pattern mining (CloSpan) (Yan et al., 2003) which are also based on pattern growth and Closed Sequential Patterns algorithm (Clasp) (Gomariz et al., 2013) and Co-occurrence MAP-Closed Sequential Patterns algorithm (CM-Clasp) (Fournier-Viger et al., 2014a) based on a vertical representation of a set of sequences. Closed sequential patterns are quite interesting because all frequent sequential patterns and their corresponding support values are recoverable without accessing the set of sequences and can be orders of magnitude smaller than all frequent patterns (Zaki and Hsiao, 2002). In other words, closed sequential patterns are lossless compression or representation of frequent sequential patterns.

Generally, all forms of concise representations of frequent sequential patterns (i.e., maximal, generator and closed) are motivated from the other branch of pattern mining - itemsets and association rules mining. These algorithms, such as AprioriClose (Pasquier et al., 1999), Closed Associate rule Mining (CHARM) (Zaki and Hsiao, 2002), Linear time Closed itemset Miner (LCM) (Uno et al., 2004) and Direct Count and Intersect Closed algorithm (DCI_CLOSED) (Lucchese et al., 2005), discovered closed itemsets and association rules from a set of transactions. For example, the AprioriClose algorithm (Pasquier et al., 1999) used a Galois connection closure mechanism to define a closed itemset lattice that identified all closed itemsets using to derive all association rules within a set of transactions without re-scanning the set of sequences. However, the patterns extracted by AprioriClose (i.e., association rules) contain items with omission but without repetitions while the items maintain a non-sequential but lexicographical order.

Other types of constraints besides concise representation of frequent patterns exist as *gap*, *duration*, *length*, *top-k*, *longest common subsequence* and *negative sequential patterns* constraints among others (Fournier-Viger et al., 2017; Mabroukeh and Ezeife, 2010). The *gap* constraint is interesting because most sequential pattern mining algorithms omit items within frequent sequential patterns. The *gap* constraint introduced specifying the minimum or maximum gap between two items within a frequent pattern. In some applications, such as genome sequencing (Karim et al., 2012), frequent contiguous patterns (i.e., patterns without omission of items - minimum *gap* = 0) are the interesting patterns to be identified. The omission of items among sequential patterns is not allowed.

The *longest common subsequence* constraint is originally based on the problem of finding the length of the longest subsequence common between two input strings. It evolved into identifying the actual longest common subsequence rather than the length. Longest Common Subsequence (LCS) algorithm (Benson et al., 2013) is the solution to this problem and is limited as it only accepts two input strings and outputs the longest common subsequence and its length. This limitation of accepting only two strings was resolved by the development of the multiple longest common sequence (MLCS) algorithm (Islam et al., 2019) that accepts two or more input strings. Nonetheless, the LCS algorithm and its variants only output the longest common subsequences and discard other common sequences which may be informative and or useful.

However, different constraint-based algorithms are not without limitations. Mining contiguous or top-k or longest common patterns from a large set of sequences result in a number of limitations such as high computational cost, noisy and redundant results, complex result and discard of long or interesting patterns. Overall, none of the existing constraints-based algorithms can extract movement patterns that are without omission of items, based on a user-defined maximal length, satisfy user-defined frequency threshold and provide a lossless compression of frequent patterns as required of a suitable algorithm for player movement profiling.

2.4.4 Snippet Growth Algorithm

At the commencement of this project, only one sequential pattern mining algorithm was based on snippet growth. Snippet-growth resolved the generation of false can-

candidate patterns that plagued apriori-based sequential pattern mining algorithms and expensive recursive scans of a set of sequences to find large patterns from a projected set of sequences that plagued pattern growth sequential pattern mining algorithms. Snippet-growth method split original sequences into a set of snippets through the n-gram model. This method assures the strict adjacency and ordering of items within sub-sequences (candidate patterns) as well as ensures the patterns are present in the set of sequences.

CCSpan (Zhang et al., 2015) is the only existing snippet-growth-based sequential pattern mining algorithm. It is also the state-of-the-art algorithm for finding frequent closed contiguous sequences from a given set of sequences. The extraction of frequent closed contiguous patterns from any set of sequences ensures that the produced result is a compact and compressed set of patterns wherein infrequent patterns are removed and some other frequent patterns (i.e., mostly frequent subpatterns) are also safely removed from the result without losing any information from the resulting frequent patterns. This avails the opportunity to re-generate frequent contiguous patterns with their support values from the resulting frequent closed contiguous patterns (because the subpatterns of each frequent closed contiguous pattern have the same support with its frequent closed contiguous patterns) without re-accessing the set of sequences. This is referred to as the *lossless compression* of frequent sequential patterns unlike the lossy compression performed by the maximal sequential pattern mining algorithms.

CCSpan is faster and consumes less computer memory in its provision of frequent closed contiguous patterns (Bermingham and Lee, 2020; Zhang et al., 2015) than four other benchmark and widely used algorithms for finding frequent closed (sequential) patterns namely BIDE (Wang and Han, 2004), CM-Clasp (Fournier-Viger et al., 2014a), ClaSP (Gomariz et al., 2013), and CloSpan (Yan et al., 2003). The major limitation of the CCSpan algorithm is its inability to scale well on a large set of sequences and especially those with very long sequences (Abboud et al., 2017; Bermingham and Lee, 2020). CCSpan algorithm uses a triple data structure to hold candidate patterns among other values and uses a conditional statement to check for duplicates. This data structure as well as a method for checking duplicates is another limitation as it increases execution runtime that inhibits optimization. Another limitation of the CCSpan algorithm is that its pre-post-sub-sequence pruning method for identifying closed contiguous patterns increases runtime and consumes memory by having to generate two

subpatterns from each candidate pattern and calculate the support values of both subpatterns besides comparing the support and checking if they all satisfy the frequency threshold criterion. CCSpan algorithmic step of candidate generation of one-length patterns and incrementally increases the step by 1 length until the maximum length required multiple scans of a set of sequences at every iteration. Although CCSpan can produce movement patterns with three (i.e., item contiguousness, user-defined frequency, and frequent pattern lossless compression) of four criteria for player movement patterns profiling, the remaining criterion (i.e., user-specified maximal length) was not satisfied. Therefore, a new sequential pattern mining algorithm that satisfies all the criteria of patterns for player movement profiling as well as resolves all limitations of CCSpan algorithm should be proposed and developed.

Overall, the limitations of sequential pattern mining algorithms (especially the state-of-the-art algorithm for finding closed contiguous patterns) present an opportunity to propose, develop and comparatively analyse a novel and better sequential pattern mining algorithm for finding (user-specified maximal length) closed contiguous sequential patterns. Additionally, the application of existing algorithms offers the means to validate the (movement) patterns produced by the proposed sequential pattern mining algorithm of this PhD study.

2.5 Chapter Summary

Rugby league football is a team sport, characterised by frequent bouts of high-intensity activities separated by lower-intensity activities where players require concurrent development of physical, technical and tactical skills in order to cope with match demands. Research aspects in rugby league utilised data containing anthropometric, physical, technical and or tactical performance indicators (Van Eetvelde et al., 2021; Whitehead et al., 2019) for various purposes (e.g., player performance monitoring and classification of players into playing level) towards developing players' skills. However, these existing performance indicators provide little or no understanding of how players accumulate external loads nor uncover a series of events leading to a desired technical-tactical event as they are often aggregated or reported in volume. Player movement profiling (Sweeting et al., 2017; White et al., 2021) uncovers match characteristics by identifying frequent groups of players' behavioural movement patterns

(i.e., movement and event activities) that sequentially happened during match games towards understanding the performance and replicating those match activities during players' training and development programmes. Existing player movement profiling methods have some limitations, particularly the algorithm that extracts movement patterns. The LCS algorithm for movement pattern extraction is not too suitable in practice because extracted movement patterns omit items (i.e. movement activities). It only identifies a single movement pattern and discards other interesting movement patterns among other limitations. More so, none of the existing sequential pattern mining algorithms [Benson et al. \(2013\)](#); [Zhang et al. \(2015\)](#) is suitable for player movement profiling. Hence, a suitable sequential pattern mining algorithm should be proposed and developed.

Additionally, the classification of rugby league players into playing positions is unexplored despite its usefulness while other classification problems (e.g., injury prediction and playing level classification) in rugby league (section 2.4.3) are thoroughly researched. In practice, the classification of players into positions is greatly useful in talent recruitment and (player) development because it identifies the skills and or qualities of players belonging to a specific positional group. Therefore, an attempt to classify rugby league players into playing positions (especially based on profiled movement patterns) should be made. More so, classification modelling should involve the use of multiple algorithms for comparative performance evaluation. Feature selection algorithm should be used for identifying significant or key predictor variables. Particularly, filter feature subset-selection methods among other categories of feature selection methods can be applied to identify key predictor variables without the need for a user to set a threshold or specify the number of important predictor variables to identify. Classification models should be developed and evaluated using a k-fold cross-validation technique and in the presence of a small number of instances, the leave-one-out validation technique can be utilised.

Overall, the quantification of rugby league players' external load through player movement profiling to identify movement patterns for separating players into playing positions should be explored. This will help identify signature movement patterns of players within playing positions which can transform how rugby league: talents can be identified; players' physical, technical and tactical skills are developed; and players' performance variability can be assessed and visualised.

Methodology

This PhD study is based on identifying movement patterns of elite rugby football league players, that can quantify players' external loads, even with respect to playing positions. Having studied and discussed various methods for player movement profiling as well as sequential pattern mining algorithms (applications in sports) in the previous chapter, this chapter discusses the methods used in this PhD study. An experimental framework of this PhD study is presented in Figure 3.1 and discussed in the sections below.

3.1 Player Movement Profiling Frameworks

Player movement profiling is based on using movement patterns to quantify players' external load towards discovering the order of match activities occurrences and how often they happened. Aspects of player movement profiling can be categorised into the generation of discrete movement sequences from Radio-Frequency (RF) or GPS data and the extraction of movement patterns. Therefore, the two existing frameworks for player movement profiling are critically discussed in the subsections below providing theoretical underpinnings on the proposed algorithm for extracting movement patterns.

3.1 Player Movement Profiling Frameworks

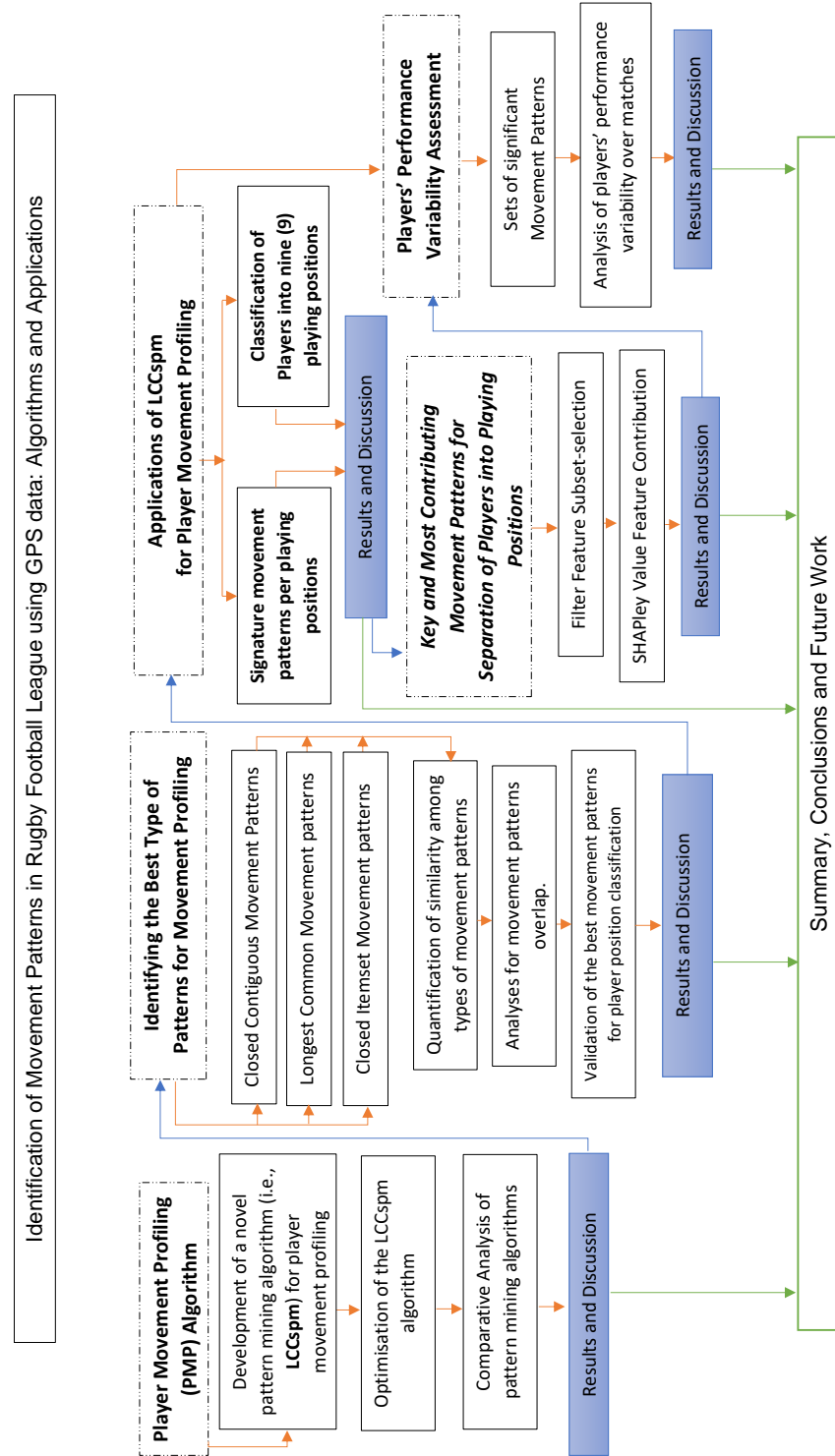


Figure 3.1: Outline of Research Method for this PhD study (Adeyemo, 2023).

3.1.1 Sweeting Framework

Following these two studies (Sweeting et al., 2014, 2017), Figure 3.2 depicts the movement sequencing technique developed and used in both studies. It is important to mention that the method was developed and used on RF data.

Players' RF tracking (positional i.e., X and Y coordinates) data were collected at the rate of 10Hz, the data was pre-processed by re-sampling into a 100-Hz data file using a customized software that implemented the Kalman filter method (Sathyan et al., 2012). Four features or variables were engineered and extracted from the resampled data (Sweeting et al., 2014, 2017).

Velocity as a variable was engineered from positional data, and its values were computed for each athlete. Velocity values for each player were computed using the formula illustrated in Equation 3.1

$$V_i = \frac{\sqrt{\Delta x^2 + \Delta y^2}}{\Delta t} \quad (3.1)$$

Acceleration as another variable was derived from velocity and its values were derived from the computed velocity values using the formula in Equation 3.2.

$$A_i = \frac{V_i - V_{i-1}}{\Delta t} \quad (3.2)$$

The angular displacement (Θ_i) was computed from the dot product of consecutive movement vectors, \mathbf{a} and \mathbf{b} as illustrated in Equation 3.3.

$$\Theta_i = \cos^{-1} \left[\frac{\mathbf{a} \cdot \mathbf{b}}{||\mathbf{a}|| \cdot ||\mathbf{b}||} \right] \quad (3.3)$$

From the angular displacement, the angular velocity was computed as described in Equation 3.4.

$$\omega_i = \frac{\Theta_i - \Theta_{i-1}}{\Delta t} \quad (3.4)$$

where t corresponded to a user-specified time epoch.

The velocity, acceleration and angular velocity values were clustered into four, three and four arbitrary clusters respectively using a one-dimensional k -means clustering algorithm. The declared clusters for each variable were then qualitatively labelled without regard to specific quantities: Velocity clusters were labelled walk, jog, sprint

3.1 Player Movement Profiling Frameworks

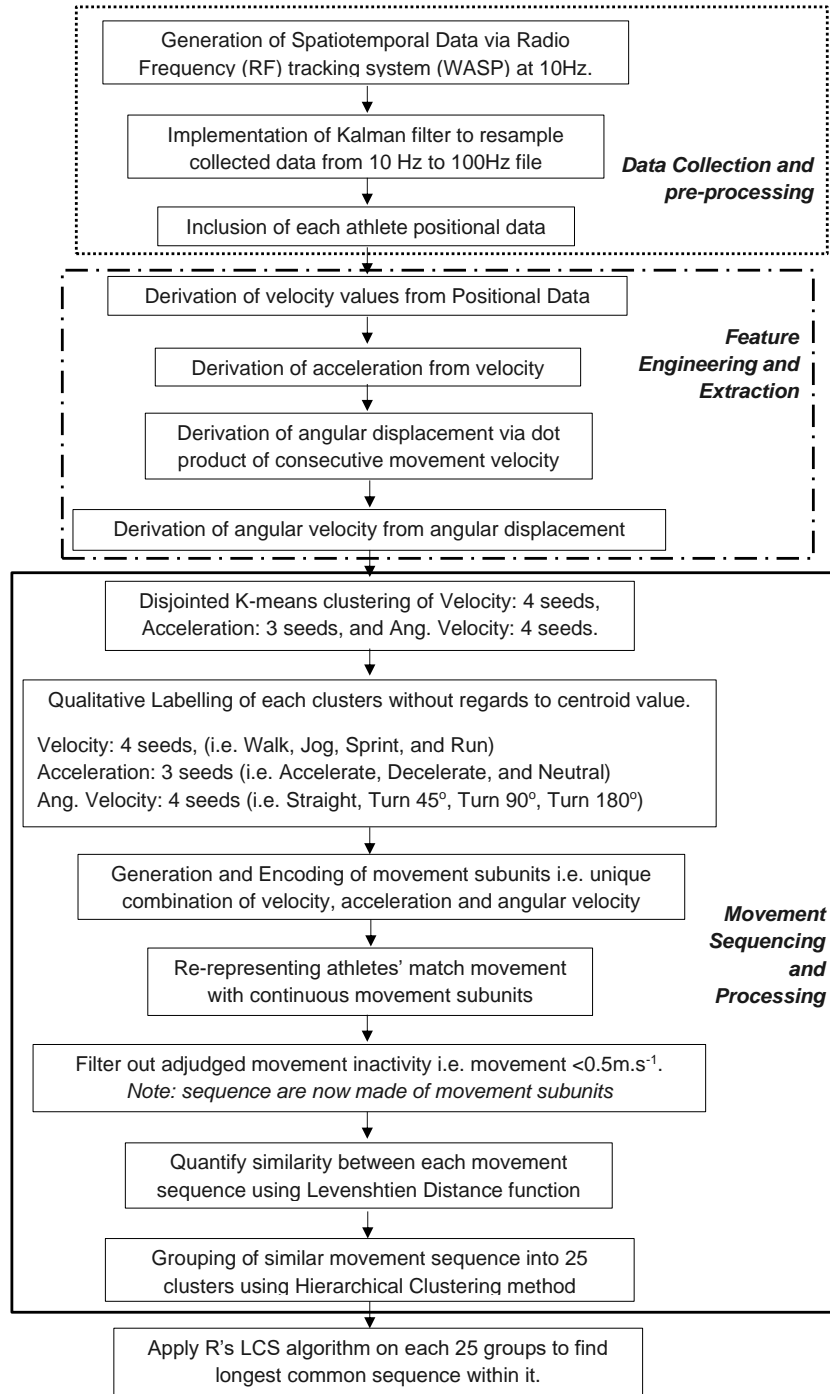


Figure 3.2: Depiction of Sweeting et al. Movement Sequencing Framework (Adeyemo, 2023).

3.1 Player Movement Profiling Frameworks

and run; acceleration clusters were labelled accelerate, decelerate and neutral; and angular velocity clusters were labelled straight, turn 45°, turn 90°, turn 180°. These qualitative labels were uniquely combined to derive forty-eight (i.e., 4 x 3 x 4) movement subunits, each assigned a unique alphanumeric letter ranging from “a-U” as identification codes.

Afterwards, each player’s period of match-play was represented by a temporal sequence of movement units. Discrete movement sequences were later isolated by delineating the temporal sequence of movement units using a threshold of 0.5 m.s⁻¹ that represents players’ moments of inactivity. Hierarchical clustering algorithm (Ward Jr, 1963) was implemented to cluster similar discrete movement sequences into twenty-five clusters. At the same time, Levenshtein distance (Levenshtein et al., 1966) was used to quantify the similarity of players’ movement sequences for each user-specified time epoch (i.e., 0.5, 0.75, 1.0, 1.25 and 1.50 seconds) respectively. The Longest Common Sequence algorithm (Kuo and Cross, 1989) was used to identify the Longest common movement patterns within each cluster. Frequently recurring movement patterns performed by individual playing positions were identified in two ways. The first is a compilation of the relative frequency of individual movement subunits per playing position. The second is the computation of the relative frequency of the LCS-derived movement patterns per playing position.

3.1.2 Sequential Movement Pattern-mining (SMP) Framework

White et al. (2021) suggested that the Sweeting framework (discussed in Section 3.1.1) for player movement profiling is not stable because it produces different movement patterns for the same set of movement strings in consecutive runs. This is possible due to the application of the K-means clustering algorithm to cluster velocity, acceleration and angular velocity data towards generating discretized locomotive activities as movement units. The instability might as well be caused by the arbitrarily qualitative labelling of each cluster. Hence, they developed the SMP framework that differs from the existing player movement profiling method mainly by its method for formulating temporal movement sequences and assigning movement sequences into hierarchical clusters.

The SMP framework defined the thresholds for discretizing GPS data (collected

3.1 Player Movement Profiling Frameworks

at the rate of 10Hz) into stable discrete movement sequences and ensured no cluster contained a single movement sequence. The SMP framework is a 4-step methodological framework (depicted in Figure 3.3) developed to process team sports GPS data to identify frequent sequential movement patterns ultimately. The first step formulates movement descriptors by extracting locomotive data (i.e. velocity and acceleration) and geospatial data (i.e., bearing) from GPS data sourced during training or competition, computes turning angle values from pairs of consecutive bearings and assigning the thresholds for discretizing those data into movement units. Table 3.1 contains the thresholds used in discretizing velocity, acceleration and turning angle values.

The second step formulates the movement units by applying the defined thresholds. A movement dictionary (i.e., the set of unique combinations of movement descriptors labelled by an alphanumeric letter) was developed to create a time-series sequence of continuous movement units (temporal movement sequence) over a period of match-played. See Table 3.2 for each movement unit and its unique descriptor character. Discrete movement sequences were isolated from the continuous movement units by removing players' movement below 1.20 m.s^{-1} because they are considered inactive periods. An example of movement sequence from Figure 3.4 is the sequential concatenation of the movement units i.e., "cbffehffffij".

The third step of the SMP framework identifies frequent movement patterns and can condense discrete sub-sequences of movement units if required. Hierarchical cluster analysis was implemented in this step to cluster similar discrete subsequences of movement units, whose similarity was quantified using the Levenshtein distance function. Cluster reassignment was carried out on clusters with one movement sequence by reassigning such movement sequences to the nearest cluster as the means to optimise the hierarchical clustering result through the prevention of single-element clusters. The longest common subsequences (LCS) algorithm was also implemented to discover the longest common movement pattern within each cluster.

The fourth and final step of the SMP framework identifies player-specific frequent SMP signatures, calculates the percentage of a frequent SMP within each player's movement pattern profile and used the Minkowski distance function to quantify players' signature movement.

3.1 Player Movement Profiling Frameworks

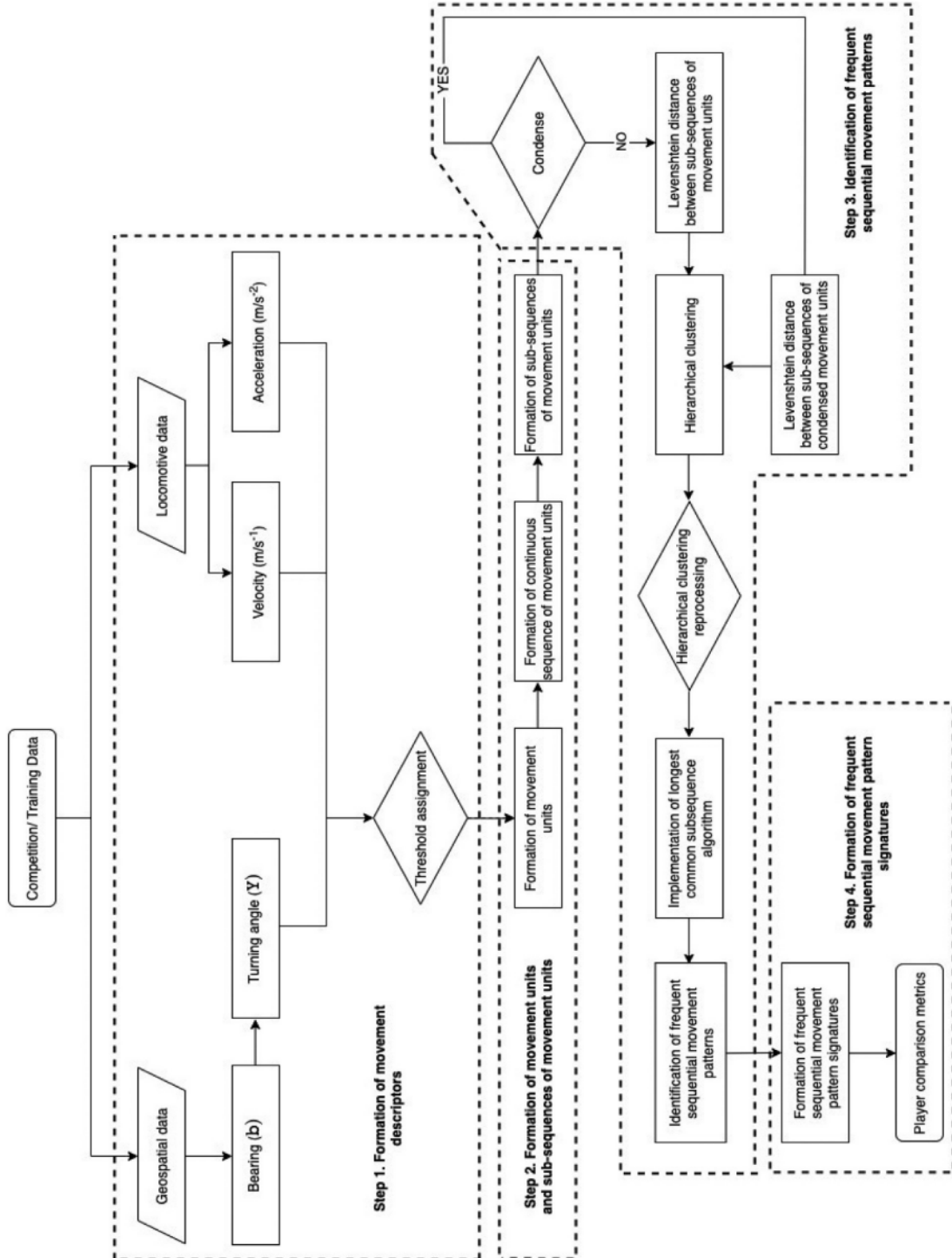


Figure 3.3: SMP Framework extracted from (White et al., 2021)

Table 3.1: The movement descriptors and threshold assignment values (White et al., 2021).

Velocity Descriptor	Velocity Threshold (m.s^{-1})	Acceleration Descriptor	Acceleration Threshold (m.s^{-2})	Turning Angle Descriptor	Turning Angle Threshold (Θ)
Walk	0.00 to <1.70	Deceleration	Min accel to ≤ -0.20	Straight	0.00 to <10.00
Jog	≥ 1.70 to ≤ 3.90	Neutral	> -0.20 to <0.20	Acute-change	≥ 10.00 to <45.00
Run	>3.90 to <5.00	Acceleration	≥ 0.20 to max accel	Large-change	≥ 45.00 to <90.00
Sprint	≥ 5.00	N/A	N/A	Backwards	≥ 90.00 to 180.00

ts	centiSeconds	Longitude	Latitude	Velocity	Acceleration	Seconds	Turning_Angle	Velocity_Descriptor	Acceleration_Descriptor	Turning_angles_Descriptor	Locomotive_event	Locomotive_event_characters
1517515079	19	-0.3681406	53.74663	0.035500001	-0.275839865	1378.6	53.53	Walk	Deceleration	Large-Change	WalkDecelerationLarge-Change	c
1517515079	29	-0.3681411	53.74663	0.0265	-0.229382828	1378.7	40.21	Walk	Deceleration	Acute-Change	WalkDecelerationAcute-Change	b
1517515079	39	-0.3681409	53.74663	0.020500001	-0.187039077	1378.8	19.19	Walk	Neutral	Acute-Change	WalkNeutralAcute-Change	f
1517515079	49	-0.3681411	53.74663	0.015500001	-0.152781263	1378.9	14.13	Walk	Neutral	Acute-Change	WalkNeutralAcute-Change	f
1517515079	59	-0.3681413	53.74663	0.011500001	-0.124585941	1379	0	Walk	Neutral	Straight	WalkNeutralStraight	e
1517515079	69	-0.3681414	53.74663	0.0085	-0.100941412	1379.1	102.94	Walk	Neutral	Backwards	WalkNeutralBackwards	h
1517515079	79	-0.3681413	53.74663	0.0065	-0.080707036	1379.2	35.17	Walk	Neutral	Acute-Change	WalkNeutralAcute-Change	f
1517515079	89	-0.368141	53.74663	0.0045	-0.065531254	1379.3	21.59	Walk	Neutral	Acute-Change	WalkNeutralAcute-Change	f
1517515079	99	-0.3681405	53.74663	0.0035	-0.051648442	1379.4	19.46	Walk	Neutral	Acute-Change	WalkNeutralAcute-Change	f
1517515080	9	-0.3681393	53.74663	0.0025	-0.041238282	1379.5	9.28	Walk	Neutral	Straight	WalkNeutralStraight	i
1517515080	19	-0.3681379	53.74663	0.152500004	0.344070315	1379.6	21.72	Walk	Acceleration	Acute-Change	WalkAccelerationAcute-Change	j
1517515080	29	-0.3681366	53.74663	0.260500014	0.528054714	1379.7			Acceleration			

Figure 3.4: Example of 10Hz GPS Data (Adeyemo, 2023).

3.2 Closed Contiguous Frequent Pattern Mining for Player Movement Analysis

Table 3.2: Movement Unit and Character (White et al., 2021)

Movement Unit	Character	Movement Unit	Character
WalkDecelerationStraight	a	RunDecelerationStraight	y
WalkDecelerationAcute-Change	b	RunDecelerationAcute-Change	z
WalkDecelerationLarge-Change	c	RunDecelerationLarge-Change	A
WalkDecelerationBackwards	d	RunDecelerationBackwards	B
WalkNeutralStraight	e	RunNeutralStraight	C
WalkNeutralAcute-Change	f	RunNeutralAcute-Change	D
WalkNeutralLarge-Change	g	RunNeutralLarge-Change	E
WalkNeutralBackwards	h	RunNeutralBackwards	F
WalkAccelerationStraight	i	RunAccelerationStraight	G
WalkAccelerationAcute-Change	j	RunAccelerationAcute-Change	H
WalkAccelerationLarge-Change	k	RunAccelerationLarge-Change	I
WalkAccelerationBackwards	l	RunAccelerationBackwards	J
JogDecelerationStraight	m	SprintDecelerationStraight	K
JogDecelerationAcute-Change	n	SprintDecelerationAcute-Change	L
JogDecelerationLarge-Change	o	SprintDecelerationLarge-Change	M
JogDecelerationBackwards	p	SprintDecelerationBackwards	N
JogNeutralStraight	q	SprintNeutralStraight	O
JogNeutralAcute-Change	r	SprintNeutralAcute-Change	P
JogNeutralLarge-Change	s	SprintNeutralLarge-Change	Q
JogNeutralBackwards	t	SprintNeutralBackwards	R
JogAccelerationStraight	u	SprintAccelerationStraight	S
JogAccelerationAcute-Change	v	SprintAccelerationAcute-Change	T
JogAccelerationLarge-Change	w	SprintAccelerationLarge-Change	U
JogAccelerationBackwards	x	SprintAccelerationBackwards	V

Overall, the LCS algorithm implemented for movement pattern extraction in both existing player movement profiling methods does not satisfy all four required criteria of movement patterns for player movement profiling. These existing frameworks have limitations (Section 2.3.2) including identifying few movement patterns from twenty-five clusters, producing only the longest common patterns while discarding other interesting patterns which may be useful in practice. Therefore, it becomes important to find a suitable pattern-mining algorithm for player movement profiling.

3.2 Closed Contiguous Frequent Pattern Mining for Player Movement Analysis

The LCS algorithm used by the existing player movement profiling frameworks is limited (discussed above), leading to the search for a suitable frequent pattern mining algorithm. There are various existing pattern mining algorithms but they lack pragmatic application for player movement profiling as reviewed in Section 2.4, besides the CCSpan algorithm which meets three of the four criteria of a suitable algorithm.

3.2.1 Closed Contiguous Sequential Pattern Mining (CCspan) Algorithm

CCSpan algorithm meets three of four criteria of a suitable algorithm for player movement profiling while having some limitations discussed in Section 2.4.4. CCSpan uses three compact data structures to execute the extraction of closed contiguous patterns. The first is a 2-tuple $(S.id, S)$ that represents the input set of sequences (S) and sequence identifier ($S.id$). The second data structure is a triple $(f, f.count, B)$ that represents a pattern (f), the pattern count ($f.count$) and an attribute B whose values are Y if the pattern is closed contiguous or N if otherwise. In the end, CCSpan outputs a set F consisting of all closed and non-closed contiguous patterns. CCSpan is known to generate a list of candidates by using the snippet-growth technique to ensure that candidate patterns do not include false patterns (i.e., patterns that do not exist in the set of sequences), saving memory consumption and lower execution runtime.

Algorithm 1 CCSpan Algorithm

Input: Set of sequences D and a minimum support threshold (σ)

Output: Complete set of closed contiguous sequential patterns

```
F ← ∅; // initialise F to store the closed contiguous patterns
Fk ← ∅; // initialize- Fk to store the length-k contiguous patterns
F1 ← init-gen( $D, \sigma$ ) // generate the frequent 1-sequences
1: for ( $k = 2$ ;  $F_{k-1} \neq \emptyset$ ;  $k++$ ) do
2:    $P_k \leftarrow \emptyset$  // initialize  $P_k$  to store the checked sequences
3:   for each sequence  $S \in D$  and  $l(S) \geq k$  do
4:     for ( $s \in S$  and  $l(s) = k$ ) do
5:       ConSP-gen( $D, s, F_{k-1}, P_k, S.id, \sigma$ ) // generate the contiguous patterns
6:     end for
7:   end for
8:    $F_k - 1 \leftarrow \text{CloConSP-gen}(F_{k-1}, F_k)$  // generate the closed contiguous patterns
9:    $F \leftarrow \cup_{k-1} F_k - 1$ ;
10: end for
11: return  $F \leftarrow \cup_k F_k$ ;
```

CCSpan snippet-growth method splits a set of sequences to generate patterns rather than enumerating all possible joins of frequent sub-sequences to produce a potentially longer pattern. CCSpan generates its list of candidates starting from a one-length pattern to the obtainable maximum length of patterns based on the given set of sequences.

3.2 Closed Contiguous Frequent Pattern Mining for Player Movement Analysis

It prunes its candidates for frequency and pattern closure at each incremental iteration. This state-of-the-art closed contiguous pattern mining algorithm implemented three pruning methods to remove non-frequent patterns and check for (non-)closed sequential sub-sequences. At initialization, CCSpan generates its one-length patterns without considering the last sequence in the set of sequences (because the support for any distinct pattern therein will be equal to “1”, which is not frequent) and prunes the one-length distinct patterns for frequency. This was employed to reduce memory usage and lower run time. During CCSpan incremental execution (i.e., starting from two-length patterns), it prunes generated snippets for duplicates by checking if it is already existing as a pattern within the created triple. Afterwards, it conducts a pre-post-sub-sequence pruning on the distinct patterns to identify infrequent patterns. Additionally, CCSpan uses this particular pruning method to check for transitivity property as a criterion for pattern closure. Lastly, it uses support pruning to compute the support of a snippet that is distinct and frequent based on a pre-post-sub-sequence check.

The application of CCSpan is explored for player movement profiling, providing a foundation for conceptual research into developing a novel algorithm for player movement profiling. The results of this are presented and discussed in Chapter 4.

3.2.2 Movement Pattern Analysis

This PhD study conducts empirical research to identify and thus validate the best type of movement patterns useful for player movement profiling. This experiment investigated if the consecutiveness of match activities matters when quantifying players’ external load. The sporting context of player positions was considered for this experimental investigation (See Section 2.2) because it is immensely useful for training customisation, talent identification and recruitment and players’ performance profiling among others.

There are three obtainable types of movement patterns (i.e., consecutive, non-consecutive and non-sequential) and all were considered to identify the best type of movement pattern for player movement profiling. Ultimately, three algorithms that can extract the three obtainable types of movement patterns from rugby league players’ GPS data were selected. Importantly, the best set of profiled movement patterns can be identified through the separation of players into two tactically different playing

3.2 Closed Contiguous Frequent Pattern Mining for Player Movement Analysis

positions, based on known differences, by applying five machine-learning classification algorithms for comparative analysis. The selected classification algorithms are discussed in the subsection below.

3.2.2.1 Classification Algorithms

Given that the classification of players into positional groups (based on movement patterns) is unexplored in rugby league (Section 2.2), various classification algorithms were succinctly studied to perform such an advanced analytics task. To identify and thus validate the best type of movement patterns for separating players into two distinct tactical playing positions as well as the separation of elite rugby players into nine playing positions, classification algorithms were used to induce knowledge from data with movement patterns as predictor variables and to measure the accuracy of separation provided by each type of movement patterns.

In this PhD study, five classification algorithms with different learning schemes were chosen and studied for various classification purposes. The classification algorithms are Decision tree, (Gaussian) Naive Bayes, Logistic Regression, Multilayered Perceptron, Random Forest, and k-Nearest neighbour. The multiple classification algorithms were selected to enable comparative performance evaluation of classification models for each classification task.

Decision Tree: Decision Tree algorithm is a non-parametric supervised learning technique (Pedregosa et al., 2011) used in this PhD study for classification. It learns simple decision rules inferred from data features as knowledge (in the form of an inverted tree) and presents a decision tree as a model. Decision Tree models are characterised (Bhargav et al., 2022) with nodes representing predictor variables or features, branches or links between nodes representing decisions, and leaves representing outputs or target variables values. The algorithm utilises a divide-and-conquer method to splits down training instances into smaller subsets of homogeneous nodes containing similar target variables values.

Random Forest: Random Forest is an ensemble of tree models (Pedregosa et al., 2011) (as depicted in Figure 3.5) that was used in this PhD study for classification. It

3.2 Closed Contiguous Frequent Pattern Mining for Player Movement Analysis

utilises a perturb-and-combine method to fit specified numbers of decision tree classification models via randomness of predictor variables and predict an instance's target variable based on the average prediction of individual classifiers. It learns from data by fitting low correlation decision trees (Bhargav et al., 2022) where each tree selects some random number of features and bootstrap sample instances with replacement. These features of the random forest model ensure better prediction or classification outcomes (compared to a single decision tree model).

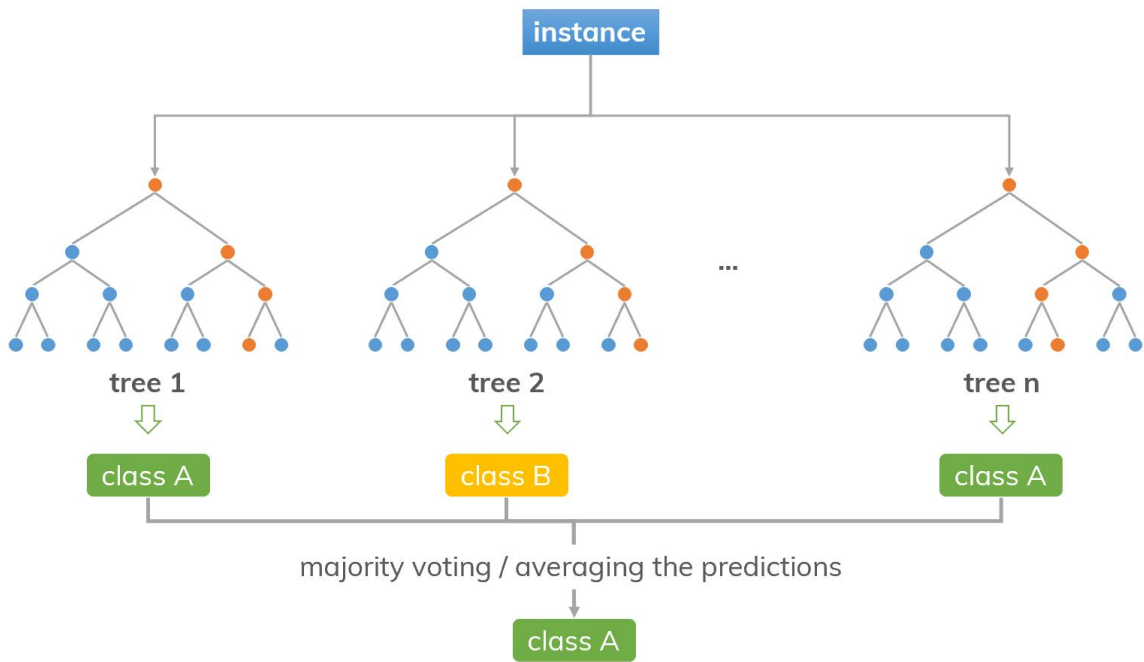


Figure 3.5: Random Forest Illustration RFI (2020)

Naïve Bayes: Naïve Bayes is a conditional probability-based classification machine learning algorithm. It uses the Bayes theorem, which strongly assumes that all predictor variables are independent of themselves (Adeyemo et al., 2020; Chen et al., 2020b), to fit predictive models. The Gaussian Naïve Bayes algorithm as implemented in *Sci-kit learn* python module (Pedregosa et al., 2011) assumed the likelihood of predictor variables to be Gaussian. It uses the formula in Equation 3.5 to compute the

3.2 Closed Contiguous Frequent Pattern Mining for Player Movement Analysis

conditional probability of an instance X_i belonging to a target variable value y .

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right) \quad (3.5)$$

where parameters μ_y and σ_y are estimated using maximum likelihood.

Logistic Regression: Logistic Regression is a statistical analysis technique used for predictive modelling (Balogun et al., 2019; Wilkens, 2021) such that independent or predictor variables are used to predict dependent or target variable values. The fitting of a Logistic Regression model is achieved through maximum likelihood (Pedregosa et al., 2011) where the optimal vectors and a constant is being determined. In the *Sci-kit learn* python module, the Logistic Regression model produces a numerical predicted probability (Equation 3.6) of the positive class $P(y_i = 1|X_i)$ which is divided by a default threshold of 0.5 for a binary case classification.

$$p(X_i) = \text{expit}(X_i w + w_0) = \frac{1}{1 + \exp(-X_i w - w_0)} \quad (3.6)$$

In the case of multinomial classification, the Logistic Regression model comprises a matrix of coefficients W containing rows of vectors W_k corresponding to a class k and produces numerical predicted probabilities $P(y_i = k|X_i)$ (as shown in Equation 3.7) where $k \in K$ and $y_i \in 1, \dots, K$ is the encoded target variable value for an instance i .

$$p_k(X_i) = \frac{\exp(X_i W_k + W_{0,k})}{\sum_{l=0}^{K-1} \exp(X_i W_l + W_{0,l})} \quad (3.7)$$

Multi-Layered Perceptron: Multi-Layered Perceptron (MLP) classification machine learning algorithm is based on Artificial Neural Network (ANN) (Mabayoje et al., 2016) and it represents a black-box learning method (Sharma et al., 2019). It is another supervised machine learning algorithm (Pedregosa et al., 2011) that learns a function $f(\cdot): S^j \rightarrow S^k$ on a training data, where j corresponds to the input dimensions and k corresponds to the output dimensions given a set of predictor variables $X = x_1, x_2, \dots, x_j$ and a target variable \mathbf{y} . It was implemented as layers of input, (multiple) hidden and output interconnected neurons (as depicted in Figure 3.6). The input layers consist of predictor variables as its set of neurons. The hidden layer(s) consist

3.2 Closed Contiguous Frequent Pattern Mining for Player Movement Analysis

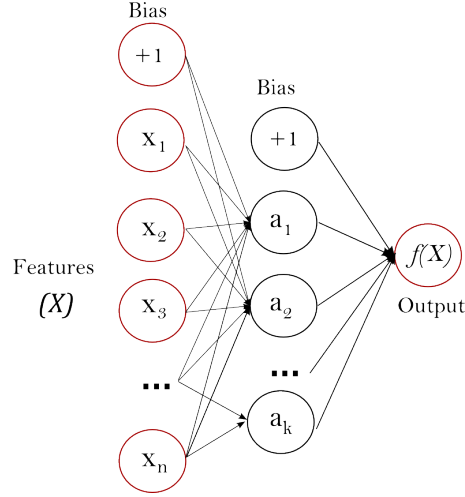


Figure 3.6: MLP with one hidden layer (Pedregosa et al., 2011)

of neurons with transformed values (i.e., weighted linear predictor variables) from the preceding layer followed by a non-linear activation function (e.g., tan function). The output layer consists of neurons which accept values from the last hidden layer and transform them into output values that correspond to the target variable values.

3.2.2.2 Pattern Similarity Measurements

Besides using the classification modelling (Section 3.2.2.1) to analyse movement patterns for identifying the best type of movement patterns, the sets of obtainable movement patterns can also be analysed for similarity and overlaps.

Jaccard Analysis: Jaccard similarity measure (Wang et al., 2019) enables exact matching of patterns between two sets and was used to quantify the similarity among the groups of extracted patterns. Given two sets of movement patterns (i.e., A and B), the Jaccard similarity index is computed as in Equation 5.1.

$$J(A, B) = |A \cap B| / |A \cup B| \quad (3.8)$$

Overlap Movement Patterns: Overlap movement patterns between two sets of movement patterns were identified using the exact matching method. Overlapping unique movement patterns between pairs of pattern mining algorithms were identified. For

each pair, the most frequent-50 and least frequent-50 extracted movement patterns of each pattern mining algorithm were checked for overlapping and visualised. This was carried out to identify where the overlapping movement patterns are located. A further analysis was carried out on the identified overlapped movement patterns to discover those patterns performed by players of each playing position. Also, the overlapped movement patterns between playing positions per pattern mining algorithm were explored.

Overall, considering the:

1. existing stable and robust method (i.e. SMP framework) for player movement profiling,
2. various types of obtainable movement patterns available for analysing sports big data, and
3. sequential nature of rugby league match activities,

movement patterns analysis (consisting of classification modelling and pattern similarity measurements) was considered to ascertain the type of movement patterns that best separate RFL players into playing positions. Therefore, pattern mining algorithms that extract the three types of obtainable (i.e., sequential but non-consecutive, sequential and consecutive, and non-sequential) movement patterns from data of two tactically different playing positions in rugby league were explored, towards validating the best patterns for player movement profiling. The two rugby league playing positions (i.e. hookers and wingers) were selected based on their known differences in tactical roles during matches and [Meir et al. \(2001\)](#) also revealed that hookers and wingers share nearly similar average body weight but perform distinct tactical roles and different on-field activities (e.g., 15-m and 40-m sprints). The results of these analyses are presented and discussed in [Chapter 5](#).

3.3 Player Movement Profiling

This PhD study also conducts applied research using the identified best type of movement patterns to complete three applications useful in practice. The first application

will identify signature movement patterns per rugby league playing position. The second application will investigate the classification of RFL players into all playing positions using movement patterns as opposed to traditional performance indicators. Also, appropriate movement pattern values as well as key and significant movement patterns for such classification analyses will be investigated. The third application will establish a set of “Movement Performance Indicators”, from extracted movement patterns, useful to assess and visualise the performance variability of players. Each application analysis is discussed in the subsections below.

3.3.1 Signature Movement Patterns of RFL Playing Positions

Following the identification of the algorithm that extracts the best type of movement patterns to quantify players’ external load, the same was used to quantify players’ external load to make discoveries about rugby league players’ positions and movement patterns. This is important because rugby league is a physically intense team sport where players are tasked with different responsibilities during competitive matches. The use of movement patterns, in this regard, will uncover the specific behavioural movement patterns of players belonging to each rugby league playing position. As there are nine playing positions in rugby league, discovering the signature movement patterns of players within each position will provide vast usefulness in practice such as customised training and talent scouting. The results of this analysis are presented and discussed in Section [6.2.1](#).

3.3.2 RFL Playing Position Classification using Movement Patterns

Following the identification of the best type of movement pattern for player movement profiling (as discussed in Section [3.2](#)), the same was considered for the classification of elite RFL players into all nine playing positions. Traditionally, physical and technical-tactical indicators that quantify players’ external load are used to develop classification models in sports literature despite their limitations. Given that movement patterns also quantify players’ external but with more granular information, empirical research was conducted to investigate if movement patterns can be used for the classification of players into playing positions. The machine learning algorithms (discussed in Section [3.2.2.1](#)) were also considered for the classification modelling analysis.

Further experiments were conducted to resolve issues faced during the classification modelling such as data imbalance problem, data high dimensionality problem, and the identification of movement patterns importance and contribution for classification modelling. The methods and or techniques used during classification modelling are discussed below.

3.3.2.1 Data Representation

Data for classification modelling is represented by observations or instances that serve as input data. These observations are often described by a set of finite predictor variables and one target variable with a set of nominal values. In sports analytics literature, various demographics, genetic markers, anthropometric measurements, and training intensity indicators (Ayala et al., 2019; Whiteside et al., 2016) are often used as predictor variables to describe observations of players for classification. In rugby league, physical performance indicators (Kempton et al., 2017) and the combination of both physical and technical-tactical performance indicators (Adeyemo et al., 2022; Whitehead et al., 2021) have also been utilized for classification modelling. These indicators (i.e., physical and technical-tactical) are collections of unexplained accumulations of players' external loads often aggregated and reported in volume. The classification modelling in this PhD study differs from others within the domain of sports analytics and rugby league research because profiled movement patterns of elite rugby league players were considered as predictor variables to describe the observation of players per match.

Movement patterns as predictor variables were considered because they describe players' match behaviour, captured the sequential nature of match-game characteristics and explain how players accumulate external load. In other words, rugby league players are represented by observations described by unique movement patterns and labelled by players' playing positions. More so, this PhD study is the first attempt to classify (rugby league) players into playing positions based on movement patterns.

3.3.2.2 Data Sampling

Considering the various classification tasks in this PhD study, different study designs were implemented leading to a varying number of observations of players representing

each playing position. When observation of player playing positions are under and or over-represented, (i.e., class imbalance problem), sampling techniques were utilized. Sampling techniques avail the balanced representation of observations of players of each playing position within a given input data for classification modelling.

Two sampling techniques were considered throughout this PhD study to resolve class imbalance when encountered. These sampling techniques (Lemaître et al., 2017) are Synthetic Minority Oversampling Technique (SMOTE) and Random Under-Sampler (RUS). SMOTE is an over-sampling technique that creates “synthetic” observations of minority class instead of executing an over-sampling with replacement. It creates a synthetic observation(s) by selecting a random observation from a selected minority class, with some neighbours and then creates as many synthetic observations at the point between the selected observation and one of its random neighbours. This method is effective for over-sampling minority class(es) because the created synthetic instances are closely related to the selected random instance. RUS, on the other hand, is simply the random removal of instance(s) from the selected majority class.

3.3.2.3 Classification Model Development and Evaluation

Developing or fitting a classification model can be carried out by splitting a given set of observations into two where the first and larger partition can be used for training and the second for testing the developed model. This method is referred to as train-test split and it is commonly used (Pantzalis and Tjortjis, 2020; Whitehead et al., 2021) in sports analytics. This method is somehow limited as it carries out model fitting and evaluation only once on the given data. Additionally, when the number of observations is limited, splitting them into train and test sets will limit the number of observations that a classification algorithm can learn from.

A k -fold cross-validation (Adeyemo et al., 2022) method provides an alternative that allows training and testing of classification models to be performed k -times after splitting a given data into k partitions and outputs the average scores of any selected performance evaluation metrics. As implemented in Sci-kit learn model (Pedregosa et al., 2011), k -fold cross-validation has three parameters namely: *n_splits*, *random_state* and *shuffle*. The *n_splits* parameter determines the number of dataset partitions created before conducting the training and testing. It also determines the

3.3 Player Movement Profiling

Table 3.3: Confusion Matrix (Adeyemo et al., 2022)

	Predicted condition		
	Total population = P + N	Positive (PP)	Negative (PN)
Actual condition	Positive (AP)	True positive (TP)	False negative (FN)
	Negative (AN)	False positive (FP)	True negative (TN)

iterative number of times the training and testing will be conducted and the number of models to be fitted and aggregated for results. The *random_state* parameter ensures the stability of the cross-validation technique for dataset partitioning as well as fitting of the models. This in turn ensures the experiment can be repeated multiple times with exact results. The *shuffle* parameter is concerned with target variable values representation in each data partition. It ensures that the instances of target variable values are not condensed together but rather scattered among themselves. This assists in making sure all data partitions contain all target variable values.

The evaluation of classification model performances on a test set of observations is often considered to be better than evaluating on a training set of instances because the observations in the test set were not included for knowledge inducement by the learning algorithms. Evaluating the performance of classification models on training observations will result in an optimistic and biased estimate of classification models. Meanwhile, performance evaluation of classification models on a test set of observations not used to fit a model will reflect a real-life application. The evaluation of classification models is important because it helps to measure the learning process of each model. Also, it helps to complete a comparative analysis of multiple classification models' performances.

Classification accuracy, precision and recall are among the primary performance evaluation metrics considered for evaluating classification models in various classification tasks while the F_1 -score or F-measure metric (Pedregosa et al., 2011) is a secondary performance evaluation metric. All primary performance evaluation metrics can be computed directly from the values within a confusion matrix (depicted in Table 3.3). The classification accuracy is simply the percentage of correctly classified instances divided by the total number of instances. Equation 3.9 computes the

classification accuracy of a classification model(s).

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \times 100 \quad (3.9)$$

Precision is the positive predictive value of the classification model, computed in Equation 3.10. It is the score of correctly classified positive observations with respect to those predicted as positive.

$$Precision = \frac{TP}{TP + FP} \quad (3.10)$$

Precision is the total number of true positive observations divided by all positive observations.

Recall is the sensitivity of the classification model, otherwise referred to as the true positive rate. It is the score of correctly classified positive observations with respect to all positive observations and computed as shown in Equation 3.11.

$$Recall = \frac{TP}{TP + FN} \quad (3.11)$$

Recall is the total number of true positive observations divided by the total number of observations that should have been identified as positives.

F-measure as the second performance evaluation metric is computed from the values of precision and recall as shown in Equation 3.12.

$$F_1 = 2 \frac{precision \times recall}{precision + recall} \quad (3.12)$$

It is simply the harmonic mean of both recall and precision.

3.3.2.4 Key Movement Patterns for Classification

As revealed through a review of the literature (Section 2.2.1), the application of feature selection methods is often based on identifying key predictor variables. Also, it is applied as a means to remove redundant or correlated predictor variables used to describe the set of observations for classification modelling.

In this PhD study, filter feature subset-selection feature selection techniques were

considered for the identification of key predictor variables because they output the best subset of features (without irrelevant and redundant features) through different mechanisms and without bias to any classification algorithms. There are only two filter feature subset-selection methods vis-a-vis (i) Correlation-based and (ii) Consistency-based Subset Evaluator techniques, which are discussed below.

Correlation-based Subset Evaluator: The Correlation-based feature subset is an example of a filter feature subset-selection method that outputs the subset of variables with the highest merit score according to the heuristic evaluation function (Ali et al., 2020; Hall, 1999). Fundamentally, this filter feature subset-selection works by evaluating subsets of features that must be uncorrelated among themselves but highly correlated with the dependent variable (i.e., target class). It uses a search method to find and select a random set of features and iteratively calculates its merit score. The subset of features with the highest merit score is outputted as the set of key predictor variables. The merit score is calculated as follows:

$$Merit_s = \frac{l\overline{tcf}}{\sqrt{l + l(l-1)\overline{tff}}} \quad (3.13)$$

where M_s holds the score after evaluating a subset of s consisting of l variables, \overline{tcf} is the average correlation values between subset variables and target variable values, and \overline{tff} is the average correlation value between subset variables (Hall, 1999).

Consistency-based Subset Evaluator: Las Vegas (LV) algorithm (Liu et al., 1996) is a probabilistic approach to finding minimum predictor variables that represent all features. It is often referred to as the consistency-based Subset Evaluator method, which is another type of feature selection technique that evaluates the level of consistency of a subset of features to the level of consistency of all features. It is able to handle noise in data when the approximate noise level is known a-priori. It selects a random subset of features from the original feature space, evaluates the *inconsistency* rate of the subset and compares it with the *inconsistency* rate of the best subset. If the new subset of the feature is at least consistent with the best subset, the new subset replaces the best.

It is also used with a search method to select subsets of features and output the subset with the consistency score closest to the full dataset consistency score. The *consistency* score is calculated (Binbusayyis and Vaiyapuri, 2019) as follows:

$$Consistency_s = 1 - \frac{\sum_{n=0}^J |D_i| - |M_i|}{N} \quad (3.14)$$

where s is the candidate feature subset, J is the number of distinct feature values in the subset, $|D_i|$ and $|M_i|$ is the number of occurrences and the cardinality of the majority class for the i^{th} feature value in the current subset.

However, both filter feature subset-selection techniques use a search method to randomly select feature subsets before computing their score. The best first search and the genetic search methods (Witten and Frank, 2002) were considered because both methods find and choose random features differently. The best first search method uses greedy hill-climbing enhanced with backtracking to search the feature space for finding and selecting predictor variables while the genetic search method performs a feature search using a genetic algorithm.

Feature Importance and Contribution: Feature importance (Polat et al., 2017; Witten and Frank, 2002) reveals the order of features (i.e. predictor variables) as important to each classification model to predict the class (i.e. target variable value) an observation belongs to. It helps to quantify and describe how important the feature was for the classification model performance. However, the feature importance of data for one model is not usually important for another model. Also, feature importances can either be modular global or local importance. Modular global feature importance measures the importance of the feature for an entire model while local importance measures the contribution of the feature for a specific observation.

Feature contribution or local importance helps in explaining how predictive model(s) utilized the predictor variables and how much influence each predictor variable had while the model made each classification decision. The ability to correctly interpret a prediction model's output is extremely important (Lundberg and Lee, 2017) because it assists in understanding the process being modelled and enables user trust and provides insight into how a model can be further enhanced. Lundberg and Lee (2017) presented SHapley Additive exPlanation (SHAP) as a unified framework for interpreting models

predictions against six existing methods such as Local Interpretable Model-agnostic Explanations (LIME), Deep Learning Important Feature (DeepLIFT) and Layer-Wise Relevance Propagation among others. SHAP values offer feature importance with local accuracy, missingness and consistency. Local accuracy offers a model's explanation for a specific observation predicted accuracy. It illustrates the contribution of each predictor variable towards getting the accuracy from the base value $E[f(z)]$ to the predicted accuracy $f(x)$. Missingness handles missing predictor variables and they have no impact on the computation of predictor variables' importance or contribution. Consistency ensures the input's attribution does not decrease even if a model changes so that some predictor variable increases or remains the same. Concisely, SHAP value is an additive predictor variable attribution method that unified six other methods for interpreting classification model(s) through feature importance and contribution.

The filter feature subset-selection feature selection and classification models' importance techniques are primarily applied to identify key movement patterns while SHAP values feature importance and contribution method was used to uncover the most contributing movement patterns used by the classification model to classify players into positional groups. Altogether, feature selection, importance and contribution methods are used to identify the significant movement patterns of players for classifying players into playing positions.

3.3.3 Movement Performance Indicators (MPIs)

Additional application of movement patterns was carried out through the use of the results of the identified key and significant extracted movement patterns across playing positions for player performance variability assessment. Players' performance variability is a longitudinal study that involves the identification of a set of movement patterns as "Movement Performance Indicators (MPIs)" useful to assess and visualise rugby league players' performance variability over a number of fixtures. Significant and key movement patterns were identified during the classification modelling of players into playing positions to establish MPIs. MPIs are identified by cross-matching key movement patterns identified through the filter feature subset-section technique (discussed in Section 3.3.2.4) and significant movement patterns identified through SHAP value feature contribution method (discussed in Section 3.3.2.4) across play-

ing positions. The “Movement Performance Indicators” are then used to assess the performance variability of players across playing positions.

Overall, the identification of the signature movement patterns of each RFL playing position was explored. An investigation of the use of movement patterns (and appropriate values) as predictor variables to classify RFL players into playing positions was considered. The consideration of the development and evaluation of multiple classification models to discover the best model was also explored. Also, the discovery of key movement patterns through two feature selection methods and significant movement patterns through feature importance and contribution techniques were made. The establishment of “Movement Performance Indicators” and its demonstration for performance variability assessment was explored. The results of this are presented and discussed in [Chapter 6](#).

3.4 Chapter Summary

This chapter presented and discussed a methodology used by this PhD study. Each subsequent chapter conducted the experiments described per section in this chapter, towards achieving the aim of this PhD study. More so, each subsequent chapter contains a method section that provides explicit information about its design as well as results and discussions.

LCCspm: l -Length Closed Contiguous sequential pattern mining algorithm

In chapter 3, the limitations of the existing framework used in sports for player movement profiling were critically analysed and the need for the development of a new algorithm for extracting player movement patterns was justified (Section). This chapter formulates, implements, optimised and evaluates a novel sequential pattern-mining algorithm. This chapter fulfils the second objective of this PhD study. The novel algorithm was presented at International Conference on Machine Learning and Application (ICMLA) and is published (Adeyemo et al., 2021) in an online proceeding by The Institute of Electrical and Electronics Engineers (IEEE).

4.1 l -Length Closed Contiguous Pattern Mining

A suitable sequential pattern mining algorithm for player movement profiling will ideally produce (movement) patterns that satisfy four criteria which are:

- user-specified maximal length,
- item contiguousness,
- user-defined frequency threshold, and
- frequent pattern lossless compression.

In a holistic effort to resolve both theoretical and practical limitations of existing algorithms for player movement profiling, this thesis explored the development of a sequential pattern mining algorithm for mining closed contiguous patterns whose interestingness is based on user-defined maximal length of patterns and user-specified support threshold while it fulfils other criteria (i.e. item contiguousness and frequent pattern lossless compression). We refer to this algorithm as l -length Closed Contiguous sequential pattern mining algorithm (i.e., *LCCspm*).

In this section, the problem of l -length closed contiguous sequential pattern problem is formulated and formalised. The proposed LCCspm algorithm approaches for candidate generation and search-space pruning as well as pattern-closure check are discussed and compared with existing algorithms. Also, the proposed LCCspm algorithm (using pseudo-code) and its complexity are analysed and discussed. However, basic definitions and concepts in sequential pattern mining are re-introduced.

4.1.1 Basic Concepts and Definitions

An item is a basic unit or the lowest granularity in pattern mining. An *itemset* is a set of characters or integers that forms up the sequences in a set of movement sequences. Let $I = \{i_1, i_2, i_3, \dots, i_k\}$ be a set of distinct items. A *sequence* $S = (j_1, j_2, j_3, \dots, j_m)$ is any series of ordered items in a set of movement sequences (*SDB*), where $j_1 \in I$ is an item for $1 \leq i \leq m$. Concisely, a sequence can be written as $j_1j_2j_3\dots j_m$. It is important to mention that an item j_1 can appear more than once in a sequence S . The size of the sequence S , denoted as $|S|$, is the number of distinct items in the sequence and the length of the same, denoted as $l(S)$, is the total number of items contained in the sequence regardless of the repeated item(s). In other words, the length of a sequence $l(S)$ is the count of all ordered itemset of any given sequence. Based on sequence s above, the length of the sequence i.e. $l(S) = m$. A set of movement sequences (*SDB*) is a list of sequences i.e. $SDB = \{S_1, S_2, \dots, S_n\}$ where each sequence has a unique identifier (i.e. ID).

Definition 4.1.1. Considering two sequences, $S_1 = (x_1x_2x_3 \dots x_m)$ and $S_2 = (y_1y_2y_3 \dots y_n)$, S_1 is contained or is a sub-sequence of S_2 (denoted as $S_1 \sqsubseteq S_2$ or $S_1 \sqsubset S_2$ if $S_1 \neq S_2$), if and only if there exist integers $1 \leq k_1 < k_2 < k_3 < \dots k_m \leq n$ and such that $x_1 = y_{k_1}, x_2 = y_{k_2}, x_3 = y_{k_3}, \dots, x_m = y_{k_m}$. S_1 can be referred to as a snippet

or sub-sequence of S_2 while S_2 can be referred to as a super-sequence of S_1 or said to contain S_1 .

Definition 4.1.2. A sub-sequence s^* is said to be contiguously contained in another (sub) sequence S , if and only if all the ordered items of s^* are exactly contained as the starting itemset of S , whose $l(S)$ must be higher between the two sequences.

Definition 4.1.3. Given a set of movement sequences G and a sub-sequence s^* , the **absolute support** of the sub-sequence s^* in a set of movement sequences G is the total count of sequences in a set of movement sequences G that contains s^* . This is denoted as $Sup_G^a(s^*) = |\{S | S \in G \wedge s^* \sqsubseteq S\}|$. Meanwhile, the percentage share of sequences in G that contains s^* is referred to as its **relative support**. This is also denoted as $Sup_G^r(s^*) = (|\{S | S \in G \wedge s^* \sqsubseteq S\}| / |G|) * 100$.

Definition 4.1.4. Given a set of movement sequences G and a sub-sequence s^* , the sub-sequence s^* is frequent if and only if its support value equals or is greater than the specified **threshold** = “ σ ”.

Definition 4.1.5. A contiguous sequential pattern s in a set of movement sequences G is referred to as a closed contiguous pattern if there exist no other contiguous sequential patterns s' such that s' are super-sequences of s and that the $Sup_G^a(s) == Sup_G^a(s')$.

Definition 4.1.6. l -length closed contiguous sequential patterns is the dictionary of all closed contiguous sequential patterns and its corresponding support value whose maximal length does not exceed l .

The parameter “ l ” is a user-defined parameter which serves as a sliding window for mining patterns. The parameter “ l ”, in this case, ensures that patterns of greater length value are not considered during candidate generation, search space pruning and pattern closure checking phases.

Given a set of sequences G , a user-defined maximal length l and a minimum support threshold σ , the problem of “ l ”-length closed contiguous pattern mining can be concisely described as the extraction of a dictionary of sub-sequences from G (i.e., where the keys contain all frequent closed contiguous patterns and the values correspond to its absolute support score) where the length of any of those key is lesser than or equal to the user-defined length l .

- The second step explored the support pruning method for removing infrequent candidates thereby reducing the search space for checking pattern closure.
- In the third step, the frequencies of the remaining candidates are afterwards processed to identify those that satisfied the closed properties of the contiguous patterns as stated in Definition 4.1.6.

Prior to discussing the proposed algorithm, the approaches taken for the proposed algorithm's patterns candidate generation and search space pruning as well as checking for the closed properties of frequent patterns are justified and discussed below.

4.1.2 Candidate Generation and Search Space Pruning Approach

As revealed in Section 2.4, the common method with which conventional sequential pattern mining algorithms generate sequential pattern candidates is by generating lower length patterns, k -length, and enumerating all possible combinations of longer length, $k+1$ length, patterns before proceeding to check for its frequency. The existing Generalized Sequential Pattern (GSP) algorithm (Srikant and Agrawal, 1996) uses the S-extension and i-extension methods for candidate generation of lower-length patterns before generating potential longer candidate patterns through joins of shorter ones. *PrefixSpan* (Han et al., 2001) perform a recursive projection of a set of sequences into a small projected size and use the level-by-level projection and bi-level projection to grow its potential candidate patterns. Other algorithms such as "CloFAST" (Fumarola et al., 2016) make use of the closed itemset enumeration tree (CIET) to enumerate its complete set of candidate patterns. These common methods of enumerating potential longer candidate patterns (discussed in Section 2.4) are reported to consume more memory and had a track record of increased execution runtime. Additionally, these methods generate false patterns which are not good for some real-world applications. In a real-time (sports) application, the production of non-adjacent items as if they exist within a pattern is unsuitable for tasks involving movement profiling or athletes. Besides, false patterns will lead to gross misrepresentation and wrong locomotive activities reproduction for training programmes and or talent identification in practice.

Hence, the proposed LCCspm algorithm utilizes a snippet-growth approach for candidate pattern generation and considers search space pruning alongside candidate

4.1 l -Length Closed Contiguous Pattern Mining

generation to reduce the total number of patterns to be held in the search space. Candidate patterns are produced by slicing sequences of the input set of sequences into the desired length of maximal patterns starting from 1-length patterns. However, all potential candidates are not generated and all sequences are not sliced to generate candidate patterns in the bid for algorithmic (memory and time) efficiency. The proposed LCCspm algorithm utilizes a dictionary data structure to store generated candidate patterns as they are being sliced out of sequences. A dictionary data structure consists of pairs of keys and values where each key is unique and can be used to reference a (list of) values. This implementation helps to remove repeated candidate patterns on the fly without having to employ another method to check for duplicates. See Example 4.1.2 for an illustration.

More so, all sequences are not considered for candidate pattern generation. Based on the user-specified relative support value, the corresponding absolute support can be calculated and used to extract a *subset* of the given set of sequences or set a range of sequences to consider. Any candidate patterns that can be generated outside of this *subset* of sequences can not satisfy the frequency threshold, thus infrequent. If mined alongside others, they will ultimately be pruned by the support pruning method. Between creating a new subset of sequences and setting the range of sequences to consider from the original set of sequences based on the absolute support value, the latter is memory efficient, a more effective method and was implemented.

Given a set of sequences G , let S be a sequence in G , “ l ” be a user-defined positive integer value and “ σ ” be a placeholder for the user-specified relative support. Candidate patterns are sliced from every sequence with sequence identifier (i.e., $S.ID$) ranging from 1 to $Sup_G^a(s^*)$ (i.e., S_1, S_2, \dots, S_q ; where q is the **absolute support value**). The length of the candidate patterns starts from one-length patterns (sub-sequence) to l -length patterns. The patterns form the dictionary keys and the values are set to a default 0.

Example 4.1.2. Considering the first sequence in Table 2.1, i.e., $S_1 = (aabbcddefcc-cdcgc)$; a relative support threshold “ σ ” = 75%; and length $l = 3$. The dictionary of candidate patterns of 1-length (i.e., C_1) is created by slicing S_1 . The following dictionary, i.e., $\{ 'a': 0, 'b': 0, 'c': 0, 'd': 0, 'e': 0, 'f': 0, 'g': 0 \}$, is being generated. The C_2 dictionary of candidate patterns from S_1 is: $\{ 'aa': 0, 'ab': 0, 'bb': 0, 'bc': 0, 'cd': 0, 'de': 0, 'ef': 0, 'fc': 0, 'cc': 0, 'dc': 0, 'cg': 0, 'gc': 0 \}$. Lastly, the C_3 dictionary

4.1 *l*-Length Closed Contiguous Pattern Mining

of candidate patterns from S_1 contains $\{ 'aab': 0, 'abb': 0, 'bbc': 0, 'bcd': 0, 'cde': 0, 'def': 0, 'efc': 0, 'fcc': 0, 'ccc': 0, 'ccd': 0, 'cdc': 0, 'dgc': 0, 'cgc': 0 \}$.

From Example 4.1.2, the keys of all dictionaries (i.e. C_1 , C_2 and C_3) is seen to contain unique candidate patterns from S_1 . The values, representing the absolute support score, are set to the default value of 0 (as the support is not computed in this step). In total, thirty-two candidate patterns were generated from only the first sequence. However, the size of the candidate patterns does not grow exponentially with respect to the number of sequences in the set of sequences. This is because the dictionary data structure was implemented to store only unique candidate patterns. See Example 3 for details.

Example 4.1.3. *Considering all the sequences in Table 2.1, i.e., $S_1S_2S_3S_4S_5S_6$ and with parameters same as declared in Example 3.2.1, the generated candidate patterns are held in a dictionary as follows: $\{ 'a': 0, 'b': 0, 'c': 0, 'd': 0, 'e': 0, 'f': 0, 'g': 0, 'h': 0, 'i': 0, 'j': 0, 'aa': 0, 'ab': 0, 'bb': 0, 'bc': 0, 'cd': 0, 'de': 0, 'ef': 0, 'fc': 0, 'cc': 0, 'dc': 0, 'cg': 0, 'gc': 0, 'ec': 0, 'cf': 0, 'fg': 0, 'gh': 0, 'hc': 0, 'ce': 0, 'ed': 0, 'dh': 0, 'eg': 0, 'ei': 0, 'ij': 0, 'aab': 0, 'abb': 0, 'bbc': 0, 'bcd': 0, 'cde': 0, 'def': 0, 'efc': 0, 'fcc': 0, 'ccc': 0, 'ccd': 0, 'cdc': 0, 'dgc': 0, 'cgc': 0, 'dec': 0, 'ecf': 0, 'cfg': 0, 'fgh': 0, 'ghc': 0, 'hce': 0, 'ced': 0, 'edh': 0, 'fcd': 0, 'dcd': 0, 'dcf': 0, 'fce': 0, 'ceg': 0, 'egh': 0, 'cei': 0, 'eij': 0 \}$*

From Example 4.1.3, sixty-two candidate patterns were seen to be generated from the sequences database which does not double the size of candidate patterns generated from one sequence in the sequences database (as explained in Example 2). In contrast, assuming a 2-tuple set was used as a data structure to hold candidate patterns, one hundred and sixty-eight candidate patterns would have been held in memory until or unless a conditional check is conducted to remove duplicates.

After generating candidate patterns, they were further pruned for frequency using the support pruning method. Given that candidate patterns were generated without considering all sequences in the sequences database, the support pruning method simply scans the database to count the number of sequences that uniquely contain each candidate pattern. The absolute support for each candidate pattern was stored as the values in the dictionary based on the corresponding key.

The approach for the discovery of closed contiguous patterns among the dictionary keys is discussed in Section 4.1.3

4.1.3 Pattern Closure Checking Approach

As discussed in Section 2.4.3, the use of a lower support threshold when extracting or mining patterns often leads to the extraction of a large number of frequent patterns and in turn, degrades the performance of most algorithms. The pattern closure problem i.e. (frequent closed pattern) was originally formulated as a solution to this problem. A set of closed patterns is a smaller but complete set (i.e. lossless) of results derived from the large set of resulting frequent patterns. Concisely, frequent closed contiguous patterns are a smaller and lossless set of frequent contiguous patterns.

As defined in Definitions 4.1.5 and 4.1.6, closed contiguous patterns are frequent patterns without super-sequences with the same support. Contextually, pattern closure checking involves investigating if a frequent contiguous pattern is contained by a longer frequent contiguous pattern (super-sequence) and whether both patterns have the same support (see Definition 4.1.5). The basic implementation of pattern closure checking entails comparing each pattern among the frequent patterns to identify the set of frequent closed contiguous patterns. This solution is computationally expensive as it is a $O(N^2)$ complexity, where N is the number of patterns.

This thesis explored another solution for pattern closure checking that avoids the $O(N^2)$ complexity. The solution is based on the (and refer to) *inverse characteristic* of frequent closed contiguous patterns which can be described as:

The set of obtainable or extracted sub-sequences (with support value) from a set of super-sequences are non-closed patterns because they are contained in the super-sequence. Therefore, the collection and removal of those extracted sub-sequences will result in discovering the frequent closed contiguous patterns.

Frequent l -length closed contiguous patterns were discovered by extracting sub-sequences from the frequent l -length contiguous patterns, pairing each extracted sub-sequences with its frequent l -length contiguous pattern support value and the final removal of those sub-sequence and support pairs from the dictionary of frequent contiguous patterns. The complexity of this pattern-checking solution as well as candidate generation and support pruning is discussed in the complexity analysis section (Section

4.1.6) after presenting the algorithm itself (Section 4.1.4).

4.1.4 LCCspm Algorithm (Memory-Optimised)

LCCspm is a three-step pattern mining algorithm (discussed in Section 4.1.1) that ultimately finds the closed contiguous patterns whose frequency equals or exceeds a given relative support $Sup_G^r(s^*)$ and whose length does not exceed a user-defined length l . It accepts a two-columned data frame as input, where the first column holds the sequence identifier ($S.id$) and the second column contains the sequences. However, it processes the second column into a list and makes use of the latent list index as $S.id$.

The candidate patterns, frequent contiguous patterns and frequent closed contiguous patterns are stored in a similar data structure: a dictionary $\{s^*, Sup_G^a(s^*)\}$, where s^* is a pattern stored as key and $Sup_G^a(s^*)$ is the absolute support of the pattern. The algorithms produce three outputs in succession. The first is the candidate patterns dictionary which is processed to obtain the second output. The second output is the frequent contiguous patterns dictionary which is also processed to obtain the third output. The third and final output is the frequent closed contiguous pattern dictionary which contains all the closed contiguous patterns in the given sequences database that satisfies the user-defined length l and relative support $Sup_G^r(s^*)$ threshold.

The *LCCspm* algorithm was formulated and presented as a pseudo-code in Algorithm 2. It is made up of three classes (i.e., Mining, Pruning and Closed) representing each of its steps. Given a set of sequences G , a relative support threshold $Sup_G^r(s^*)$ as a threshold (i.e., σ) and a user-defined l , three global variables (i.e., 0, $IandF$) are defined for storing generated candidates, pruned contiguous patterns and l -length closed contiguous patterns. Three class objects (i.e., *explorer*, *prune* and *closure*) were instantiated which correspond to the three stages of *LCCspm* (See lines 1 - 4). The *explorer* object of class “Mining” is responsible for loading and transforming a set of sequences, calculation of sequence with the maximum length, computing the summation of all sequence lengths, calculation of the average length of the sequences in the sequences database, identification of itemsets (i.e. unique items in the sequences database), calculation of the sequences database size (i.e., the total number of sequences) and generation of candidate patterns.

The *prune* object of class “Pruning” converts the user-specified relative support

4.1 l -Length Closed Contiguous Pattern Mining

Algorithm 2 LCCspm (Memory-Optimised) Pseudocode

Input: SDB G , $Sup_G^r(s^*) = \sigma$, maximal-length l

Output: l -length closed contiguous sequential patterns, $SUP^a(s^*)$

```

 $O \leftarrow \emptyset$  // dictionary for storing the generated candidates
 $I \leftarrow \emptyset$  // dictionary for storing the frequent contiguous patterns
 $F \leftarrow \emptyset$  // dictionary for storing  $l$ -length frequent closed contiguous patterns
1:  $explorer \leftarrow \text{Mining}(\text{str}(\text{name}))$  // Instantiate class object "Mining"
2:  $explorer.load\_data(G)$  // loads data  $G$ 
3:  $prune \leftarrow \text{Pruning}(\text{str}(\text{name}), explorer)$  // Instantiate class object "Pruning"
4:  $closure \leftarrow \text{Closed}(\text{str}(\text{name}))$  // Instantiate class object "Closed"
5:  $O = explorer.setl\_mine(\text{maximal-length } l, \sigma)$  // generate contiguous patterns
6:  $I = prune.setsupportprunning(O, \sigma)$  // contiguous patterns support pruning
7:  $F = closure.closed(I)$  // discover closed contiguous patterns
8: return  $F$ 

```

into absolute support, computes the absolute support of candidate patterns, and outputs the dictionary of contiguous patterns and their respective absolute support scores.

The *closure* object of class "Closed" is tasked with identifying closed contiguous patterns from a dictionary of contiguous patterns.

Each class contains defined functions which are inherited by its object. The function "*load_data()*" of the *explorer* object was invoked (Algorithm 2, line 2) for loading a two-column data frame that contains sequence ID and sequences (in this order) and transforming it into a list containing a set of sequences. The pseudo-code of the function "*load_data()*" is presented in Algorithm 3. From Algorithm 3, the function *load_data()* accepts the path to the data frame directory and returns no output. The function performs the processing of the sequences into a list (Algorithm 3, line 2) and computes the size of the set of sequences (Algorithm 3, line 3). The function "*load_data()*" also computes the maximum length of sequences. It declares the placeholder in Algorithm 3, line 4 and runs the code snippet (line 7 - 12) to find the sequence with the maximum length. Also, the total length of all sequences in the set of sequences is computed (Algorithm 3, line 13 - 16) and stored in a placeholder *sdbSummation* (Algorithm 3, line 5). Lastly, the function "*load_data()*" calculates the average length of the set of sequences and stores it in a placeholder (Algorithm 3, line 17).

The function "*setl_mine()*" of the object *explorer* which receives two parameters (maximal-length l , σ) was invoked (Algorithm 2, line 5) for candidate pattern gener-

4.1 l -Length Closed Contiguous Pattern Mining

Algorithm 3 Function: `load_data(G)`

Input: path to data frame G

Output: $\{s^*, 0\}$

```
1:  $data \leftarrow \text{dataframe}(\text{path})$  // stores the 2-columned dataframe
2:  $SDB \leftarrow \text{data.iloc}[0:,1].\text{tolist}()$  // List of sequences
3:  $sdbSize \leftarrow \text{len}(SDB)$  // compute the size of the set of sequences
4:  $sdbMaxLen = 0$ 
5:  $sdbSummation = 0$ 
6:  $sdbAvgLength = 0$ 
7: for sequence in  $SDB$  do
8:    $\_maxl \leftarrow \text{len}(\text{sequence})$ 
9:   if ( $\_maxl > sdbMaxLen$ ) then
10:     $sdbMaxLen = \_maxl$ 
11:   end if
12: end for
13: for sequence in  $SDB$  do
14:    $length \leftarrow \text{len}(\text{sequence})$ 
15:    $sdbSummation += length$ 
16: end for
17:  $sdbAvgLength = \text{round}(sdbSummation / sdbSize)$ 
```

ation through snippet growth and search space pruning. The first parameter, maximal length l , ensures that snippets of candidate patterns do not exceed the value of l . The second parameter, σ , is the relative support, ensuring that sequences that will produce infrequent patterns are not considered during the snippet growth method for candidate pattern generation. The pseudo-code of the function `setl_mine()` is presented in Algorithm 4. It has two local variables, the first is the $Sup_G^a(s^*)$ which is the computed absolute support value. The absolute support is calculated by dividing the supplied relative support σ by 100 and multiplying the result by the $sdbSize$ before rounding the final score (Algorithm 4, line 1). The computed absolute support was used to set the range of sequences to consider during snippet growth candidate pattern generation (Algorithm 4, line 7). The second local variable is declared as an empty dictionary (Algorithm 4, line 2). It stores the snippets (i.e. generated patterns) through dictionary updates and removes duplicates on the fly because the dictionary does not support duplicates (Algorithm 4, line 11). The generation of candidates pattern through snippet growth is performed by generating patterns starting from 1-length contiguous patterns

4.1 l -Length Closed Contiguous Pattern Mining

Algorithm 4 Function: `setl_mine(maximal-length l , σ)`

Input: l , $\sigma = Sup_G^r(s^*)$

Output: GeneratedCandidates = $\{s^*, 0\}$

```

1:  $Sup_G^a(s^*) = \text{round}((\sigma / 100) * sdbSize)$ 
2: GeneratedCandidates  $\leftarrow \emptyset$  // dictionary for storing the generated candidates
3: if  $l == sdbMaxLen$  then
4:    $l = l - 2$ 
5: end if
6: for  $n$  in range  $l$  do
7:   for  $m$  in range  $Sup_G^a(s^*)$  do
8:      $start \leftarrow 0$ 
9:      $end \leftarrow n + 1$ 
10:    while  $end \leq \text{len}(SDB[m])$  do
11:      GeneratedCandidates.update( $\{SDB[m][start : end] : 0\}$ )
12:       $start++ = 1$ 
13:       $end++ = 1$ 
14:    end while
15:  end for
16: end for
17: return GeneratedCandidates

```

up to the user-defined length l contiguous patterns and pairing such patterns with the default value 0 before storing them into the dictionary (Algorithm 4, line 6 - 16). Afterwards, the updated dictionary is returned (Algorithm 4, line 17).

The function “*setsupportprunning()*” of the object *prune* which receives two parameters (O , σ) was invoked (Algorithm 2, line 6) for support pruning. The first parameter, O , is the dictionary of contiguous patterns. The second parameter, σ , at this phase ensures that candidate patterns with relative support that equals or exceeds the set threshold are considered while others are discarded. This function starts the second phase of the *LCCspm* algorithm - the identification of frequent contiguous patterns. Its pseudo-code is presented in Algorithm 5. It computes the equivalent absolute support of contiguous patterns using the given relative support and size of the given set of sequences (Algorithm 5, line 1). This function has one local variable *SupportList*, also declared as an empty dictionary which is used for storing frequent candidate patterns (Algorithm 5, line 2). The function counts the number of sequences that contain each candidate pattern (Algorithm 5, line 3-8) but only considers those whose absolute sup-

4.1 l -Length Closed Contiguous Pattern Mining

Algorithm 5 Function: `setsupportpruning(O, σ)`

Input: $mined, \sigma = Sup_G^r(s^*)$

Output: $SupportList = \{s^*, Sup_G^a(s^*)\}$

```

1:  $Sup_G^a(s^*) = \text{round}((\sigma / 100) * sdbSize)$ 
2:  $SupportList \leftarrow \emptyset$  // dictionary for storing the frequent candidates
3: for  $s^*$  in  $mined.keys()$  do
4:    $supp\_value \leftarrow \text{len}([i \text{ for } i \in SDB \text{ if } s^* \text{ in } i])$ 
5:   if  $supp\_value \geq Sup_G^a(s^*)$  then
6:      $SupportList.update(\{s^* : supp\_value\})$ 
7:   end if
8: end for
9: return  $SupportList$ 

```

port is equal and or greater than the user-specified threshold (Algorithm 5, line 5-7) and store them into the local variable *SupportList*. In the end, it returns a dictionary of frequent contiguous patterns (Algorithm 5, line 9).

The function “*closed()*” of the object *closure* was invoked (Algorithm 2, line 7) to identify the closed contiguous patterns from the dictionary of contiguous patterns. It takes only one parameter, I . The parameter passes the dictionary of contiguous passed into the function. Lastly, the *LCCspm* algorithm output a dictionary of closed contiguous patterns F , where $F = \{s^*, Sup_G^a(s^*)\}$. The pseudo-code of the function *closed()* is presented in Algorithm 6. It has two local variables *ClosedList* and *nonClosed*. *ClosedList* is the dictionary that stores the l -length closed contiguous patterns while *nonClosed* holds the set of extracted sub-sequences. Each frequent contiguous pattern is treated as a super-sequence and its sub-sequences were extracted with the exception of 1-length patterns. For each pair of frequent contiguous pattern and its absolute support value, its sub-sequences were extracted (Algorithm 6, line 4 - 6), each sub-sequences is paired with the super-sequence absolute support value and added to the *nonClosed* local variable (Algorithm 6, line 7). In Algorithm 6 (line 11), All frequent contiguous patterns that are contained in the extracted sub-sequences were removed. The remnants are the l -length closed contiguous patterns stored in *ClosedList* which was returned (Algorithm 6, line 12).

The *LCCspm* (Memory-Optimised) algorithm was implemented using the Python programming language. It was written in the Object-Oriented Programming format. Each object (i.e., *explorer*, *prune* and *close*) was written as a class. It is available

Algorithm 6 Function: $\text{closed}(I)$

Input: SupportList

Output: ClosedList = $\{s', \text{Sup}_G^a(s')\}$

```

1: ClosedList  $\leftarrow \emptyset$  // dictionary for storing  $l$ -length closed contiguous patterns
2: nonClosed  $\leftarrow \text{set}()$  // set of extracted sub-sequences
3: for  $s^*, \text{supp\_value}$  in SupportList do
4:   for width in range(1, len( $s^*$ )) do
5:     for begin in range(len( $s^*$ ) + 1 - width) do
6:        $s^\beta \leftarrow s^*$  [begin: begin + width]
7:       nonClosed.add( $s^\beta, \text{supp\_value}$ )
8:     end for
9:   end for
10: end for
11: ClosedList  $\leftarrow \text{dict}((\text{SupportList.items}() - \text{nonClosed}))$ 
12: return ClosedList

```

as a package which can be cloned or downloaded from GitHub repository ([Adeyemo, 2021a](#)).

4.1.5 LCCspm Algorithm (Time-Optimised)

LCCspm (Memory-Optimised) is designed as a three-step pattern mining algorithm and it is envisaged the support counting phase reduced the algorithm's efficiency. Considering how LCCspm (Memory-Optimised) counts the support of each candidate pattern (Algorithm 5, line 6), it passes over a given set of sequences multiple times (corresponding to the number of candidate patterns). This can be a limitation when extracting frequent patterns, especially from a voluminous set of sequences. Multiple passes on a set of sequences are among the identified limitations of existing apriori-based sequential pattern mining algorithms (discussed in Section 2.4.1) which are resolved by either vertical representation of a given set of sequences or utilization of pattern growth technique (discussed in Section 2.4.2).

The LCCspm (Time-Optimised) algorithm was developed by implementing a form of vertical representation of generated candidate patterns to achieve maximum efficiency in time and cost for extracting closed contiguous frequent patterns. The pattern growth was not considered because it requires recursive scans while being expensive and also this time-optimised LCCspm algorithm is based on the snippet growth ap-

4.1 l -Length Closed Contiguous Pattern Mining

Algorithm 7 LCCspm (Time-Optimised) Pseudocode

Input: SDB G , $Sup_G^r(s^*) = \sigma$, maximal-length l

Output: l -length closed contiguous sequential patterns, $SUP^a(s^*)$

- $J \leftarrow \emptyset$ // set for storing l -length frequent closed contiguous patterns
 - 1: $lccspmObject \leftarrow LCCspm(str(name))$ // Instantiate class object “LCCspm”
 - 2: $lccspmObject.load_data(G)$ // loads a set of sequences G
 - 3: $J = lccspmObj.run(maximal_length\ l, \sigma)$ // discover closed contiguous patterns
 - 4: **return** J
-

proach to avoid false pattern generation. To efficiently count patterns’ support, a set of 2-tuple data structure is utilised to store candidate patterns and index pairs (in this order) as each pattern is generated through the snippet growth method. This data structure was considered because it removes duplicates on the fly. Importantly, the LCCspm (Time-Optimised) algorithm was developed to extract all closed contiguous frequent patterns from a given set of sequences based on a **single** pass while ensuring too much memory was not consumed.

The LCCspm (Time-Optimised) algorithm was formulated and presented as a pseudocode in Algorithm 7. It is made up of just one class (i.e., LCCspm). Given a set of sequences G , a relative support threshold $Sup_G^r(s^*)$ as σ and a user-defined l , one global variable (i.e., J) is defined for storing l -length frequent closed contiguous patterns. The class object (i.e., **lccspmObject**) was instantiated to run the LCCspm (Time-Optimised) algorithm (Algorithm 7, line 1). The “lccspmObject” object of class “LCCspm” performs all the necessary tasks for the algorithm to run, including loading and transforming a set of sequences, calculation of sequence with the maximum length, computing the summation of all sequence lengths, calculation of the average length of the sequences in the sequences database, identification of itemsets (i.e. unique items in the sequences database), calculation of the sequences database size (i.e., the total number of sequences), generation of candidate patterns and extraction of frequent closed contiguous patterns. The function **run** of “lccspmObject” object (Algorithm 7, line 3) executes the LCCspm (Time-Optimised) algorithm.

The pseudo-code of the function **run** is presented in Algorithm 8. Instead of counting the support of each candidate pattern, the LCCspm (Time-Optimised) algorithm first paired up each candidate pattern (snippet) with the index of the sequences where it was sliced and add such pair to a set of 2-tuple (Algorithm 8, lines 2-15). Afterwards,

4.1 l -Length Closed Contiguous Pattern Mining

Algorithm 8 Function: run(maximal-length l, σ)

Input: $l, \sigma = Sup_G^r(s^*)$

Output: ClosedContiguousFrequentPatterns = $\{s', Sup_G^a(s')\}$

```
1:  $Sup_G^a(s^*) = \text{round}((\sigma / 100) * sdbSize)$ 
2: Candidates  $\leftarrow \emptyset$  // set for storing the candidates pattern and index pairs
3: for  $n$  in range  $l$  do
4:   for  $m$  in range  $G$  do
5:      $start \leftarrow 0$ 
6:      $end \leftarrow n + 1$ 
7:     while  $end \leq \text{len}(G[m])$  do
8:        $pattern = G[m][start:end]$ 
9:        $pair (pattern, m+1)$ 
10:      Candidate.add(pair)
11:       $start++ = 1$ 
12:       $end++ = 1$ 
13:    end while
14:  end for
15: end for
16: FrequentCandidates  $\leftarrow \emptyset$ 
17: Candidates = sorted (Candidates, key=operator.itemgetter(0))
18: for pattern, index in groupby(Candidates, operator.itemgetter(0)) do
19:    $pattern\_Indexes = (pattern, [item[1:] \text{ for } item \text{ in } indexes])$ 
20:    $absSupport = \text{len}(pattern\_Indexes[1])$ 
21:   if  $absSupport \geq Sup_G^a(s^*)$  then
22:      $frequentPair = (pattern, absSupport)$ 
23:     FrequentCandidates.add(frequentPair)
24:   end if
25: end for
26: nonClosed  $\leftarrow \text{set}()$  // set of extracted sub-sequences
27: for  $s^*, supp\_value$  in FrequentCandidates do
28:   for width in range(1, len( $s^*$ )) do
29:     for begin in range(len( $s^*$ ) + 1 - width) do
30:        $s^\beta \leftarrow s^* [begin: begin + width]$ 
31:       nonClosed.add( $s^\beta, supp\_value$ )
32:     end for
33:   end for
34: end for
35: ClosedContiguousFrequentPatterns  $\leftarrow$  FrequentCandidates – nonClosed
36: return ClosedFrequentCandidates
```

(Algorithm 8, lines 16-25), extracted pairs of candidate patterns and indexes are sorted and grouped by patterns where multiple indexes of a unique pattern are temporarily stored in a list. The length of the list is then computed to obtain the absolute support of each candidate pattern without re-scanning a given set of sequences. The obtained absolute support is used to test each pattern frequency and only those pairs that are frequent are retained. From Algorithm 8 (lines 26- 36), the closed contiguous patterns are discovered from the frequent contiguous patterns and returned.

4.1.6 LCCspm Complexity Analysis

The *LCCspm* algorithm running time is mostly spent on checking for support. Before support pruning, the algorithm spends meagre time splitting and storing snippets from sequences. However, the time spent for snippet splitting and checking (i.e., candidate generation) for duplicates is greatly reduced due to implemented dictionary structure for storing the snippet. The support pruning process computes support for each candidate pattern. For each candidate, it scans the database to compute its absolute support. Both the candidate generation and pruning processes run at $O(n)$ complexity where n is the length of the sequences database during candidate generation and the total number of candidate patterns during the support pruning stage.

LCCspm technique for finding all closed contiguous patterns from a dictionary of contiguous patterns is linearly scalable at the complexity of

$$O(n * d^2)$$

In order to identify closed contiguous patterns among the frequent contiguous patterns, *LCCspm* was designed and developed to avoid a function in a quadratic order by having one outer loop that runs at $O(n)$ times and two inner loops running at $O(d^2)$ complexity where n is the length of frequent dictionary that stored the frequent contiguous pattern and d is the length of each pattern whose sub-patterns are extracted.

In other words, the *LCCspm* algorithm (Algorithm 6) for identifying closed contiguous patterns runs at $O(n)$ with a scale factor of the function d^2 . And in each of the three steps of *LCCspm*, the *LCCspm* computational complexity maintains a $O(n)$ complexity.

4.2 Evaluating LCCspm Algorithm (Memory-Optimised)

This section contains the datasets, experiment settings, results and discussion of evaluating the LCCspm algorithm (memory-optimised) algorithm.

4.2.1 Datasets and Experimental Setting

This comparative analysis considered two real-time case studies of rugby league players' movements and one real-time study of international football players' match events to compare the performances of LCCspm and CCSpan.

For the first case study of rugby league players' movement pattern extraction, the SMP framework method (discussed in Section 3.1.2) for processing GPS data into discrete movement sequences were utilized. GPS data of thirty-three Super League players that participated in a single Rugby Football League (RFL) match was processed into movement sequences. The elite players played for Salford Red Devils and Winger Warriors during the 2019 season. A set of discrete movement sequences was generated from the GPS data. This set of sequences is referred to as "Small sequential player movements data". The second case study of rugby league players' movement pattern extraction involved the processing of a total of eighty-seven Super League players that participated in five RFL matches. These elite players participated in the Castleford Tigers vs. Warrington wolves, Salford red devils vs. Wigan warriors, Salford red devils vs. Castleford tigers, St. Helens vs. Wigan warriors and St. Helens vs. Salford red devils match games. This set of sequences data is referred to as "Large sequential player movements data". Both data were developed to test the algorithms' performance on both small and large volumes of sequences.

For the case study of international football players' match event extraction, the men's FIFA 2018 world cup match event data was assessed and utilised. The men's FIFA 2018 world cup was coded and publicly made available by StatsBomb (Sta). Match events comprise technical and tactical events that occurred during matches or training, they are video-sourced and usually coded by experts. The data contained 64 matches and specifies events types, teams, players, shots, possession, play pattern and timestamp. This data was specially developed to test the algorithms' scalability on lengthy sequences.

4.2 Evaluating LCCspm Algorithm (Memory-Optimised)

Table 4.2: FIFA 2018 Match Event Dictionary ([Adeyemo, 2023](#))

Match Event Type	Code
Starting XI	a
Half Start	b
Pass	c
Ball Receipt*	d
Carry	e
Pressure	f
Duel	g
Shot	h
Goal Keeper	i
Camera On	j
Miscontrol	k
Ball Recovery	l
Clearance	m
Block	n
Foul Committed	o
Foul Won	p
Camera off	q
Dribble	r
Dispossessed	s
Dribbled Past	t
Interception	u
50/50	v
Injury Stoppage	w
Player Off	x
Referee Ball-Drop	y
Player On	z
Half End	A
Substitution	B
Shield	C
Error	D
Offside	E
Tactical Shift	F

Each unique match event *type* was encoded as an alphabetical character (See Table 4.2) and was processed sequentially based on the *timestamp*. Each match was encoded into one sequence. Thus, the set of match events sequences contained 64 sequences of encoded match event *types*. This set of match event sequences is referred to as “FIFA match event data”. The average length of the sequences is 3,561 and the maximum length is 5,026.

These sets of sequences (i.e., both small and large sequential player movement data and the FIFA match event data) were analysed to conduct the comparative analysis experiments between the LCCspm and CCSpan sequential pattern mining algorithms for discovering closed contiguous patterns. Comprehensive performance experiments were conducted, by using various relative support thresholds (e.g., 25%, 50%, and 75%), to evaluate both algorithms. The runtime and memory consumed as well as the total number of closed contiguous patterns found by both LCCspm and CCSpan were obtained for performance evaluation per experiment. The performance of LCC-

4.2 Evaluating LCCspm Algorithm (Memory-Optimised)

spm was assessed in terms of execution runtime and memory consumption (Zhang et al., 2015). The runtime was measured in seconds and the memory consumption was measured in megabytes. Likewise, the performance of the CCSpan algorithm was also measured using the same metrics. Lastly, the resulting frequent closed contiguous subsequences using both algorithms were cross-examined for similarity and or discrepancy.

Afterwards, the performance of the LCCspm (Time-Optimised) algorithm was evaluated and compared against the state-of-the-art closed contiguous pattern mining algorithm - CCSpan and the LCCspm (Memory-Optimised) algorithm. The algorithms were applied to the large sequential data containing 38,366 movement sequences of rugby league players to extract frequent movement patterns. Again, the performance of the algorithms for frequent movement pattern extraction was based on 5%, 25%, 50% and 75% relative support thresholds. The length of obtainable movement patterns varied from one-length movement patterns to maximal-length movement patterns. These comparative analysis experiments were carried out on an Intel(R) Corei5 @ 1.60GHz with 8GB (7.87GB usable) of RAM, running a Windows 10 Education (64-bit) operating system.

4.2.2 Results and Discussion

The results of the LCCspm algorithm on both small and large sequential player movement data and men's FIFA 2018 match event sequence data are presented and discussed.

4.2.2.1 Small Sequential Data

This data contained 7,818 sequences of movement units with an average length of 61. Table 4.3 presents the lengths of discovered patterns per relative support threshold, the runtime and memory consumed as well as the total number of closed contiguous patterns found by both LCCspm and CCSpan. From Table 4.3, LCCspm results are obtained following an incremental iteration (based on user-defined lengths) until the maximal l value per relative support threshold. Whereas, the CCSpan algorithm only produces a one-time result per relative support threshold.

4.2 Evaluating LCCspm Algorithm (Memory-Optimised)

At a 5% relative support threshold, the state-of-the-art algorithm (CCSpan) algorithm discovered six hundred and thirty-seven frequent closed contiguous patterns. On the other hand, the LCCspm algorithm incremental discovered thirty-six 1_{fcp} (i.e., 1-length frequent closed contiguous patterns) within 0.48 seconds while consuming 31.7 megabytes of memory. When the l parameter was set to 4, the LCCspm discovered four hundred and eighty-three 4_{fcp} while it took 66.56 seconds and consumed 36.7 megabytes of memory. LCCspm also discovered six hundred and thirty-seven 7_{fcp} , when the parameter l was set to 7. The patterns and their count tallies with those discovered by CCSpan. However, LCCspm discovered those 7_{fcp} from the small player movement data within 601.44 seconds of runtime and consumed 82.3 megabytes of computer memory. Whereas, CCSpan consumed 263 megabytes of computer memory and took 2,534 seconds of runtime.

When the relative support threshold was increased to 25%, the state-of-the-art algorithm (CCSpan) algorithm discovered a one-time result of eighty frequent closed contiguous patterns. When the LCCspm algorithm parameter l was set to 1 and the σ parameter was set to 25%, seventeen 1_{fcp} were discovered which took 0.18 seconds and consumed 31.9 megabytes of computer memory. LCCspm discovered sixty-two 3_{fcp} at 25% relative support. The analysis took 6.49 seconds and consumed 32.9 megabytes of computer memory. Eighty 5_{fcp} were discovered at 25% relative support which took 78.26 seconds of runtime and 43.5 megabytes of computer memory. Again, the patterns and pattern count discovered by both LCCspm and CCSpan are tallied. However, it took the CCSpan algorithm 364.018 seconds of execution runtime and 59 megabytes of memory to identify the same closed contiguous pattern under the same parameter condition.

At 50% relative support, both algorithms discovered a maximum of 3-length frequent closed contiguous patterns. Both LCCspm and CCSpan algorithms discovered twenty-six 3_{fcp} . It took CCSpan 26 seconds of runtime and 50 megabytes of computer memory. Meanwhile, LCCspm discovered the same patterns within 5.63 seconds and consumed only 32.7 megabytes of computer memory. Besides, LCCspm produced was able to produce the same frequent closed contiguous patterns at lower lengths for lower runtime and lesser memory consumption.

4.2 Evaluating LCCspm Algorithm (Memory-Optimised)

Table 4.3: Performance Evaluation Results on Small Player Movement Sequential Data (Adeyemo et al., 2021)

Support	5%						25%						50%			75%	
length	1	2	3	4	5	6	7	1	2	3	4	5	1	2	3	1	2
LCCSpm_PatternCount	36	162	373	483	540	609	637	17	48	62	78	80	11	19	26	6	9
CCSpan_PatternCount	637						80						26			9	
LCCSpm_Runtime (secs)	0.48	2.71	15.48	66.56	124.67	278.55	601.44	0.18	0.99	6.49	27.78	78.26	0.18	1.12	5.63	0.12	0.93
CCSpan_Runtime (secs)	2534						364.018						13.911			2.25	
LCCSpm_Memory (MB)	31.7	32.6	32.7	36.7	44.7	57.8	82.3	31.9	32.4	32.9	35.2	43.5	31.7	32.1	32.7	31.8	32
CCSpan_Memory (MB)	263						59						50			52	

4.2 Evaluating LCCspm Algorithm (Memory-Optimised)

Table 4.4: Small SDB Top-15 Patterns ($l = 7, \sigma = 5\%$) (Adeyemo et al., 2021)

Encoded	Support	Decoded Movements
b	6746	WalkDecelerationAcute-Change
v	6700	JogAccelerationAcute-Change
u	6598	JogAccelerationStraight
j	6591	WalkAccelerationAcute-Change
uv	6155	JogAccelerationStraight, JogAccelerationAcute-Change
i	6139	WalkAccelerationStraight
vu	6105	JogAccelerationAcute-Change, JogAccelerationStraight
vv	6031	JogAccelerationAcute-Change, JogAccelerationAcute-Change
n	6008	JogDecelerationAcute-Change
a	5630	WalkDecelerationStraight
m	5539	JogDecelerationStraight
uu	5505	JogAccelerationStraight, JogAccelerationStraight
uvv	5169	JogAccelerationStraight, JogAccelerationAcute-Change, JogAccelerationAcute-Change
r	5081	JogNeutralAcute-Change
vvu	5051	JogAccelerationAcute-Change, JogAccelerationAcute-Change, JogAccelerationStraight

The 75% relative support was the highest support threshold set for the experiments on the small player movement data. Nine $2_{f_{ccp}}$ were discovered by both LCCspm and CCSpan algorithms. CCSpan consumed 52 megabytes of computer memory and spent 2.25 seconds to discover those nine $2_{f_{ccp}}$. Besides the discovery of six $1_{f_{ccp}}$ at 75% with a shorter runtime and lesser compute, LCCspm also discovered those nine $2_{f_{ccp}}$ within 0.93 seconds of runtime and consumed 32 megabytes of computer memory.

Across the results per relative support thresholds, it is seen that the length of discovered frequent patterns reduces as the value for support increases for both LCCspm and CCSpan. This indicates that the higher the set support threshold, the lesser the number of discoverable frequent closed contiguous patterns. This is in tandem with studies (Fournier-Viger et al., 2017; Mabroukeh and Ezeife, 2010) that comparatively evaluate pattern mining algorithms' performances.

At the cross-examination of the performance results of LCCspm parameters $l = 7, \sigma = 5\%$ with CCSpan, LCCspm is four times faster than CCSpan and it uses 69% lesser memory as consumed by CCSpan. The cross-examination of the LCCspm performance at $l = 5$ and $\sigma = 25\%$ against CCSpan revealed that LCCspm was approximately five times faster than CCSpan while it uses about 26% lesser memory as consumed by CCSpan. LCCspm performance at $l = 3$ and $\sigma = 50\%$ was approximately three time faster than CCSpan while using about 35% lesser memory that was consumed by CC-

4.2 Evaluating LCCspm Algorithm (Memory-Optimised)

Table 4.5: Small SDB Bottom-15 Patterns ($l = 7, \sigma = 5\%$) (Adeyemo, 2023)

Encoded	Support	Decoded Movements
ijji	398	Walk Acceleration Straight, Walk Acceleration Acute-Change, Walk Acceleration Straight, Walk Acceleration Straight
nmnmn	397	Jog Deceleration Acute-Change, Jog Deceleration Straight, Jog Deceleration Acute-Change, Jog Deceleration Straight, Jog Deceleration Acute-Change
mp	397	Jog Deceleration Straight, Jog Deceleration Backwards
nnno	395	Jog Deceleration Acute-Change, Jog Deceleration Acute-Change, Jog Deceleration Acute-Change, Jog Deceleration Acute-Change, Jog Deceleration Large-Change
oc	395	Jog Deceleration Large-Change, Walk Deceleration Large-Change
nmnmn	395	Jog Deceleration Acute-Change, Jog Deceleration Straight, Jog Deceleration Straight, Jog Deceleration Straight, Jog Deceleration Acute-Change
TSSS	394	Sprint Acceleration Acute-Change, Sprint Acceleration Straight, Sprint Acceleration Straight, Sprint Acceleration Straight
vuvw	393	Jog Acceleration Acute-Change, Jog Acceleration Straight, Jog Acceleration Acute-Change, Jog Acceleration Large-Change
ffi	393	Walk Neutral Acute-Change, Walk Neutral Acute-Change, Walk Acceleration Straight
rnmm	392	Jog Neutral Acute-Change, Jog Deceleration Acute-Change, Jog Deceleration Acute-Change, Jog Deceleration Straight
mqr	392	Jog Deceleration Straight, Jog Neutral Straight, Jog Neutral Acute-Change
fjf	392	Walk Neutral Acute-Change, Walk Acceleration Acute-Change, Walk Neutral Acute-Change
vwu	392	Jog Acceleration Acute-Change, Jog Acceleration Large-Change, Jog Acceleration Straight
aef	391	Walk Deceleration Straight, Walk Neutral Straight, Walk Neutral Acute-Change
HHHG	391	Run Acceleration Acute-Change, Run Acceleration Acute-Change, Run Acceleration Acute-Change, Run Acceleration Straight

Span. Lastly, LCCspm performance at $\sigma = 75\%$ and $l = 2$ was at least two times faster than CCSpan in runtime execution and used 38% lesser memory. Furthermore, the performance results of lower iteration of LCCspm l values at 5%, 25%, 50% and 75% relative support thresholds reveal that the algorithm developed in this PhD study is consistent in its use of lesser memory and took shorter execution runtime.

The top-15 patterns found in the small player movement sequences data using the LCCspm parameters $l = 7, \sigma = 5\%$ are presented in Table 4.4 alongside its frequency and its decoded movements sequence. The top patterns comprise one-length frequent closed contiguous patterns to three-length frequent closed contiguous patterns. The most frequent two-length closed contiguous pattern is “uv” denoted as Jog Acceleration Straight and Jog Acceleration Acute-Change. The pattern had an absolute support of 6155 (i.e., it is contained in 6155 of 7,818 sequences of the small player movement data). The topmost three-length pattern uvv had absolute support of 5169 sequences. This pattern is decoded as Jog AccelerationStraight and [Jog Acceleration Acute-Change]x2 performed contiguously by RFL Super League players that participated in the match between Salford red devils and Winger warriors.

On the contrary, the least performed fifteen movement patterns found in the small player movement sequences data using the LCCspm parameters $l = 7, \sigma = 5\%$ are presented in Table 4.5 alongside its frequency and its decoded movements sequence. This

4.2 Evaluating LCCspm Algorithm (Memory-Optimised)

set of least performed movement patterns comprises two-length frequent closed contiguous patterns to five-length frequent closed contiguous patterns. At the top of the list (Table 4.5) is the four-length closed contiguous pattern *ijii* denoted as Walk Acceleration Straight, Walk Acceleration Acute-Change and [Walk Acceleration Straight]x2. The pattern had an absolute support of 398 (i.e., it is contained in 398 of 7,818 sequences of the small player movement data). Other interesting discovered movement patterns include the three-length *mqr* pattern denoted as Jog Deceleration Straight, Jog Neutral Straight, Jog Neutral Acute-Change with an absolute support value of 392. A four-length movement pattern *TSSS* denoted as Sprint Acceleration Acute-Change and [Sprint Acceleration Straight]x3 is also an interesting pattern which had an absolute support of 394. Another discovered movement pattern of interest is the *HHHG* decoded as [Run Acceleration Acute-Change]x3 and Run Acceleration Straight which had a 391 absolute support value. The five-length pattern *nmnmn* (Table 4.5) had absolute support of 397 sequences. This pattern, decoded as [Jog Deceleration Acute-Change and Jog Deceleration Straight]x2 and Jog Deceleration Acute-Change, was performed contiguously by RFL Super League players that participated in the match between Salford red devils and Winger warriors.

4.2.2.2 Large Sequential Data

This data contained 38,366 sequences of movement units with an average length of 71. Table 4.6 presents the lengths of discovered patterns per relative support threshold, the runtime and memory consumed as well as the total number of closed contiguous patterns found by both LCCspm and CCSpan. From Table 4.6, LCCspm results are obtained following an incremental iteration (based on user-defined lengths) until the maximum l value per relative support threshold. Once again, the CCSpan algorithm only produces a one-time result per relative support threshold. At a 5% relative support threshold, the state-of-the-art algorithm (CCSpan) algorithm for mining closed contiguous patterns could not analyse the large player movement sequential data, upholding its inability to scale well (Abboud et al., 2017; Bermingham and Lee, 2020) on a large set of sequences. LCCspm discovered seven hundred and eighty-three 7_{fcp} , when the parameter l was set to 7 at an execution runtime of 5,399 seconds and consumed 161.5 megabytes of used computer memory.

4.2 Evaluating LCCspm Algorithm (Memory-Optimised)

Table 4.6: Performance Evaluation Results on Large Player Movement Sequential Data (Adeyemo et al., 2021)

Support	5%							25%							50%				75%	
length	1	2	3	4	5	6	7	1	2	3	4	5	1	2	3	4	1	2		
LCCSpm Pattern Count	34	152	393	542	642	737	783	17	50	72	93	102	12	23	30	32	7	10		
CCSpan Pattern Count	-							102							32				10	
LCCSpm Runtime (secs)	2.10	12.52	71.65	420.72	983.56	2465.14	5399.00	1.30	7.73	61.26	252.33	852.28	0.80	5.13	42.22	258.70	0.58	5.48		
CCSpan Runtime (secs)	-							2902.387							544.31				38.376	
LCCSpm Memory (MB)	36.5	37.2	38.4	42.2	58.8	95	161.5	36.5	36.7	37.7	41.7	62	35.5	35.7	37.6	40.8	35.5	35.7		
CCSpan Memory (MB)	-							96							91				82	

(-) indicates unable to analyse

4.2 Evaluating LCCspm Algorithm (Memory-Optimised)

Table 4.7: Large SDB Top-15 Patterns ($l = 7, \sigma = 5\%$) (Adeyemo et al., 2021)

Encoded	Support	Decoded Movements
b	33713	WalkDecelerationAcute-Change
i	32596	WalkAccelerationStraight
j	32561	WalkAccelerationAcute-Change
v	32484	JogAccelerationAcute-Change
u	32315	JogAccelerationStraight
uv	30561	JogAccelerationStraight, JogAccelerationAcute-Change
vu	30494	JogAccelerationAcute-Change, JogAccelerationStraight
n	29926	JogDecelerationAcute-Change
a	29190	WalkDecelerationStraight
uu	28823	JogAccelerationStraight, JogAccelerationStraight
vv	28408	JogAccelerationAcute-Change, JogAccelerationAcute-Change
m	28259	JogDecelerationStraight
uuv	26258	JogAccelerationStraight, JogAccelerationStraight, JogAccelerationAcute-Change
vu	26073	JogAccelerationAcute-Change, JogAccelerationStraight, JogAccelerationStraight
nn	25312	JogDecelerationAcute-Change, JogDecelerationAcute-Change

At a 25% relative support, the CCSpan algorithm produced a one-time result that discovered one hundred and two closed contiguous patterns. The LCCspm algorithm discovered fifty 2_{fcp} which took 7.73 seconds and consumed 36.7 megabytes of computer memory when its parameter l was set to 1 and the σ parameter was set to 25% relative support. LCCspm also discovered ninety-three 4_{fcp} at 25% relative support. The analysis took 252.33 seconds and consumed 41.7 megabytes of computer memory. A one hundred and two 5_{fcp} were discovered at 25% relative support which took 852.28 seconds of runtime and 62 megabytes of computer memory. However, with the same relative support threshold of 25%, the state-of-the-art algorithm (i.e., CCSpan) same set of frequent closed contiguous patterns but took 2,902.39 seconds of execution runtime and 96 megabytes of computer memory.

At 50% relative support, both algorithms discovered a maximum of 4-length frequent closed contiguous patterns (i.e., 4_{fcp}). Both LCCspm and CCSpan algorithms discovered thirty-two 4_{fcp} . It took CCSpan 544.31 seconds of execution runtime and 91 megabytes of computer memory. Meanwhile, LCCspm discovered the same patterns at 258.70 seconds and consumed only 40.8 megabytes of computer memory. LCCspm was again able to produce frequent closed contiguous patterns at lower lengths for lower runtime and lesser memory consumption. For example, LCCspm with parameters set at $l = 2$ and $\sigma = 50\%$ relative support, discovered twenty-three 2_{fcp} at 5.13 seconds and for 35.7 megabytes of computer memory.

4.2 Evaluating LCCspm Algorithm (Memory-Optimised)

Table 4.8: Large SDB Bottom-15 Patterns ($l = 7, \sigma = 5\%$) (Adeyemo, 2023)

Encoded	Support	Decoded Movements
naa	1964	Jog Deceleration Acute-Change, Walk Deceleration Straight, Walk Deceleration Straight
uvvvuu	1963	Jog Acceleration Straight, Jog Acceleration Acute-Change, Jog Acceleration Acute-Change, Jog Acceleration Acute-Change, Jog Acceleration Straight, Jog Acceleration Straight, Jog Acceleration Straight
nnmmn	1961	Jog Deceleration Acute-Change, Jog Deceleration Acute-Change, Jog Deceleration Acute-Change, Jog Deceleration Straight, Jog Deceleration Straight, Jog Deceleration Acute-Change
nnmmn	1956	Jog Deceleration Acute-Change, Jog Deceleration Acute-Change, Jog Deceleration Straight, Jog Deceleration Straight, Jog Deceleration Acute-Change, Jog Deceleration Acute-Change
eeeeef	1956	Walk Neutral Straight, Walk Neutral Straight, Walk Neutral Straight, Walk Neutral Straight, Walk Neutral Straight, Walk Neutral Acute-Change, Walk Neutral Straight
qm	1955	Jog Neutral Straight, Jog Deceleration Straight, Jog Neutral Straight,
uvvvuu	1954	Jog Acceleration Straight, Jog Acceleration Acute-Change, Jog Acceleration Straight, Jog Acceleration Acute-Change, Jog Acceleration Acute-Change, Jog Acceleration Straight, Jog Acceleration Straight
bd	1954	Walk Deceleration Acute-Change, Walk Deceleration Backwards
fffee	1951	Walk Neutral Acute-Change, Walk Neutral Acute-Change, Walk Neutral Acute-Change, Walk Neutral Straight, Walk Neutral Straight, Walk Neutral Straight
fjf	1933	Walk Neutral Acute-Change, Walk Acceleration Acute-Change, Walk Neutral Acute-Change
CD	1929	Run Neutral Straight, Run Neutral Acute-Change
HGGH	1928	Run Acceleration Acute-Change, Run Acceleration Straight, Run Acceleration Straight Run Acceleration Acute-Change
qrqr	1927	Jog Neutral Straight, Jog Neutral Acute-Change, Jog Neutral Straight, Jog Neutral Acute-Change
rvq	1925	Jog Neutral Acute-Change, Jog Acceleration Acute-Change, Jog Neutral Straight
iee	1922	Walk Acceleration Straight, Walk Acceleration Straight, Walk Neutral Straight, Walk Neutral Straight

The 75% relative support was also the highest support threshold set for the experiments on the large player movement data. Ten $2_{f_{ccp}}$ were discovered by both LCCspm and CCSpan algorithms. CCSpan used 82 megabytes of computer memory and spent 38.38 seconds to discover those ten $2_{f_{ccp}}$. Besides the discovery of seven $1_{f_{ccp}}$ at 75% with a shorter runtime and lesser compute, LCCspm also discovered those ten $2_{f_{ccp}}$ at 5.48 seconds of execution runtime and 35.7 megabytes of computer memory.

Across the results per relative support thresholds, the length of discovered frequent patterns is seen to reduce as the value for support increases for both LCCspm and CCSpan. This also reinforces that, the higher the set support threshold, the lesser the number of discoverable frequent closed contiguous patterns. From the analysis results of the large player movement data for both algorithms, the performance results of LCCspm parameters $l = 2, \sigma = 75\%$ with CCSpan revealed LCCspm is seven times faster than CCSpan and it uses only 44% of the total memory consumed by CCSpan. LCCspm performance at $l = 4$ and $\sigma = 50\%$ was approximately two time faster than CCSpan while it used about 45% of the total memory used by CCSpan. Lastly, LCCspm performance at $\sigma = 25\%$ and $l = 5$ was at least three times faster than CCSpan in runtime execution and used 64% of the total memory used by CCSpan. Also, the results of lower iteration of LCCspm l values are consistently lower than higher values (See Table 4.6).

Having cross-examined the outputted patterns from both algorithms and found them matching, Table 4.7 reveals the top 15 patterns found in the large SDB based on parameters $l = 7, \sigma = 5\%$, its frequency and its decoded movements. The top-15 move-

4.2 Evaluating LCCspm Algorithm (Memory-Optimised)

ment patterns comprise one-length frequent closed contiguous patterns to three-length frequent closed contiguous patterns. The most frequent two-length closed contiguous pattern is “*uv*” denoted as Jog Acceleration Straight and Jog Acceleration Acute-Change. The pattern had an absolute support of 6155 (i.e., it is contained in 33,713 of 38,366 sequences of the large player movement data). The topmost three-length pattern “*uvv*” had an absolute support of 26,258 sequences. This pattern is decoded as [Jog Acceleration Straight]x2 and Jog Acceleration Acute-Change performed contiguously by RFL Super League players in five matches.

The bottom-15 movement patterns that were discovered from the large player movement data comprise two-length frequent closed contiguous patterns to seven-length patterns (Table 4.8). At the top of the list in Table 4.8 is the three-length closed contiguous pattern *naa* denoted as Jog Deceleration Acute-Change, [Walk Deceleration Straight]x2. The pattern had absolute support of 1,964 (i.e., it is contained in 1,964 of 38,366 sequences of the large player movement data). Other discovered interesting movement patterns include the three-length *qmq* pattern denoted as Jog Neutral Straight, Jog Deceleration Straight, and Jog Neutral Straight with an absolute support value of 1,955. A six-length movement pattern *nnmmnn* denoted as [Jog Deceleration Acute-Change]x2, [Jog Deceleration Straight]x2 and [Jog Deceleration Acute-Change]x2 is also an interesting pattern which had absolute support of 1,956. Another discovered movement pattern of interest is the *uvvvvvu* decoded as [Jog Acceleration Straight and Jog Acceleration Acute-Change]x2, Jog Acceleration Acute-Change and [Jog Acceleration Straight]x2 which had a 1,954 absolute support value. The four-length pattern *HGGH* (Table 4.8) had absolute support of 397 sequences. This pattern, decoded as Run Acceleration Acute-Change, [Run Acceleration Straight]x2 and Run Acceleration Acute-Change, was performed contiguously by RFL Super League players that participated in the selected five match games.

Again, the performance results of lower iteration of LCCspm *l* values at 5%, 25%, 50% and 75% relative support thresholds reveal that the algorithm developed in this thesis is consistent in its use of lesser memory and took shorter execution runtime.

4.2.2.3 Men's FIFA 2018 World Cup Match-event Data

This data contained 64 sequences of encoded match events. The average length of the sequences is 3,561 and the maximum length is 5,026. Table 4.9 presents the lengths of discovered patterns per relative support threshold, the runtime and memory consumed as well as the total number of closed contiguous patterns found by both LCCspm and CCSpan. LCCspm results are obtained following an incremental iteration (based on user-defined lengths) until the maximum l value per relative support threshold. Across the relative support thresholds, the CCSpan algorithm only produces a one-time result per relative support threshold.

From Table 4.9, the state-of-the-art algorithm (i.e., CCSpan) could not analyse the set of sequences for support threshold lower or equal to 25% when tested. This experimental outcome further upholds CCSpan inability to scale properly even on a small set of sequences with long sequences. On the contrary, at 25% relative support and $l = 10$, the LCCspm algorithm discovered one thousand eight hundred and eighty-six 10_{fccp} (i.e., 10-length frequent closed contiguous patterns) after spending 124.95 seconds of execution runtime. When the l parameter was set to 20, the LCCspm discovered three thousand eight hundred and sixteen 20_{fccp} and it took a runtime of 598.02 seconds. LCCspm also discovered four thousand two hundred and forty-seven 40_{fccp} , when the parameter l was set to 40. LCCspm discovered those 40_{fccp} from the men's FIFA 2018 world cup match event set of sequences having an execution runtime of 1349.93 seconds when its parameter σ was set to 25% relative support.

At 50% relative support, both algorithms discovered a maximum of 29-length frequent closed contiguous patterns (i.e., 29_{fccp}). Both LCCspm and CCSpan algorithms discovered one thousand and sixty-eight 29_{fccp} . It took CCSpan 5446.81 seconds of execution runtime and 134 megabytes of computer memory while LCCspm discovered the same patterns at 563.77 seconds and consumed only 62 megabytes of computer memory. LCCspm produced frequent closed contiguous patterns at lower lengths for lower runtime and lesser memory consumption. For example, LCCspm parameters set at $l = 15$ and $\sigma = 50\%$ relative support, discovered eight hundred and seventy-two movement patterns at runtime of 190.87 seconds and for 62 megabytes of computer memory. The 75% relative support was also the highest support threshold set for the experiments on the men's FIFA 2018 world cup match events data.

4.2 Evaluating LCCspm Algorithm (Memory-Optimised)

Table 4.9: Performance Evaluation Results on Men's FIFA 2018 World Cup (Adeyemo et al., 2021)

Support (%)	25				50				75			
length	10	20	30	40	15	20	25	29	3	6	9	18
LCCSpm_PatternCount	1886	3816	4238	4247	872	1021	1057	1068	69	138	120	120
CCSpan_PatternCount	-				1068				120			
LCCSpm_Runtime (secs)	124.95	598.02	913.47	1349.93	190.87	321.45	458.25	563.77	0.79	10.75	32.98	143.77
CCSpan_Runtime (secs)	-				5446.81				650.23			
LCCSpm_Memory (MB)	119				62				46.59			
CCSpan_Memory (MB)	-				134				169			

(-) indicates unable to analyse

4.2 Evaluating LCCspm Algorithm (Memory-Optimised)

A hundred and twenty 18_{fccp} were discovered by both LCCspm and CCSpan algorithms. CCSpan used 169 megabytes of computer memory and spent 650.23 seconds to discover those 18_{fccp} . Besides the discovery of sixty-nine 3_{fccp} at 75% with a shorter runtime and lesser compute, LCCspm also discovered those one hundred and thirty-eight 9_{fccp} at 32.98 seconds of execution runtime and 46.59 megabytes of computer memory.

Across the results per relative support thresholds, the length of discovered frequent patterns is seen to reduce as the value for support increases for both LCCspm and CCSpan. At 25% relative support, the maximum length of an extracted frequent closed contiguous match event patterns contained forty match events. When the relative support was set to 50%, the maximum length of an extracted frequent closed contiguous match event patterns contained twenty-nine match events. Last, at 75% relative support, the maximum length of an extracted frequent closed contiguous match event patterns contained nine match events.

After cross-examining the performance results of LCCspm parameters $l = 29$, $\sigma = 50\%$ with CCSpan, LCCspm is approximately ten times faster than CCSpan and it used about 46% of the total memory consumed by CCSpan. The evaluation of the performance of the algorithms at 75% relative support (with LCCspm l parameter set to 9) revealed that LCCspm is approximately 20 times faster than CCSpan while it used about 28% of the total memory consumed by CCSpan. Yet again, the results of lower iteration of LCCspm l values are consistently lower than its higher values (See Table 4.9).

Having cross-examined the outputted patterns from both algorithms, LCCspm and CCSpan produced the same patterns. Table 4.10 reveals the top 15 patterns found in the men's FIFA 2018 World Cup data based on parameters $l = 29$, $\sigma = 50\%$, its frequency and its decoded movements. The discovered top-15 match event patterns are two to four-length patterns. The match event pattern fcd is one of the many interesting 4-length frequent closed contiguous patterns that was discovered. It is decoded as Pressure, Pass, Ball Receipt*, Carry. fcd was performed in 60 of 64 matches of the 2018 World Cup tournament.

Two of the top 15 discovered patterns (i.e., dc and cc) that were performed in all matches World Cup (i.e., 64) are two-length patterns. Interestingly, longer match events were discovered by both algorithms. The bottom-15 patterns at 50% relative

4.3 Evaluating LCCspm Algorithm (Time-Optimised)

Table 4.10: Men’s FIFA 2018 World Cup Top-15 Patterns ($l = 29$, $\sigma = 50\%$) (Adeyemo et al., 2021)

Encoded	Support	Decoded
dc	64	Ball Receipt*, Pass
cc	64	Pass, Pass
fc	63	Pressure, Pass
ec	63	Carry, Pass
cde	63	Pass, Ball Receipt*, Carry
fcd	62	Pressure, Pass, Ball Receipt*
ecd	62	Carry, Pass, Ball Receipt*
dec	62	Ball Receipt*, Carry, Pass
dcd	62	Ball Receipt*, Pass, Ball Receipt*
ccd	62	Pass, Pass, Ball Receipt*
cdec	61	Pass, Ball Receipt*, Carry, Pass
fcde	60	Pressure, Pass, Ball Receipt*, Carry
ecde	60	Carry, Pass, Ball Receipt*, Carry
dcde	60	Ball Receipt*, Pass, Ball Receipt*, Carry
ccde	60	Pass, Pass, Ball Receipt*, Carry

support comprise two to twenty-one discovered match events (Table 4.11). The discovered match event pattern (containing the most number of match events) on the list is *ecdefcdecdecdecdecdec* (See Table 4.11 for decoded closed contiguous match events) which was performed in 32 of the 64 World Cup matches. Two short patterns were also on the list, namely: *co* (denoted as Pass, Foul Committed) and *fp* (denoted as Pressure, Foul Won). Both match events were also performed in 32 matches.

4.3 Evaluating LCCspm Algorithm (Time-Optimised)

This section contains the datasets, experiment settings, results and discussion of evaluating the LCCspm algorithm (time-optimised) algorithm.

4.3.1 Dataset and Experimental Settings

This comparative analysis considered the large sequential data of rugby league players’ movements (Section 4.2.1) to compare the performances of LCCspm (Memory-

4.3 Evaluating LCCspm Algorithm (Time-Optimised)

Table 4.11: Men’s FIFA 2018 World Cup Bottom-15 Patterns ($l = 29$, $\sigma = 50\%$) (Adeyemo, 2023)

Encoded	Support	Decoded
cdecdecdecdecdecdecfde	32	Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Pressure, Ball Receipt*, Carry
cdecdecdecfcdecdecdec	32	Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Pressure, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*
co	32	Pass, Foul Committed
decdecfcdecdecdecfcde	32	Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pressure, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pressure, Pass, Ball Receipt*, Carry,
cdecdcdecdecdecdec	32	Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*
pcdec	32	Foul Won, Pass, Ball Receipt*, Carry, Pass
dek	32	Ball Receipt*, Carry, Miscontrol
fcdecdecfcde	32	Pressure, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*,
ffdecdec	32	Pressure, Pressure, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass
defdecfd	32	Ball Receipt*, Carry, Pressure, Pass, Pressure, Ball Receipt*, Pass, Ball Receipt*
decdecfcde	32	Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Pass, Pressure, Ball Receipt*, Carry
cdecfcdecfcdec	32	Pass, Ball Receipt*, Carry, Pressure, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Pressure, Ball Receipt*, Carry, Pass
cdecfcdecfcfd	32	Pass, Ball Receipt*, Carry, Pass, Pressure, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Pressure, Ball Receipt*
ecdecfcdecdecdecdec	32	Carry, Pass, Ball Receipt*, Carry, Pressure, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass, Ball Receipt*, Carry, Pass
fp	32	Pressure, Foul Won

Optimised), CCSpan and LCCspm (Time-Optimised). The performance of the algorithms for frequent movement pattern extraction was based on 5%, 25%, 50% and 75% relative support thresholds.

4.3.2 Results and Discussion

The length of obtainable movement patterns varied from one-length movement patterns to maximal-length movement patterns. From Table 4.12, both memory and time-optimised LCCspm algorithms results are obtained following an incremental iteration (based on user-defined lengths) until the maximum l value per relative support threshold. The results of both memory-optimised LCCspm and CCSpan algorithms remain the same from Section 4.2.2.2. Once again, the CCSpan algorithm only produces a one-time result per relative support threshold. Meanwhile, both variants of the LCCspm algorithms provided results in each iteration.

At a 5% relative support threshold, the LCCspm (Time-Optimised) algorithm was able to identify the same sets of frequent movement patterns as the LCCspm (Memory-Optimised) variant, per varied l value. However, the memory consumed and execution runtime for each iteration varied. For example, it took LCCspm (Memory-Optimised)

4.3 Evaluating LCCspm Algorithm (Time-Optimised)

12.52 seconds of execution runtime to discover the 152 $2_{f_{ccp}}$ movement patterns while LCCspm (Time-Optimised) took only 7.86 seconds. LCCspm (Memory-Optimised) discovered the 783 frequent $7_{f_{ccp}}$ movement patterns following 5,399 seconds of execution runtime while LCCspm (Time-Optimised) variant discovered the same set of frequent movement patterns following 52.32 seconds. The LCCspm (Time-Optimised) algorithm consumed more memory than the LCCspm (Memory-Optimised) algorithm, e.g. LCCspm (Time-Optimised) consumed 1998 megabytes of computer memory to discover frequent $7_{f_{ccp}}$ movement patterns while the LCCspm (Memory-Optimised) used 161 megabytes of computer memory.

At 25% relative support threshold, the LCCspm (Time-Optimised) algorithm identified the same sets of frequent movement patterns as the LCCspm (Memory-Optimised), per varied l value. However, the memory consumed and execution runtime for each iteration varied. For example, it took LCCspm (Memory-Optimised) 61.26 seconds of execution runtime to discover the 72 $3_{f_{ccp}}$ movement patterns while LCCspm (Time-Optimised) took only 12.54 seconds.

Also, LCCspm (Memory-Optimised) discovered the 102 frequent $5_{f_{ccp}}$ movement patterns following 852.28 seconds of execution runtime while its time-optimised variant discovered the same set of frequent movement patterns following only 29.04 seconds. However, the LCCspm (Time-Optimised) consumed more memory than the LCCspm (Memory-Optimised) algorithm, e.g. LCCspm (Time-Optimised) consumed 1277.1 megabytes of computer memory to discover frequent $5_{f_{ccp}}$ movement patterns while the memory-optimised approach used only 62 megabytes of computer memory.

Similarly, when the relative support threshold was set to 50%, the optimised variant of the LCCspm algorithm took shorter execution runtime while consuming more computer memory than the naive variant of the LCCspm algorithm. The optimised variant of LCCspm algorithm was able to find all frequent closed contiguous movement patterns following a single scan of the given set of sequences. It finds all those patterns quicker than the naive LCCspm approach. This was possible through the implementation of a set of 2-tuples that stored generated candidate pattern and its index pair as a form of vertical representation. This enables the quick count of candidate patterns' support and test for frequency, unlike the naive LCCspm approach.

4.3 Evaluating LCCspm Algorithm (Time-Optimised)

Table 4.12: LCCspm Algorithm (Time-Optimised) Performance Evaluation Results on Large Player Movement Sequential Data (Adeyemo, 2023)

Support	5%							25%					50%				75%	
length	1	2	3	4	5	6	7	1	2	3	4	5	1	2	3	4	1	2
LCCspm Pattern Count	34	152	393	542	642	737	783	17	50	72	93	102	12	23	30	32	7	10
CCSpan Pattern Count	-							102					32				10	
LCCspm(Opt.) Pattern Count		152	393	542	642	767	783	17	50	72	93	102	12	23	30	32	7	10
LCCSpm Runtime (secs)	2.10	12.52	71.65	420.72	983.56	2465.14	5399.00	1.30	7.73	61.26	252.33	852.28	0.80	5.13	42.22	258.70	0.58	5.48
CCSpan Runtime (secs)	-							2902.387					544.31				38.376	
LCCSpm(Opt.) Runtime (secs)	3.17	7.86	14.39	20.69	28.9	41.45	52.32	2.24	6.41	12.54	19.67	29.04	2.32	6.47	11.97	19.42	3.67	7.73
LCCSpm Memory (MB)	36.5	37.2	38.4	42.2	58.8	95	161.5	36.5	36.7	37.7	41.7	62	35.5	35.7	37.6	40.8	35.5	35.7
CCSpan Memory (MB)	-							96					91				82	
LCCSpm(Opt.) Memory (MB)	57.5	255.3	549.1	928.5	1251.9	1603.1	1998.1	57.5	255.9	548.4	898.4	1277.1	57.5	255.9	548.4	898.2	57.3	255.3

(-) indicates unable to analyse

However, it is limited because all possible candidate patterns must be generated to enumerate all indexes for support counting. Overall, the LCCspm (Time-Optimised) consumed more memory than the naive LCCspm because all possible candidate patterns were generated via snippet growth from a given set of sequences, unlike the naive LCCspm approach that limited candidate patterns generation to those that are potentially frequent.

4.4 Chapter Summary

This chapter proposed and implemented a novel frequent closed contiguous sequential pattern mining algorithm, LCCspm. LCCspm satisfied all four criteria for extracting patterns for player movement profiling. LCCspm algorithm uses a dictionary data structure for storing extracted patterns and support pairs. It uses a snippet-growth approach for candidate pattern generation. Also, it finds closed contiguous patterns from frequent patterns by using novel *inverse characteristic* properties of closed contiguous patterns.

Due to the potential limitation of LCCspm (Memory-Optimised), in terms of how it counts support for each candidate pattern, a time-optimised LCCspm uses a vertical representation of candidate patterns by pairing each generated candidate with its index was developed. The unique indexes of each candidate pattern when grouped enable quick computation of its support or frequency. The optimised version of LCCspm also uses a set of 2-tuple data structure to store extracted patterns and support pairs.

This chapter also evaluates the performance of the proposed and implemented LCCspm algorithms, through comparative analysis, with the existing state-of-the-art algorithm for extracting closed contiguous patterns (i.e., CCSpan) and between themselves. Both algorithms were applied to two real-life use cases. The results show that LCCspm (Memory-Optimised) not only provides results in shorter runtime but with a lower compute as opposed CCSpan. More so, the empirical results further showcased the scalability prowess of LCCspm (Memory-Optimised) in both small and large sets of sequence sizes. CCSpan was unable to analyse rugby league players' large sequential data (i.e. five matches) using a low-value (i.e. 5%) support threshold.

The LCCspm (Time-Optimised) algorithm was able to identify user-defined lengths of frequent closed contiguous patterns faster than the memory-optimised approach.

The LCCspm (Memory-Optimised) algorithm scans a given set of sequences first for candidate pattern generation and many times for counting support of each candidate pattern. The LCCspm (Time-Optimised) algorithm scans a given set of sequences only once to extract all frequent user-defined lengths of closed contiguous patterns. It outperformed the memory-optimised approach of LCCspm (in terms of execution runtime) by two orders of magnitude when low relative support and large user-defined length parameters were set. However, it consumes more computer memory than the naive approach.

Overall, LCCspm algorithms aimed to provide fast, scalable and reliable algorithms to discover frequent player movement patterns that can further be utilised to identify the unique/positional player movements, compare the teams on a competition level or analyse the demand of matches. Also, this chapter evaluates the performance of both variants of the LCCspm algorithms and the results showcased the applicability and scalability of both algorithms. In the next chapter, the LCCspm algorithm will be validated for player movement profiling against the LCS algorithm within the SMP framework (i.e. currently applied method in sports) as well as the AprioriClose algorithm to investigate if the consecutive aspect of extracted movement pattern for player movement profiling matters.

Comparison of LCCspm with Existing Player Movement Profiling Frameworks

The previous chapter presented and evaluated the performance of the novel LCCspm algorithm. This chapter focuses on the comparison of different types of movement patterns and validates the best for player position profiling, based on the rationale presented in Section 3.2. This chapter fulfils the third objective of this PhD study. The results of this chapter are submitted for publication in the Plos One journal.

5.1 Experimental Method

The experimental framework depicted in Figure 5.1 illustrates an overview of data collection, processing and analysis methods. The parameter settings of the selected algorithms are presented and discussed in Section 5.1.2. The three-step method for comparing the algorithm towards identifying and validating the best type of movement patterns is presented and discussed in Section 5.1.3.

5.1.1 Data and Processing

An observational repeated measures design was used in which GPS data from 50 elite male rugby league players who participated in 319 fixtures within the 2019 and 2020 seasons were considered. Two rugby league playing positions were selected hook-

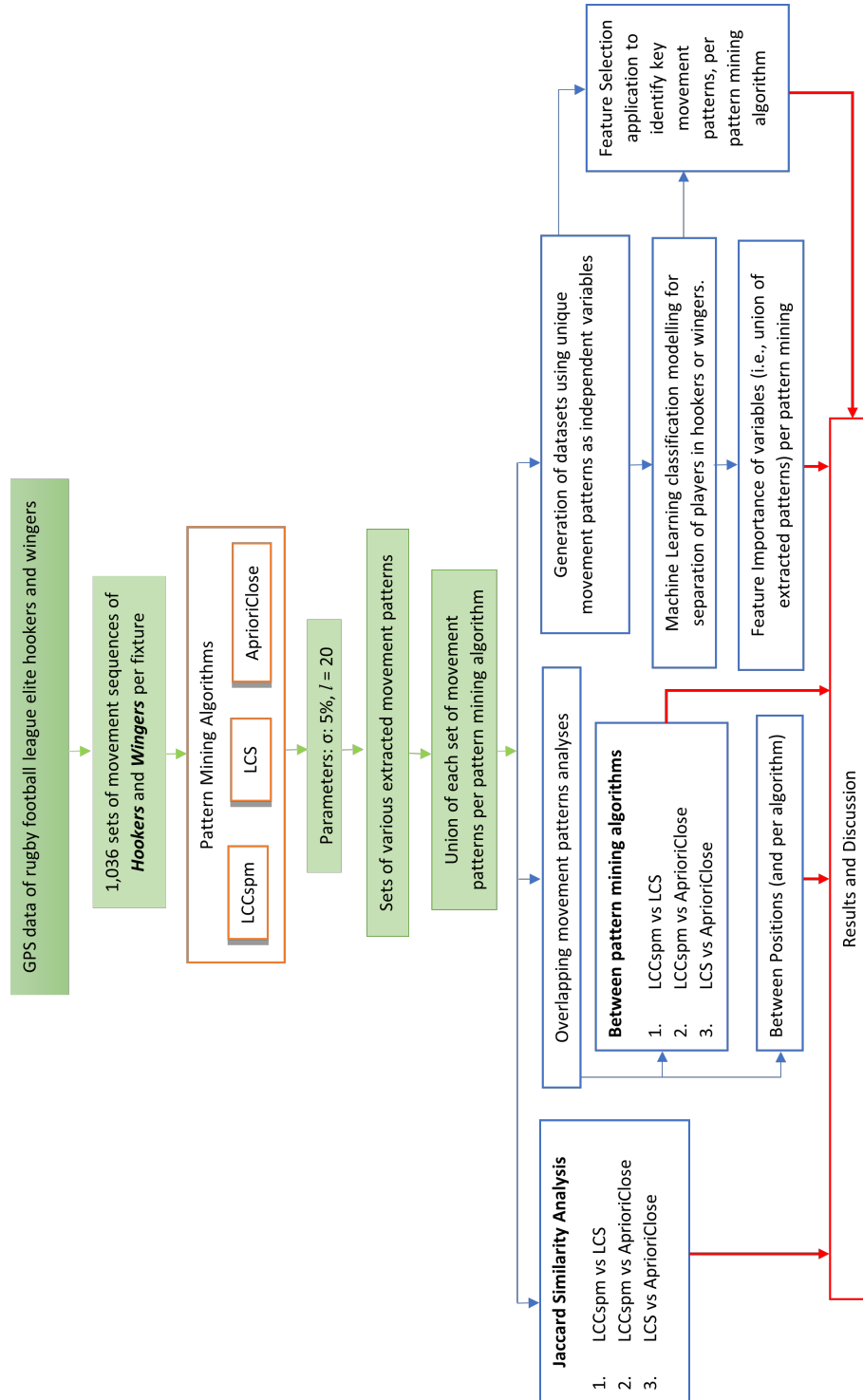


Figure 5.1: Workflow for Identifying the best sets of Movement Patterns (Adeyemo et al., 2023)

ers ($n = 22$) and wingers ($n = 28$). The method for generating movement sequences from global positioning systems data as published by [White et al. \(2021\)](#) and discussed in Section 3.1.2 was followed to obtain sets of movement sequences. The data of every player were extracted separately per match. We refer to this data granularity as “player-per-fixture level”. A total of one thousand and thirty-six GPS data at the player-per-fixture level was processed into sets of discrete movement sequences. Movement patterns were extracted from each set of discrete movement sequences and represented the frequent recurring movement patterns profiled for each player within a match. The average number of sequences in each set of movement sequences is approximately seven hundred and seventeen (i.e., 717.23). The sum of all discrete movement sequences across all sets of movement sequences equals seven hundred and forty-three thousand and fifty (i.e., 743,050).

5.1.2 Pattern Mining Algorithms and Parameter Settings

Applying these pattern mining algorithms (i.e., “LCCspm”, “LCS”, and “Apriori-Close”) on the same sets of sequences to extract different sets of movement patterns is the first step of the solution towards addressing the problem of identifying which pattern mining algorithm discovers the best type of movement patterns to separate between groups of rugby league players. This step assists in identifying the various sets of unique movement patterns (per pattern mining algorithm) that can be used as predictor variables towards developing input data for classification modelling.

LCCspm was used to extract sequential and consecutive movement patterns. LCS algorithm of the existing “SMP” framework ([White et al., 2021](#)) for player movement profiling, was used to extract sequential but non-consecutive movement patterns. A-close or AprioriClose [Pasquier et al. \(1999\)](#) being a pattern mining algorithm for extracting frequent closed itemset for association rules mining, was used to extract non-sequential movement patterns. LCCspm has two parameters: support (determines patterns’ frequency) and length (determines patterns’ maximal length). Apriori-Close has one parameter: support (determines patterns’ frequency) LCS has no parameter. The support parameter of both the LCCspm and Apriori close algorithms was set to 5% to extract a large number of frequent movement patterns because a high support threshold will identify few frequent patterns ([Fournier-Viger et al., 2017](#); [Mabroukeh](#)

and Ezeife, 2010) from the player's discrete movement sequences (i.e., active periods within a match). Additionally, 5% support was set because movement patterns with support higher than 5% are extractable from the results obtained using 5% support and made use of during analysis. However, high support value will identify few frequent movement patterns but may discard other interesting or contributing lower-frequency movement patterns. LCCspm length parameter was set to 20 (i.e. 2-second time frame of rugby league period of play) to ensure more and longer patterns are extracted although the maximum length of identified frequent closed contiguous movement patterns (Chapter 4) is 7. Meanwhile, the movement patterns extracted by AprioriClose and LCS algorithms were later filtered to exclude patterns containing more than 20 movement units. More so, the support and length parameter values are widely used parameters (Abboud et al., 2017; Bunker et al., 2021; Mabroukeh and Ezeife, 2010) for extracting large and longer length frequent patterns. Additionally, the unique movement patterns extracted by each algorithm were identified by computing the union of all extracted movement patterns. The unique movement patterns (per algorithm) were subjected to further analysis (discussed in the section below) to identify and validate the best pattern-mining algorithm for player movement profiling into playing positions.

5.1.3 Movement Pattern Validation

Three steps were taken to identify and thus validate the best pattern mining algorithm to extract movement patterns for profiling rugby league players into playing positions. First, similarity analysis of the different sets of unique movement patterns obtained per pattern mining algorithm was considered. The analysis of overlap movement patterns between pattern mining algorithms, positions and positions per algorithm was also carried out. Lastly, the separation of hookers and wingers into playing positions based on the different sets of extracted movement patterns was conducted and measured.

5.1.3.1 Jaccard Analysis

Jaccard similarity measure (Wang et al., 2019) enables exact matching of patterns between two sets and was used to quantify the similarity among the groups of extracted patterns. Given two sets of movement patterns (i.e., A and B), the Jaccard similarity

index is computed as in Equation 5.1.

$$J(A, B) = |A \cap B| / |A \cup B| \quad (5.1)$$

5.1.3.2 Overlap Movement Patterns

Overlap movement patterns between two sets of movement patterns were identified using the exact matching method. Overlapping unique movement patterns between pairs of pattern mining algorithms were identified. For each pair, the most frequent-50 and least frequent-50 extracted movement patterns of each pattern mining algorithm were checked for overlapping and visualised. This was carried out to identify where the overlapping movement patterns are located.

A further analysis was carried out on the identified overlapped movement patterns to discover those patterns performed by players of each playing position. Also, the overlapped movement patterns between playing positions per pattern mining algorithm were explored.

5.1.3.3 Separation of Players into Hookers and Wingers

This is the third step to validate which pattern mining algorithm produced the best set of movement patterns to separate RFL hookers and wingers. This step will assess and measure the extent to which each set of unique movement patterns (per pattern mining algorithm) can separate between rugby league hookers and wingers. The separation of players into playing positions was achieved through Machine learning classification modelling. The identification of the best set of movement patterns to separate RFL Super League hookers and wingers is mainly based on the separation (i.e., classification) accuracy.

Datasets for classification modelling were generated, where the unique set of movement patterns derived from the profiled movement patterns extracted per pattern mining algorithm were the predictor variables. The values of each observation are either 1 if players performed the unique movement patterns within fixtures or 0 if otherwise. The values of the dependent variable are either hooker or winger depending on the players' playing position. Five machine-learning classification algorithms (discussed in Section 3.2.2.1) representing five different types of learning techniques were se-

Table 5.1: Classification Algorithms Parameter settings (Adeyemo et al., 2023)

Algorithms	Parameters
Decision Tree	<i>default</i>
Gaussian Naïve Bayes	<i>default</i>
Random Forest	random_state = 1; others are <i>default</i>
Logistic Regression	penalty = “l1”, solver = “liblinear”
MLP	max_iter = 300, random_state = 5

lected for classification modelling. The classification algorithms were implemented in the scikit-learn (version 1.1.3) python module (Pedregosa et al., 2011). The parameters for fitting each classification algorithm are presented in Table 5.1.

The classification models were fitted via cross-validation technique (discussed in Section 3.3.2.3) with the *n_splits* parameter set to 10, *random_state* parameter set to 10 and the *shuffle* parameter set to “True”. The cross-validated models’ performances were evaluated (discussed in Section 3.3.2.3) through aggregated accuracy, precision, recall and f1-score metrics respectively. The performance scores of each classification model were collected and presented.

5.1.3.4 Significant Movement Patterns for Hookers and Wingers Separation

The identification of significant predictor variables for classification modelling is crucial (as discussed in Section 2.2.1). It will assist in the identification of important movement patterns for separating rugby league players into hookers and wingers. At the completion of classification modelling and performance evaluations, the classifier with the most accurate models across the three types of extracted movement patterns was further analysed for feature importance (discussed in Section 3.3.2.4) of movement patterns per pattern mining framework. The feature importance score of each movement pattern per cross-validated model was collected and aggregated to identify the important movement patterns. This was done to identify top 20 important movement patterns (per pattern mining algorithm) used for classification model development.

The dataset wherewith classification algorithms fitted their most accurate models was further analysed to identify key movement patterns necessary for classification modelling. The filter feature subset-selection methods (discussed in Section 3.3.2.4) were applied to identify key movement patterns as predictor variables for re-

classification modelling. To compare both filter feature subset-selection and search methods, Correlation-based Subset Evaluator (discussed in Section 3.3.2.4) and Consistency based Subset Evaluator methods (discussed in Section 3.3.2.4) were utilised with both best first and genetic search methods. New data subsets were generated based on the identified key movement patterns and used for classification re-modelling. The performances of the classification models were re-evaluated.

The source code for computing the Jaccard similarity of unique movement patterns, visualization of overlapped frequent patterns between the pattern mining algorithms and playing positions, machine learning classification model development and evaluation, and feature importance scores analyses are all available publicly and online in a GitHub repository ([Adeyemo, 2021b](#)).

5.2 Results

The results of the validation analysis are presented in a step-wise manner. First, the results of the extraction of movement patterns based on the three-pattern mining framework as well as the similarity check and identification of overlapping patterns were presented. The extent of players' separation into groups based on each type of extracted movement pattern was measured and presented. The results of top movement pattern importance scores (per pattern mining framework) were uncovered and presented. The identification of key movement patterns and classification remodelling results are also presented.

5.2.1 Jaccard Analysis

All one thousand and thirty-six sets of movement sequences contained a total of seven hundred and forty-three thousand and fifty (i.e., 743,050) movement sequences. LCC-spm algorithm extracted a unique set of three thousand eight hundred and eighty-one (i.e., 3881) frequent closed contiguous movement patterns. The LCS algorithm (of the "SMP" framework) extracted a unique set of two thousand five hundred and thirteen (i.e., 2513) frequent longest common movement patterns. The AprioriClose algorithm extracted a unique set of one hundred and fifty-five (i.e., 155) frequent closed (association rules) movement patterns.

Table 5.2: Similarity of Movement Patterns per Algorithm (Adeyemo et al., 2023)

Jaccard Similarity Score			
Algorithms	LCCspm	LCS	AprioriClose
LCC	1.0	0.19	0.008
SMP	0.19	1.0	0.009
APR	0.008	0.009	1.0

Table 5.2 reports the results of Jaccard similarity analysis to quantify the similarity in the extracted unique sets of movement patterns between pattern mining algorithms. Overall, Jaccard scores ranged from 0.008 to 0.19 suggesting limited similarity among the movement patterns extracted by the three algorithms.

5.2.2 Overlapping Movement Patterns

5.2.2.1 Overlapping Movement Patterns Between Algorithms

LCCspm vs. LCS

One thousand and twenty-two (1022) unique movement patterns overlapped between LCCspm (26% of total) and LCS (40% of total) algorithms. In the most frequent-50 extracted movement patterns for each pattern mining algorithm, thirty-two movement patterns overlapped between LCCspm and LCS algorithms (64%) with no overlap in the least-50 frequent movement patterns. Figure 5.2 highlights the visualisation of the overlapped movement patterns based on the frequency count of LCCspm algorithm between LCCspm and LCS algorithms.

These overlapped patterns are tabulated in Table 5.3 alongside the corresponding overall absolute support values (i.e., for all sets of movement sequences) of both pattern mining frameworks respectively. The overall absolute support was computed by adding all the resulting absolute support of each frequent movement pattern as extracted per pattern mining framework across the one thousand and thirty-six sets of movement sequences. From Table 5.3, the patterns “uv” denoted (using Table 3.2) as jog acceleration straight and jog acceleration acute-change (with an LCCspm absolute support of 224,742 and LCS absolute support of 161,353), “uuv” denoted as

Table 5.3: Overlapped patterns and support values between Top-50 LCCspm and LCS Frameworks (Adeyemo, 2023)

Sequences	LCCspm Absolute Support	LCS Absolute Support
uv	224,742	161,353
vu	223,879	170,751
ji	221,307	5,572
uu	212,727	60,562
ef	209,012	81,361
fe	204,828	32,067
vv	203,470	23,461
ee	197,139	32,695
ff	195,164	22,691
uuv	191,260	143,918
vuu	190,010	72,909
uvv	175,771	29,161
vvu	172,223	40,578
uuu	171,704	33,777
uvu	165,958	7,504
ab	152,905	6,477
vuuu	148,755	37,045
uuuv	148,571	34,068
vuv	145,079	11,625
mnn	132,013	11,543
uuuu	130,326	19,256
eef	129,526	31,236
uuvu	126,671	17,182
uvuu	125,477	19,709
fee	123,648	6,503
uuvv	122,463	24,942
vvuu	122,127	10,303
eee	121,348	13,184
ffe	113,588	16,971
efe	111,994	16,424
uvvu	109,374	63,49
vuuuu	108,694	17,293

jog acceleration with acute-change of direction is the most frequent. The movement pattern “ij” (walk acceleration straight and walk acceleration with acute-change of

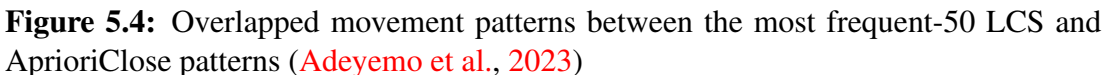
direction) had a “LCCspm” absolute support of 219,190 and “AprioriClose” absolute support of 138,576. The movement pattern “ef” (walk neutral straight and walk neutral

Figure 5.3: Overlapped movement patterns between the most frequent-50 AprioriClose and LCCspm patterns (Adeyemo et al., 2023)

acute-change) had a “LCCspm” absolute support of 209,012 and “AprioriClose” absolute support of 28,690. The results of the overall absolute support of the overlapped movement patterns between the “LCCspm” and “AprioriClose” movement patterns indicated that those top overlapped movement patterns are more identified as closed contiguous movement patterns than closed association rules movement patterns.

LCS vs. AprioriClose

Twenty-five movement patterns overlapped between the LCS (1% of total) and AprioriClose (16.13% of total) algorithms. In the most frequent-50 extracted movement patterns for each pattern mining algorithm, 3 movement patterns overlapped between LCS and AprioriClose algorithms while there were no overlapped patterns between the least 50 frequent movement patterns. Fig 5.4 highlights the visualisation of the overlapped movement patterns based on the frequency count of LCS algorithm. The movement pattern “ef” (walk neutral straight and walk neutral with acute-change of direction) was the second most frequent overlapping pattern (Fig 5.4). The movement pattern “uv” denoted (using Table 3.2) as jog acceleration straight and jog acceleration acute-change is the most frequent overlap movement pattern (with a “LCS” absolute support of 161,353 and “AprioriClose” absolute support of 119,091). The movement pattern “ef” denoted as walk neutral straight and walk neutral acute-change is the second most frequent overlap pattern with a “LCS” absolute support of 81,361 and “AprioriClose” absolute support of 28,690. Movement pattern “a” denoted as walk acceleration straight and walk acceleration acute-change (with a “LCS” absolute support



5.2.2.2 Overlapped Frequent-50 Movement Patterns between Positions

(a) Hooks

(b) Wingers

The movement patterns “ji” denoted as walk acceleration acute-change and walk

acceleration straight and “fee” denoted as walk neutral acute-change and [walk neutral straight] x 2 were mainly performed by wingers while movement patterns “uuuuu” denoted as [jog acceleration straight] x 4 and jog acceleration acute-change, and “mn” jog deceleration straight and jog deceleration acute-change were mainly performed by hookers among other overlapped movement patterns.

All overlapped movement patterns (“ef, uv and ij”) between the most frequent-50 frequent LCCspm and AprioriClose patterns (Fig 5.3) were performed by hookers and wingers. Similarly, both hookers and wingers performed the overlapped movement patterns (“ef, uv and a”) between the most frequent-50 frequent movement patterns extracted by LCS and AprioriClose algorithms (Fig 5.4).

5.2.2.3 Overlapped Movement Patterns between Positions per Algorithm

LCCspm

Two thousand two hundred and eighty-two (2,282) and three thousand one hundred and seventy-four (3,174) sets of frequent closed contiguous movement patterns were identified by LCCspm to profile hookers and wingers respectively. A total of one hundred five hundred and seventy-five (1,575) movement patterns overlapped between both playing positions (visualized in Figure 5.6) as extracted by LCCspm algorithm. Also, LCCspm profiled seven hundred and seven (707) closed contiguous movement



(a) Based on Hookers' Support Frequency



(b) Based on Wingers' Support Frequency

Figure 5.6: Overlapped movement patterns within positions as extracted by LCCspm (Adeyemo et al., 2023)

patterns uniquely performed by hookers and another set of one thousand five hundred and ninety-nine (1599) movement patterns performed only by wingers.

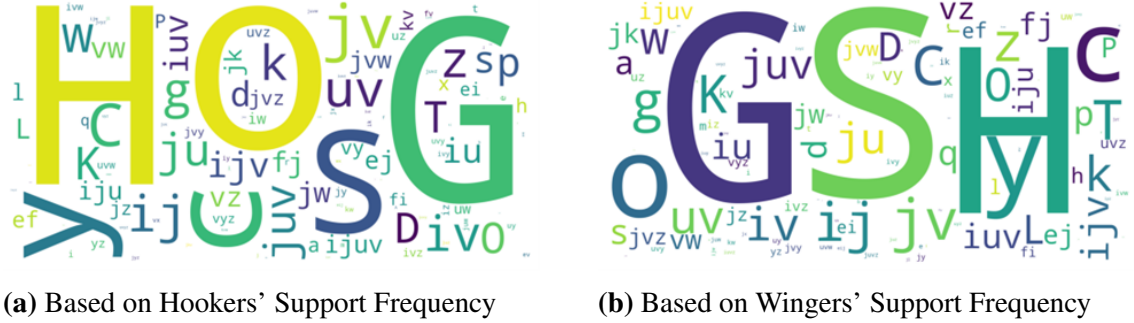


Figure 5.8: Overlapped movement patterns within positions as extracted by Apriori-Close (Adeyemo et al., 2023)

5.2.3 Separation of Players into Hookers and Wingers

Rugby league players were classified into playing positions based on the set of unique movement patterns across playing positions. The classification results for each set of movement patterns were presented and the classification model with the most accurate classification (across pattern mining algorithms) was further analysed for feature importance. Furthermore, the results of identifying the key movement patterns from the set of movement patterns with the best classification results are also presented.

5.2.3.1 Separation Analysis

Three datasets were generated for classification modelling. The first dataset (representing LCCspm algorithm) contained 3,881 predictor variables. The second dataset (representing LCS algorithm) contained 2,849 predictor variables. The third dataset (representing AprioriClose algorithm) contained 155 predictor variables. The distribution of the target variables values labels were five hundred and thirty-six (i.e., 536) *wingers* and five hundred (i.e., 500) *hookers*. The datasets are fairly balanced, in terms of the number of observations per target variable value, for classification modelling. The accuracy of the selected five (5) machine learning classification algorithms after modelling on all three datasets are reported in Table 5.4.

A reduction in performances of all classification models on the “LCS” movement patterns dataset was recorded when compared with their performances on the “LCCspm” movement patterns dataset. The Decision tree classification model on the “LCS” movement patterns dataset had a 57.72% accuracy, f1-score of 0.57 and 0.58 precision

Table 5.4: Classifiers’ Separation Accuracies using various sets of Movement Sequences (Adeyemo et al., 2023)

Classifier	Accuracy (%)	Precision	Recall	F1-Score	Framework
Decision Tree	82.83	0.83	0.83	0.83	LCC
Gaussian Naïve Bayes	58.09	0.73	0.58	0.5	
Random Forest	88.61	0.89	0.89	0.89	
Logistic Regression	89.77	0.9	0.9	0.09	
MLP	91.02	0.91	0.91	0.91	
Decision Tree	57.72	0.58	0.58	0.57	SMP
Gaussian Naïve Bayes	50	0.51	0.51	0.47	
Random Forest	63.8	0.64	0.64	0.64	
Logistic Regression	65.83	0.66	0.66	0.66	
MLP	61.78	0.62	0.62	0.62	
Decision Tree	73.56	0.74	0.74	0.73	APR
Gaussian Naïve Bayes	54.36	0.6	0.53	0.43	
Random Forest	81.76	0.82	0.82	0.82	
Logistic Regression	82.05	0.82	0.82	0.82	
MLP	80.90	0.81	0.81	0.81	

and recall scores. This is more than a 20% decrease in classification accuracy when compared to Decision tree performance on the “LCCspm” movement patterns dataset despite “LCCspm” and “LCS” movement patterns being a consecutive type of movement pattern. The best-performing classifier on “LCS” movement patterns dataset, the Logistic Regression model, had 65.83% accuracy and 0.66 f1-score, precision and recall scores respectively.

All classification models fitted on the “AprioriClose” movement patterns dataset performed better than those fitted on the “LCS” movement patterns dataset despite having a lower number of independent variables. The Decision tree classification model on the “AprioriClose” movement patterns dataset had 73.56% accuracy, f1-score of 0.73 and 0.74 precision and recall scores respectively. This is approximately 18% increase in classification accuracy when compared to Decision tree performance on the “LCS” movement patterns tabular dataset but approximately 10% lowered accuracy in comparison to its performance on “LCCspm” movement patterns tabular dataset. Logistic Regression model is again the best-performing classifier on “AprioriClose” movement patterns dataset with 82.05% accuracy and 0.82 f1-score, precision and recall scores respectively.

All classifiers fitted on the LCCspm dataset achieved the highest accuracies when compared to their counterparts fitted on the LCS and AprioriClose datasets. The MLP classifier fitted on the dataset having LCCspm movement patterns as its independent variables had the highest individual accuracy of 91.02% among all other classifiers fitted on any of the three datasets. MLP classifier achieved 61.78% and 80.9% accuracies on the LCS and AprioriClose datasets respectively. Meanwhile, the accuracy of the Gaussian Naive Bayes classifiers is the lowest among other classification algorithms, across all algorithms.

Consequently, the LCCspm algorithm used for mining closed contiguous movement patterns provided the most separating players into playing positions based on the classification models’ performances. The AprioriClose algorithm used for mining closed itemsets movement patterns provided the second-best data-driven insights (among three selected pattern mining algorithms) to separate players into playing positions. Meanwhile, the LCS algorithm of the “SMP” framework ranked provided the least data-driven insights to separate players into playing positions.

5.2.3.2 Significant Movement Patterns for the Separation

Feature Importance

From Table 5.4, the Logistic Regression algorithm fitted two of the three most accurate classification models per pattern mining algorithm. It fitted the most accurate classification models on the AprioriClose and LCS datasets, with accuracy of 82.95% and 65.83% respectively. Meanwhile, it fitted the second-best accurate classification model of 89.77% accuracy on the LCCspm dataset. As such, further analysis for the top-20 feature importance scores of the movement patterns used by the Logistic regression models per pattern mining algorithm was conducted and reported in Table 5.5.

Feature Selection

All classification models fitted their most accurate model on the data with closed contiguous movement patterns as predictor variables. Hence, key movement patterns within the data were identified using the filter feature subset-selection methods (discussed in Section 3.3.2.4). The Correlation-based feature subset using the best first search method identified 47 key closed contiguous movement patterns out of the original 3881 movement patterns for hookers and wingers. This subset of 47 key closed contiguous movement patterns had the highest score (0.381) out of all 196663 subsets. The Correlation-based feature subset using the genetic search method identified 987 key closed contiguous movement patterns from the original 3881 movement patterns. This subset of 987 key closed contiguous movement patterns had the highest score (0.0129) out of all 20 subsets.

The Consistency-based feature subset using the best first search method identified 28 key closed contiguous movement patterns out of the original 3881 movement patterns for hookers and wingers. This subset of 28 key closed contiguous movement patterns had the highest score (0.998) out of all 127554 subsets. The Consistency-based feature subset using the genetic search method identified 1414 key closed contiguous movement patterns out of the original 3881 movement patterns for hookers and wingers. This subset of 1414 key closed contiguous movement patterns had the highest score (0.99903) out of all 20 subsets.

Table 5.5: Logistic Regression Top 20 Important Patterns and Scores per Algorithm (Adeyemo et al., 2023)

(a) Top 20 APR Patterns Importance Scores		(b) Top 20 SMP Patterns Importance Scores		(c) Top 20 LCC Patterns Importance Scores	
APR Patterns	Importance Score	SMP Patterns	Importance Score	LCC Patterns	Importance Score
q	2.12	eeeeefeeeeeeee	1.95	iiii	2.02
G	2.03	ii	1.57	SS	1.67
ik	1.59	ff	1.41	iiie	1.40
P	1.44	eee	1.40	GGG	1.28
juv	1.24	ee	1.38	qrq	1.17
h	1.10	fee	1.28	iji	1.16
L	1.10	uuvvu	1.26	jjji	1.13
jvz	0.99	qqqqqq	1.15	ijjj	0.95
iuvy	0.90	vuvvu	1.11	HH	0.93
o	0.85	eeeeeeefee	1.10	mmr	0.91
iju	0.85	fe	1.07	iii	0.85
ijw	0.82	nnmn	1.00	uvvuuv	0.83
ijvz	0.81	ef	0.95	vuvvu	0.83
jk	0.76	j	0.94	zz	0.82
iuz	0.69	mmmmn	0.89	jie	0.81
uvz	0.55	GG	0.84	GGSS	0.79
juw	0.49	eeee	0.77	ie	0.78
juvw	0.49	i	0.76	ijf	0.77
ivz	0.48	uuvvvu	0.76	fe	0.73
jl	0.45	efeeeeeee	0.76	ijii	0.72

Table 5.6: Classification results on the Key Closed Contiguous Movement Patterns (Adeyemo, 2023)

Classifier	Accuracy(%)	Average Accuracy (%)	Precision	Recall	F1	FS_Method	No_Features
Decision Tree	82.83	82.06	0.82	0.82	0.82	-	3881
Gaussian Naïve Bayes	58.09		0.73	0.58	0.5		
Random Forest	88.61		0.89	0.89	0.89		
Logistic Regression	89.77		0.9	0.9	0.9		
MLP	91.02		0.91	0.91	0.91		
Decision Tree	84.74	85.62	0.85	0.85	0.85	Consistency Subset Evaluator using Best First Search	28
Gaussian Naïve Bayes	78		0.82	0.78	0.77		
Random Forest	88.61		0.89	0.89	0.89		
Logistic Regression	87.94		0.88	0.88	0.88		
MLP	88.8		0.89	0.89	0.89		
Decision Tree	80.51	79.98	0.81	0.81	0.8	Consistency Subset Evaluator using Genetic Search	1414
Gaussian Naïve Bayes	55.87		0.71	0.56	0.46		
Random Forest	88.22		0.88	0.88	0.88		
Logistic Regression	88.32		0.88	0.88	0.88		
MLP	86.97		0.87	0.87	0.87		
Decision Tree	80.79	85.83	0.81	0.81	0.81	Correlation-based feature subset selection using Best First Search Method	47
Gaussian Naïve Bayes	84.26		0.85	0.84	0.84		
Random Forest	87.93		0.88	0.88	0.88		
Logistic Regression	88.9		0.89	0.89	0.89		
MLP	87.26		0.87	0.87	0.87		
Decision Tree	81.08	79.82	0.81	0.81	0.81	Correlation-based feature subset selection using Genetic Search Method	987
Gaussian Naïve Bayes	54.71		0.69	0.56	0.44		
Random Forest	88.03		0.88	0.88	0.88		
Logistic Regression	88.22		0.88	0.88	0.88		
MLP	87.06		0.87	0.87	0.87		

The MLP classifier fitted on the original closed contiguous dataset (3881 features) achieved the highest accuracy of 91.02% with precision, recall, and F1-scores of 0.91 (Table 5.6). However, the logistic regression classifier fitted on the dataset with 47 features (identified by the correlation-based feature subset selection using the best first search method) achieved an accuracy of 88.9% with F1-score, precision and recall scores of 0.89 (Table 5.6). The classifiers produced the most stable performance on the key 47 closed contiguous movement patterns by achieving an aggregated accuracy of 85.85% across classifiers (Table 5.6).

5.3 Discussion

This chapter's objective of identifying the best type of movement patterns for profiling player movement by comparing the proposed and developed algorithm with the existing player movement profiling framework and AprioriClose algorithm was completed. The classification results of this study revealed the extent of separating players into playing positions, based on each set of frequent movement patterns, extracted from the same sets of movement sequences, under the same parameter condition. Table 5.4 shows that the separation of elite rugby players into hooker and winger positions based on their frequent movement patterns is best done using their extracted closed contiguous movement patterns profiled by the LCCspm algorithm. The LCCspm closed contiguous movement pattern using the Multi-Layered Perceptron classifier performed best to classify hookers and wingers in professional rugby league, with an overall accuracy of 91.02%. The AprioriClose closed itemsets (non-consecutive) movement patterns offered a better separation accuracy than the longest common subsequence movement patterns of the LCS algorithm. AprioriClose movement patterns provided a decent separation (through Logistic Regression accuracy of 82.05%). Its lowered accuracy can be attributed to the nature of its movement patterns being non-consecutive, non-sequential and without repeated movement activity. Also, the results of this study indicate player movement profiling using LCCspm will discover more numbers of movement patterns for profiling players from the same sets of movement sequences than AprioriClose and LCS algorithms. This implies there are more discoverable consecutive movement patterns than non-consecutive and non-sequential movement patterns. More so, Jaccard similarity scores (1 being full similarity) ranged from

0.008 to 0.19 among movement patterns algorithms (Table 5.2), suggesting a lack of similarity in the extracted patterns overall. LCCspm and LCS sets of movement patterns shared higher similarity because both algorithms extract some form of sequential movement patterns as opposed to AprioriClose non-sequential patterns. Based on these results, the set of movement patterns extracted by LCCspm is identified and validated as the best for profiling movement patterns of rugby league players playing positions.

The application of LCCspm to profile the movement of hookers and wingers revealed wingers performed 892 movement patterns more than hookers. This suggests a more variable movement profile of wingers than hookers. There were overlapped movement patterns between hookers and wingers (Fig 5.6), but the frequency at which the movement patterns were performed differs by playing positions. Overlapped movement patterns with a combination of movement units u , v which indicates accelerated jogs with some acute change in direction or on straights were mostly performed by hookers (Fig 5.6a). Wingers on the other hand performed overlapped movement patterns that included accelerated walks with some acute direction changes as indicated by movement units j and i (Fig 5.6b).

This study, through the application of LCCspm algorithm, also identified groups of movement activities performed uniquely by hookers and wingers. For example, the sequential movement pattern GGGGGGGGGGGSSSSSSS (Run Acceleration Straight [x12] and Sprint Acceleration Straight [x7]) was performed by hookers only. Equally, only wingers completed the sequential movement pattern of “TSSTSTTSST” (Sprint Acceleration Acute-change, Sprint Acceleration Straight [x2], Sprint Acceleration Acute-change, Sprint Acceleration Straight, Sprint Acceleration Acute-change [x2], Sprint Acceleration Straight [x2], Sprint Acceleration Acute-change). It is well established that wingers complete greater high-speed ($>5\text{m.s}^{-1}$) activity during matches than hookers (wingers: 626m vs. hookers: 285m) (Dalton-Barron et al., 2020), although these differences are less pronounced with global acceleration-based measures (e.g., average acceleration over a period of time). These differences are likely due to the vastly different tactical roles of wingers (e.g., returning kicks in attack leading to open space to move at high speed) vs. hookers (e.g., repositioning behind the play the ball to distribute possession). Applying pattern mining algorithms to uncover the sequential nature of the occurrences of activity can enable the better capability to classify positional groups and aid in enhanced training specificity.

It is also noteworthy to point out that the most important variables used by the Logistic Regression classification model are mostly not part of the most frequent-50 overlapping patterns profiled by LCCspm. This indicates that the not-too-frequent movement patterns and those uniquely performed by players of each playing position provided insights into players' playing position separation. The twenty most important LCCspm movement patterns used for fitting the Logistic Regression classifier consist of 2 to 6-length on-field movement activities (Table 5.5c). The second most important variable "SS" (denoted as [sprint acceleration straight] x2) and ninth most important variable "HH" (denoted as [run acceleration acute-change] x2) discovered by the LCCspm pattern mining algorithm are the only patterns to include on-field activities "S" and "H" in its set of most important movement patterns across all three pattern mining algorithms. The nineteenth important movement pattern "fe" in Table 5.5c is the only pattern present in the most frequent LCCspm and LCS overlapped movement pattern in Fig 5.2). This further validates the LCCspm algorithm and closed contiguous movement patterns as the best pattern to profile rugby league players into playing positions.

There is a significant difference between the total number of key closed contiguous movement patterns and the original closed contiguous movement patterns used at predictor variables (Table 5.6). The correlation-based feature subset selection using the best first search method identified forty-seven key movement patterns. This is approximately 1.2% of the original movement patterns in the dataset. Additionally, all classifiers produced stable performances when fitted on the dataset with forty-seven key movement patterns where it achieved a 3.84% increased accuracy compared to the aggregated performance of classifiers on the 3881 closed contiguous movement patterns. This establishes the usefulness of the correlation-based feature subset selection technique (using the best first search method) as the feature selection algorithm for finding key movement patterns.

5.4 Chapter Summary

The previous chapter evaluated the computational performance of LCCspm for pattern extraction from team sports. This chapter presents experiments that validated the effectiveness of LCCspm as an algorithm for player movement profiling and that movement

patterns produced by LCCspm can be integrated into a framework for player position separation. Through this chapter, this PhD study validates pattern mining algorithms for movement pattern extraction and solved the problem of identifying the set of movement patterns that best separates rugby league players into playing positions.

LCCspm and LCS algorithms extracted movement patterns that shared some form of similarity while AprioriClose movement patterns shared no similarity because itemsets do not consider the order of item appearance. Experiments on the use of pattern mining algorithms to extract movement patterns for the separation of players into positions show that closed itemset movement patterns are better than the longest common movement patterns, despite the large numbers of longest common movement patterns extracted by LCS algorithm of the “SMP” framework and being a consecutive movement pattern. Importantly, a set of closed contiguous movement patterns is best for separating players into positions because all classification models were most accurate on the dataset generated with LCCspm unique movement patterns as predictor variables. Therefore, mining closed contiguous movement patterns for profiling the on-field activities of players is recommended.

The identification of key movement patterns revealed that the correlation-based feature subset selection technique using the best first search method identified a few key movement patterns. These key movement patterns offered the highest aggregated classification accuracy across five classification algorithms with different learning schemes. Thus, the correlation-based feature subset selection technique using the best first search method is recommended for identifying key predictor variables without users’ and classification algorithms’ bias.

Although closed contiguous patterns are now validated as the best type of movement pattern to profile rugby league players into two distinct playing positions based on known differences, there are other seven rugby league playing positions besides hookers and wingers which should be explored. Additionally, the appropriate predictor variables values for such classification modelling should be investigated. In this chapter, only the presence or absence of the completed unique movement patterns per fixture was considered. Other predictor variable values (i.e. the count or frequency of completed unique movement patterns with respect to the total number of movement sequences) may offer different results. These identified limitations are addressed in the next chapter.

Applications of LCCspm Movement Patterns in Rugby Football League

The previous chapter validated LCCspm algorithm the algorithm that produced the best type of movement patterns for player movement profiling. This chapter presents three applications (rationale discussed in Section 3.3) of the validated algorithm (i.e., LCCspm) for player movement profiling in Sections 6.2.1, 6.2.2 and 6.2.3. This chapter fulfils the fourth objective of this PhD study. The results of this chapter are submitted to the 10th Workshop on Machine Learning and Data Mining for Sports Analytics, 2023.

6.1 Experimental Method

The experimental framework depicted in Figure 6.1 illustrates an overview of data collection, processing and analysis methods. This experimental method is broken down into three subsections, besides the data and processing section. The method to identify the signature movement patterns for each RFL playing position is discussed in Section 6.1.2. The method to classify elite rugby football league players into nine player positions is presented and discussed in Section 6.1.3 while the method for identifying the key movement patterns for the same classification is presented and discussed in subsection 6.1.3.1. The method for assessing players' performance variability using the sets of key movement patterns is presented and discussed in Section 6.1.4.

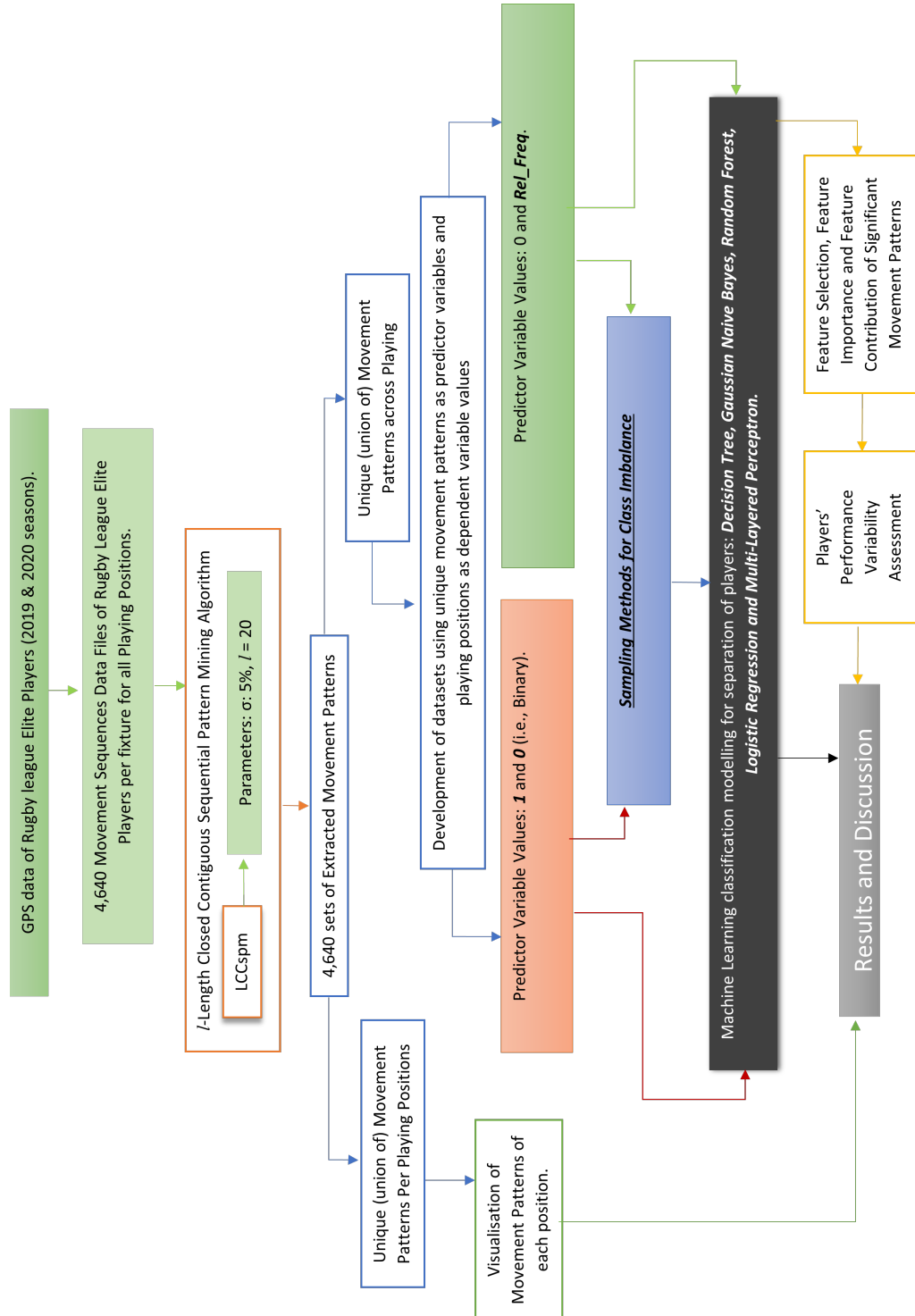


Figure 6.1: Workflow for RFL Players' Position Profiling, Classification and Performance Variability Assessment (Adeyemo, 2023)

6.1.1 Data and Processing

An observational repeated measures design was used in which 10Hz GPS data from two hundred and seventeen (217) elite male Rugby Football League players that played in fixed playing positions within the 2019 and 2020 seasons were collected via wearable sensors (Catapult S5, Catapult Innovations, Melbourne, Australia) worn during matches. The distribution of the players across playing positions was thirty-one (31) Centres, eight (8) Five-Eighths, twenty-two (22) Full-Backs, twenty-six (26) Half-Backs, twenty-two (22) Hookers, eight (8) Loose-Forwards, forty-seven (47) Prop-Forwards, twenty-five (25) Second-Rows and twenty-eight (28) wingers. Same as in Section 5.1.1, movement sequences were generated from rugby league players' GPS data using the method discussed in Section 3.1.2. The GPS data of two hundred and seventeen (217) elite male Rugby Football League players that participated in three hundred and thirty-eight fixtures were collected. See Table 6.1 for the distribution of players' sets of movement sequences per fixture per playing position. In total, all participants played in four hundred and fifty-four fixtures (match games) within the 2019 and 2020 seasons. The data of every player were extracted separately per match. We refer to this data granularity as "player-per-fixture level". A total of four thousand six hundred and forty GPS data at the player-per-fixture level was processed into sets of discrete movement sequences. Movement patterns were extracted from each set of discrete movement sequences and represented the frequent recurring movement patterns profiled for each player within a match.

6.1.2 Signature Movement Patterns of RFL players

LCCspm was applied to extract user-defined lengths of closed contiguous patterns from the movement sequences to discover patterns for each playing position and for classification modelling. LCCspm was used to extract movement patterns whose length does not exceed twenty (20) (i.e. 2-second time frame) and is at least five per cent (i.e., 5%) relatively frequent from every four thousand six hundred and forty sets of movement sequences. The unique set of movement patterns was derived from the extracted movement patterns by finding the union of all extracted movement patterns.

For each playing position, a union of extracted movement patterns performed by players within the playing position is identified. Afterwards, the unique set of move-

Table 6.1: Original and SMOTE Distribution of Players per Fixture per Playing Position (Adeyemo, 2023)

Playing Positions	Original observations	Percentage Increase (%)	SMOTE observations
Centres	779	-	779
Five-Eighths	188	275	705
Full-Backs	378	100	756
Half-Backs	619	15	711
Hookers	499	50	748
Loose-Forwards	230	220	736
Prop-Forwards	898	-	898
Secondrows	513	50	769
Wingers	536	35	723
Total Number	4640		6827

ment patterns performed only by players within the position (considered as the signature movement patterns) was identified and visualised. Also, across all playing positions, the unique set of movement patterns performed by all players was identified.

6.1.3 RFL Playing Position Classification using Movement Patterns

The unique set of movement patterns across all playing positions was used as predictor variables to develop two tabular datasets. The two datasets differ from one another due to the values of their predictor variables. The first dataset was populated with “Binary” values while the second was populated with “Rel_Freq” values or zeros. We refer to both datasets as “original datasets”. The target variable values of the developed datasets for classification modellings were all nine playing positions.

Due to the class imbalance problem found in both original datasets (Table 6.1), both oversampling of minority playing positions and undersampling of majority playing positions were considered in this study. The “SMOTE” method (discussed in Section 3.3.2.2) was applied to over-sample seven minority playing positions (See Table 6.1). The generated dataset was referred to as “oversampled dataset”. Also, the “Random-Under-Sampler” method (discussed in Section 3.3.2.2) was applied to under-sample all playing position observations except the minority class (i.e., Five-Eighths).

For each input dataset (based on the predictor variables values), the total number of observations for each playing position varied from original to over-sampled mi-

nority playing positions and under-sampled majority playing positions. The original datasets consist of four thousand six hundred and forty observations representing players' movement sequences at player-per-fixture level. The over-sampled datasets contained six thousand eight hundred and twenty-seven observations. The under-sampled datasets contained one thousand six hundred and ninety-two observations. In total, six datasets were developed as input for classification modelling. Three (i.e. original, over-sampled and under-sampled) datasets with "Binary" predictor variables values and the other three datasets were populated with zeros or "Rel.Freq" values.

Classification models were fitted with the same algorithm as selected in section 5.1.3.3. Five machine learning classification algorithms (discussed in Section 3.2.2.1) representing five different types of learning techniques were used to fit classification models on both "Binary" and "Rel.Freq" datasets. The selected machine learning algorithms are Decision Tree, Gaussian Naive Bayes, Random Forest, Logistic Regression and Multi-Layered Perceptron as they all fit distinct models. The classification algorithms were implemented in the scikit-learn (version 1.1.3) python module (Pedregosa et al., 2011). The parameters for fitting each classification algorithm are presented in Table 5.1.

The classification models were fitted via cross-validation technique (discussed in Section 3.3.2.3) with the *n_splits* parameter set to 10, *random_state* parameter set to 10 and the *shuffle* parameter set to "True". The cross-validated models' performances were evaluated (discussed in Section 3.3.2.3) through aggregated accuracy, precision, recall and f1-score metrics respectively. The performance scores of each classification model were collected and presented. Classification models were developed and evaluated on the six input datasets.

6.1.3.1 Key Movement Patterns for Classification

The input datasets used for the classification of elite Rugby Football League players into the nine playing positions based on movement patterns suffer from high dimensionality (i.e., a large number of predictor variables). Additionally, a large number of identified movement patterns as predictor variables may contain correlated ones as well as variables that do not offer quality information to classify observations of rugby league players into correct target variable values. Given that the importance and

benefits of players' playing position classification are the implementations of players' positional profiling for talent identification and performance variability assessment, the current feature space for the classification of players into playing positions is not tenable.

The type of predictor variables values (i.e. "Binary" or "Rel.Freq") of the data that offered the highest classification accuracy is considered the appropriate predictor variables values. Hence, such data and its sampled variants are used for the analysis to identify key movement patterns for classification and referred to as "main" datasets. The top significant unique movement patterns as predictor variables that were important for accurate classification modelling, as identified by classification model feature importance technique, feature selection technique (discussed in Section 3.3.2.4) and SHapley Additive exPlanation (SHAP) (discussed in Section 3.3.2.4), were used to reduce the number of identified signature movement patterns.

Feature Selection: The identification of key movement patterns to classify elite Rugby Football League Players into nine playing positions was carried out by applying the filter-based feature subset selection techniques (discussed in section 3.3.2.4). The correlation-based feature subset selection method and the consistency subset evaluator method were implemented, both with the best first search method. Both feature selection techniques were selected because they output the best subset of features through different mechanisms and without user bias.

The two feature selection techniques discussed in Section 3.3.2.4 (i.e., correlation-based feature subset selection and consistency-based feature subset selection using the best first search method) were applied to the "main" datasets to identify key movement patterns. The identified key movement patterns were used to extract new data subsets and refer to these new data subsets as *key movement patterns datasets*. The "main" datasets and its two (2) *key movement patterns datasets* are used as input data to re-fit classification models.

Classification models are re-fitted with the same algorithm as selected in Section 5.1.3.3 with the same parameter defined in Table 5.1 following the same procedure as discussed in Section 3.3.2.3. The classification algorithms were implemented in the sci-kit-learn (version 1.1.3) python module (Pedregosa et al., 2011). The performance scores of each classification model were collected and presented. Also, the accuracy

scores of classification models per feature selection method were aggregated to check for stability and validate which feature selection method identified the key or a minimum number of movement patterns across the selected classification algorithms.

Feature Importance and Contribution: Following the classification of players into playing positions via five machine learning algorithms fitted on six different input datasets, the model with the highest classification accuracy was further analysed. The movement patterns that were most important to the most accurate classification model were identified through the model’s feature importance technique.

The “kernelExplainer” SHAP values technique (discussed in Section 3.3.2.4) was applied to identify the contribution of each unique movement pattern towards the classification of elite rugby league players into playing positions based on the most accurate classification model. Given that the classification modelling of players into playing positions will generate SHAP values of all observations for each playing position (i.e. nine sets of SHAP values), the most significant movement patterns per playing position were first identified. Afterwards, a matrix of SHAP values of all observations was generated by selecting each observation’s SHAP value within its playing position. With the generated matrix, the top significant movement patterns across playing positions were identified.

6.1.4 Assessment of Players’ Performance Variability

Rugby league players’ performance over two seasons (2019 and 2020) was assessed based on a set of significant movement patterns across playing position levels. The longitudinal performance variability of rugby league players was assessed using the obtained set of identified significant movement patterns.

The set of unique movement patterns that were identified by the feature selection technique as key movement patterns and the other set that was identified by the SHAP value feature contribution (discussed in Section 3.3.2.4) were cross-matched for overlap. Each of the overlapped movement patterns (considered as “Movement Performance Indicators”) was used to assess the performance variability of players across playing positions. For each selected player, a matrix of a set of movement patterns over fixtures was generated and populated by the relative frequency of each movement

performance indicator if performed by the player during a specific match game. Per movement pattern, the longitudinal performance variability of the selected player is visualized for all participated match games.

6.2 Results and Discussion

Tables 3.2 contained pairs of movement units and characters that are useful to denote the movement patterns extracted and presented.

6.2.1 Signature Movement Patterns of RFL Playing Positions

The total number of movement sequences across all four thousand six hundred and forty sets of movement sequences is three million three hundred and forty thousand two hundred and sixteen (3,340,216). The extraction of closed contiguous movement patterns with a maximum length of two seconds (i.e., 20) and five per cent relative frequency support threshold from the four thousand six hundred and forty sets of movement sequences resulted in the discovery of various sizes of discovered movement patterns per-player-per-fixture. Across all sets of movement sequences, the maximum number of movement patterns extracted per player per fixture was one thousand five hundred and eighty-six (1586) while the minimum number of extracted movement patterns was three. A total of one million four hundred and ninety-two thousand and twenty-five (1,492,025) movement patterns were extracted while the average number of movement patterns extracted per set of movement sequences is three hundred and twenty-one (321).

The identified and unique (i.e., signature) movement patterns performed by players within each playing position are discussed in the sections below. These signature movement patterns revealed the unique external load completed by players belonging to each group.

6.2.1.1 Centres

A total of three thousand three hundred and seven movement patterns were identified as those performed by centres. The number of times each identified movement pattern was performed by all centre players varied from one to three hundred and sixty-five

6.2 Results and Discussion



Figure 6.2: Signature Movement Patterns per Playing Positions (Adeyemo, 2023)

thousand six hundred and ninety-two. Of these identified movement patterns, one thousand two hundred and forty-one movement patterns were uniquely performed (i.e. signature movement patterns) by elite rugby league centres. This is approximately 37.5% of the total number of movement patterns identified to be performed by rugby league centre players. The signature movement patterns performed by elite rugby league centres are visualised in Figure 6.2a. Some of the frequently performed centres' signature movement patterns are "eeeeee, qqquqq, iiiiii, OLP and CDCCC".

6.2.1.2 Five Eighths

A total of one thousand one hundred and ninety-nine movement patterns were identified as those performed by five-eighths. The number of times each identified movement pattern was performed by all centre players varied from six to fifty-six thousand four hundred and five. Of these identified movement patterns, only thirteen movement patterns were uniquely performed (i.e. signature movement patterns) by elite rugby league five eights. This is approximately 1.08% of the total number of movement patterns identified to be performed by rugby league five-eighth players. The signature movement patterns performed by elite rugby league five-eighths are visualised in Figure 6.2b. These signature movement patterns are characterised by unique combinations of movement units "u" "v", "q" "r", "m", "n", "e" and "a".

6.2.1.3 Full Backs

A total of one thousand five hundred and sixty-one movement patterns were identified as those performed by full-backs. The number of times each identified movement pattern was performed by all full-back players varied from one to ninety-six thousand five hundred and seventy-four. Of these identified movement patterns, only one hundred and twenty-two movement patterns were uniquely performed (i.e. signature movement patterns) by elite rugby league full-backs. This is approximately 7.82% of the total number of movement patterns identified to be performed by rugby league full-backs. The signature movement patterns performed by elite rugby league centres are visualised in Figure 6.2c. Some of the frequently performed fullbacks' signature movement patterns are "TP, yyyzyyyy, nnmo, LT, xxv and SU".

6.2.1.4 Half Backs

A total of two thousand eight hundred and eighty-one movement patterns were identified as those performed by half-backs. The number of times each identified movement pattern was performed by all full-back players varied from one to one hundred and sixty-nine thousand nine hundred and fifty-eight. Of these identified movement patterns, only six hundred and twelve movement patterns were uniquely performed (i.e. signature movement patterns) by elite rugby league half-backs. This is approximately 21.24% of the total number of movement patterns identified to be performed by rugby league half-back players. The signature movement patterns performed by elite rugby league centres are visualised in Figure 6.2d. These signature movement patterns are characterised by unique combinations of movement units “e”, “f”, “G”, “v”, “u”, “q”, “i”, “n” and “m”.

6.2.1.5 Hookers

A total of one thousand two hundred and eighty-two movement patterns were identified as those performed by hookers. The number of times each identified movement pattern was performed by all full-back players varied from one to one hundred and twelve thousand six hundred and four. Of these identified movement patterns, only three hundred movement patterns were uniquely performed (i.e. signature movement patterns) by elite rugby league hookers. This is approximately 23.4% of the total number of movement patterns identified to be performed by rugby league hooker players. The signature movement patterns performed by elite rugby league centres are visualised in Figure 6.2e. Some of the frequently performed hookers’ signature movement patterns are “vuuuuq, bbbbb, vwvv, and GTS”. Players within this playing position frequently performed signature movement patterns with a longer number of continuous movement patterns than in other playing positions.

6.2.1.6 Loose Forwards

A total of one thousand one hundred and seventy-seven movement patterns were identified as those performed by loose forwards. The number of times each identified movement pattern was performed by all full-back players varied from four to three

hundred and seventy-four thousand four hundred and twenty-eight. Of these identified movement patterns, only thirteen movement patterns were uniquely performed by elite rugby league loose-forwards. The movement patterns uniquely performed (i.e. signature movement patterns) by rugby league loose-forward players is approximately 1.11% of the total number of movement patterns performed by players of the playing position. The signature movement patterns performed by elite rugby league loose forwards are visualised in Figure 6.2f. Some of the frequently performed loose forwards' signature movement patterns are "HGS, jvw, zznmm, and ffeffe".

6.2.1.7 Prop Forwards

A total of two thousand seven hundred and twelve movement patterns were identified as those performed by prop forwards. The number of times each identified movement pattern was performed by all full-back players varied from one to one hundred and forty-five thousand two hundred and forty-nine. Of these identified movement patterns, only seven hundred and seventy-one movement patterns were uniquely performed (i.e. signature movement patterns) by elite rugby league prop forwards. This is approximately 28.43% of the total number of movement patterns identified to be performed by rugby league prop-forward players. The signature movement patterns performed by elite rugby league prop forwards are visualised in Figure 6.2g. These signature movement patterns are characterised by unique combinations of movement units "e", "f", "v", "u", "q", "i", "m", "a" and "r".

6.2.1.8 Second Rows

A total of one thousand nine hundred and sixty-seven movement patterns were identified as those performed by second rows. The number of times each identified movement pattern was performed by all full-back players varied from one to one hundred and fifteen thousand one hundred and sixty-eight. Of these identified movement patterns, only one hundred and ninety-four movement patterns were uniquely performed (i.e. signature movement patterns) by elite rugby league second rows. This is approximately 9.86% of the total number of movement patterns identified to be performed by rugby league second row players. The signature movement patterns performed by elite rugby league second rows are visualised in Figure 6.2h. Some of the fre-

quently performed second rows' signature movement patterns are “SSSSSTSSSSSS, KKKKKLKK, GGGHS, KOOKKOK and uvvuuuv”.

6.2.1.9 Wingers

A total of three thousand one hundred and seventy-four movement patterns were identified as those performed by wingers. The number of times each identified movement pattern was performed by all full-back players varied from one to one hundred and twenty-eight thousand eight hundred and thirty. Of these identified movement patterns, one thousand and sixty-six movement patterns were uniquely performed (i.e. signature movement patterns) by elite rugby league wingers. This is approximately 33.59% of the total number of movement patterns identified to be performed by rugby league winger players. The signature movement patterns performed by elite rugby league wingers are visualised in Figure 6.2i. Some of the frequently performed wingers' signature movement patterns are “GGSSSSSSS, uuuuuuuuuGGGGGGG, aeef, eeife and fffg”.

6.2.2 RFL Playing Position Classification using Movement Patterns

Across all playing positions, a unique set of seven thousand one hundred and sixty-seven (7,167) movement patterns were derived by computing the union of the extracted movement patterns for all players per fixture. This unique set of movement patterns was used as predictor variables to develop datasets for classification modelling.

Classification results are presented and discussed based on the binary predictor variable values (i.e., Section 6.2.2.1) and relative frequency count predictor variable values (i.e., Section 6.2.2.2)

It is noteworthy to point out that the overall default probability of correctly predicting the playing positions of rugby players per fixture among the nine playing positions is computed as $(100 / 9) \%$ i.e., 11.11%.

6.2.2.1 Classification Based on Binary values

Three datasets were generated for classification modelling. The first dataset represents the original distribution of the player per fixture observations. The second dataset rep-

resents the undersampled distribution of the player per fixture observations. The third dataset represents the oversampled distribution of the player per fixture observations.

The performance evaluation of all five (5) classification models fitted on the original, oversampled and undersampled “Binary” values datasets were obtained (Table 6.2). Elite Rugby Football League players can be classified into nine playing positions based on movement patterns via the original “Binary” values dataset at a minimum accuracy of 10.3%, 0.23 precision, 0.17 recall and 0.1 f1-score as produced by the Gaussian Naive Bayes model.

The Multi-Layered Perceptron classification model classified rugby league players on the original “Binary” values dataset at an increased accuracy of 49.96%, 0.46 f1-score, 0.47 recall and 0.49 precision. Random Forest fitted the most accurate classification model capable of classifying the Rugby Football League elite players into playing positions at an accuracy of 50.3%, 0.51 precision and 0.42 f1-score and recall scores respectively.

Four of the five classification models fitted on the original “Binary” values dataset achieved an accuracy higher than the overall default prediction probability except for the Gaussian Naive Bayes model. The low performance of the Naive Bayes algorithm can be associated with its naive assumption of predictor variables whereas the variables are high-dimensional and there maybe be correlations.

Similarly, RFL players can be classified by the Decision Tree classification model into their playing positions based on movement patterns via the undersampled “Binary” data at an accuracy of 10.4%, 0.1 precision, 0.11 recall, and 0.1 F1-score (Table 6.2). The MLP classifier classification model achieved an accuracy of 11.4%, 0.1 precision and recall score of 0.12 and F1-score of 0.09 via the undersampled “Binary” dataset. The Random Forest classification model produced the maximum accuracy of 11.7%, and 0.12 F1-score, recall and precision scores respectively.

Table 6.2: Classifiers’ Separation Accuracies of RFL Playing Positions based on Binary Values (Adeyemo, 2023)

Classifier	Precision	Recall	F1_Score	Sampling Method	Column Value	Accuracy (%)
Decision Tree	0.29	0.29	0.29	N/A	Binary	31.31
Gaussian Naïve Bayes	0.23	0.17	0.1			10.3
Random Forest	0.51	0.42	0.42			50.3
Logistic Regression	0.45	0.44	0.44			47.09
MLP	0.49	0.47	0.46			49.96
Decision Tree	0.1	0.11	0.1	Random Undersampler	Binary	10.4
Gaussian Naïve Bayes	0.1	0.11	0.06			10.99
Random Forest	0.12	0.12	0.12			11.7
Logistic Regression	0.1	0.11	0.1			10.52
MLP	0.1	0.12	0.09			11.4
Decision Tree	0.44	0.44	0.44	SMOTE	Binary	43.97
Gaussian Naïve Bayes	0.39	0.24	0.19			23.71
Random Forest	0.74	0.73	0.73			73.14
Logistic Regression	0.65	0.66	0.65			65.26
MLP	0.72	0.72	0.72			71.74

The separation accuracies of the selected five machine learning classification models on the undersampled “Binary” dataset were lower than the overall default prediction probability in three cases (Decision tree, Gaussian Naive Bayes, and Logistic Regression classification models). Meanwhile, the accuracies of the other two classifiers (i.e., Random Forest and MLP) were slightly but insignificantly higher than the overall default prediction probability when fitted on the undersampled “Binary” tabular dataset (Table 6.2). This indicates that more observations are required to improve classification accuracy of rugby league players into playing positions.

The performances of the classification models towards classification of elite Rugby Football League players into playing positions based on movement patterns via the oversampled “Binary” dataset revealed that the Gaussian Naive Bayes classification model had the minimum accuracy of 23.71%, 0.39 precision, 0.24 recall and 0.19 f1-score (Table 6.2). The Multi-Layered perceptron classification model fitted on the same dataset produced an accuracy of 71.74% and 0.72 f1-score, recall and precision scores respectively. The classification model with the maximum performance on the oversampled “Binary” was Random Forest model with an accuracy of 73.14%, 0.74 precision score and 0.73 f1-score and recall scores respectively. Four of the five classification models fitted on the oversampled “Binary” dataset had accuracies above 50% except the Decision tree and Gaussian Naive Bayes classification models. The Decision Tree low performance (i.e. underfitting) can be attributed to the large number of predictor variables as the classification model’s choice of decision splits will be difficult because of numerous (and maybe correlated) variables (Table 6.2). This is evident in the performance prowess of the Random Forest model which was able to build multiple uncorrelated trees with bootstrapped observations that led to excellent classification accuracy.

Each classification model had better performance on the original “Binary” dataset (with the exception of Gaussian Naive Bayes of 10.3%) than the undersampled “Binary” dataset. All classification models produced their best classification performances on the oversampled “Binary” dataset and the Random Forest classification model was the most accurate classification model (Table 6.2). In practice, sports practitioners can utilise the Random Forest model with a binary-based values and a large number of observations to successfully classify rugby league players into playing positions.

6.2.2.2 Classification Based on Relative Frequency values

Another three datasets were also generated for this classification modelling. The first dataset represents the original distribution of the player per fixture observations. The second dataset represents the undersampled distribution of the player per fixture observations. The third dataset represents the oversampled distribution of the player per fixture observations.

The performance evaluation of all five classification models fitted on the original, oversampled and undersampled “Rel.Freq” datasets were obtained (Table 6.3). Elite Rugby Football League players were classified into nine playing positions based on movement patterns via the original “Rel.Freq” dataset at the lowest accuracy of 10.41%, 0.24 precision, 0.16 recall and 0.1 f1-score as produced by the Gaussian Naive Bayes model. The Logistic Regression classification model classified players of the original “Rel.Freq” values dataset at an increased accuracy of 53.53%, 0.46 f1-score, 0.45 recall and 0.53 precision.

Multi-Layered Perceptron fitted the most accurate classification model capable of classifying Rugby Football League elite players into playing positions on the original “Rel.Freq” dataset at an accuracy of 53.79%, 0.51 precision and 0.49 f1-score and recall scores respectively. Four of the five classification models fitted on the original “Rel.Freq” dataset achieved an accuracy higher than the overall default prediction probability except for the Gaussian Naive Bayes model (Table 6.3). The low performance of the Naive Bayes algorithm can be associated with its naive assumption of predictor variables whereas these variables are highly dimensional and there may be correlations among them.

Elite Rugby Football League players were classified using movement patterns into their playing positions by the Decision tree classification model fitted on the undersampled “Rel.Freq” dataset at a low accuracy of 9.1% and 0.09 precision, recall and f1-score scores respectively (Table 6.3). The Gaussian Naive Bayes classification model achieved an accuracy of 11.58%, 0.07 f1-score, 0.12 recall and 0.11 precision scores respectively. The Logistic regression classification model produced the highest accuracy of 12.41%, f1-score and precision scores of 0.12 and recall of 0.13 via the undersampled “Rel.Freq” dataset.

Table 6.3: Classifiers’ Separation Accuracies of RFL Playing Positions based on Relative Frequency Values (Adeyemo, 2023)

Classifier	Precision	Recall	F1_Score	Sampling Method	Column Value	Accuracy (%)
Decision Tree	0.32	0.31	0.31	N/A	Relative Frequency	33.99
Gaussian Naïve Bayes	0.24	0.16	0.1			10.41
Random Forest	0.53	0.43	0.44			51.9
Logistic Regression	0.53	0.45	0.46			53.53
MLP	0.51	0.49	0.49			53.79
Decision Tree	0.09	0.09	0.09	Random Undersampler	Relative Frequency	9.1
Gaussian Naïve Bayes	0.11	0.12	0.07			11.58
Random Forest	0.1	0.1	0.1			10.17
Logistic Regression	0.12	0.13	0.12			12.41
MLP	0.1	0.12	0.07			11.64
Decision Tree	0.46	0.46	0.46	SMOTE	Relative Frequency	45.51
Gaussian Naïve Bayes	0.38	0.23	0.18			22.09
Random Forest	0.73	0.74	0.73			73.41
Logistic Regression	0.6	0.61	0.6			60.37
MLP	0.73	0.73	0.73			72.85

The separation accuracies of the selected five machine learning classification algorithms on the undersampled “Rel.Freq” dataset were lower than the default prediction probability in two cases which are the Decision Tree and Random Forest classification models (Table 6.3). Meanwhile, the accuracies of the other three classifications model (i.e., Gaussian Naive Bayes, Logistic Regression and MLP) were slightly higher than the default prediction probability when fitted on the undersampled “Rel.Freq” dataset (Table 6.3). This result also reinforces that a low number of observations leads to lower separation accuracy as the classification models’ accuracies are lower in comparison to models’ accuracies on the original “Rel.Freq” dataset.

Using the oversampled “Rel.Freq” dataset, the performances of the classification models towards classification of elite Rugby Football League players into playing positions based on movement patterns revealed that the Gaussian Naive Bayes classification model had the lowest accuracy of 22.09%, 0.38 precision, 0.23 recall and 0.18 f1-score (Table 6.3). The Multi-Layered perceptron classification model fitted on the same dataset produced an accuracy of 72.85% and 0.73 f1-score, recall and precision scores respectively. The classification model with the highest performance on the oversampled “Rel.Freq” dataset was the Random Forest model with an accuracy of 73.41%, 0.74 recall score and 0.73 f1-score and precision scores respectively. Four of the five classification models fitted on the oversampled “Binary” dataset had accuracies above 50% except for the Decision tree and Gaussian Naive Bayes classification models. The Decision Tree low performance (i.e. underfitting) can be attributed to a large number of predictor variables as the classification choice of decision splits will be difficult because of numerous (and maybe correlated) variables (Table 6.3). The Random Forest model performance is attributed to its multiple uncorrelated trees with bootstrapped observations that led to excellent classification accuracy.

Each classification model had better performance on the original “Rel.Freq” dataset (with the exception of Gaussian Naive Bayes of 10.41%) than on the undersampled “Rel.Freq” dataset. Furthermore, all classification models produced their best classification performances on the oversampled “Rel.Freq” dataset and the Random Forest classification model was the most accurate (Table 6.3). The comparative analysis of the performance of the classification models between the original “Binary” and “Rel.Freq” datasets revealed that all classification models had their respective better performance on the “Rel.Freq” dataset. For example, the Multi-Layered perceptron classification

model had the highest classification accuracy of 53.79% on the original “Rel_Freq” dataset whereas it achieved a 49.96% on the original “Binary” dataset. Similarly, the Logistic Regression classifier had a better accuracy of 12.41% on the undersampled “Rel_Freq” dataset when compared to its 10.52% accuracy on the undersampled “Binary” tabular dataset. Also, the Random Forest classifier had a 73.41% accuracy on the oversampled “Rel_Freq” dataset which is not significantly better but higher than its 73.14% accuracy on the oversampled “Binary” dataset.

In the end, the performances of the classification models on the original, undersampled and oversampled “Binary” or “Relative” datasets indicate that accurate classification of elite Rugby League players into playing positions based on movement patterns depends on the total number of observations. A large number of observations can guarantee the production of classification models with excellent classification accuracy. Also, the “Rel_Freq” dataset (whether the original or undersampled or oversampled) offered better separation accuracy.

6.2.2.3 Key Movement Patterns for Classification

Feature Selection

The use of seven thousand one hundred and sixty-seven predictor variables for classification modelling of elite rugby players into playing positions is generally bad practice and does not offer useful practical applications. From the results of classifying elite rugby league players into playing positions based on two types of predictor variable values (Section 6.2.2), the “Rel_Freq” input datasets (especially on the original input data) offered better data quality for the classification of players into nine rugby league playing positions. Hence, the original and oversampled “Rel_Freq” input datasets were considered for this analysis and are referred to as “main” datasets.

Each one of the “main” datasets and its two (2) *key movement patterns datasets* (generated using identified key movement patterns) were used as input data to fit classification models. Altogether, classification models were re-developed and re-evaluated on the six input datasets. The results of the classification of elite rugby league players based on identified key movement patterns on each of the main data are presented and discussed below.

Original relative frequency dataset

The application of filter-based feature subset selection feature selection method on the original “Rel.Freq” dataset resulted in the identification of varying and different numbers of key movement patterns. The correlation-based feature subset using the best first search method identified fifty-four (54) key movement patterns out of the original “Rel.Freq” seven thousand one hundred and sixty-seven (7,167) movement patterns for classification of elite Rugby Football League players into nine playing positions. This subset of fifty-four closed contiguous movement patterns had the highest merit score of 0.126 out of all the four hundred and twenty-one thousand one hundred and fifty (421,150) evaluated subsets (See Appendix B.1). Also, these identified key movement patterns are approximately 0.75% of the total number of movement patterns as predictor variables in the original “Rel.Freq” dataset.

The consistency-based feature subset using the best first search method identified thirty-two (32) key movement patterns out of the original “Rel.Freq” seven thousand one hundred and sixty-seven (7,167) movement patterns for classification of elite Rugby Football League players into playing positions. This subset of thirty-two key closed contiguous movement patterns had the highest score of 0.999 out of all the two hundred and sixty-four thousand five hundred and nineteen (264,519) evaluated subsets (See Appendix B.2). Also, these identified key movement patterns are approximately 0.45% of the total number of movement patterns as predictor variables in the original “Rel.Freq” dataset.

The Multi-Layered Perceptron classifier fitted on the original “Rel.Freq” dataset with seven thousand one hundred and sixty-seven movement patterns as predictor variables still achieved the highest accuracy of 53.79%, 0.51 precision, f1-score and recall scores of 0.49 respectively (Table 6.4). However, the Logistic Regression classifier fitted on the *key movement patterns* “Rel.Freq” datasets with fifty-four movement patterns identified by the correlation-based feature subset selection using the best first search method achieved an accuracy of 49.83%, 0.49 f1-score, precision and recall scores of 0.89 (Table 6.4). The Decision Tree classifier fitted on the *key movement patterns* “Rel.Freq” dataset with thirty-two movement patterns identified by the consistency subset evaluator using the best first search method achieved an accuracy of 48.21%, 0.41 f1-score, 0.44 precision and 0.42 recall score (Table 6.4).

Table 6.4: Classification results on original and its key movement patterns datasets based on Relative Frequency Values (Adeyemo, 2023)

Classifier	Accuracy (%)	Average Accuracy (%)	Precision	Recall	F1-score	Feature selection Method	No of Features
Decision Tree	33.99	41.99	0.33	0.33	0.33	-	7167
Gaussian Naïve Bayes	10.41		0.24	0.16	0.1		
RandomForest	51.9		0.53	0.43	0.44		
Logistic Regression	53.53		0.53	0.45	0.46		
MLP	53.79		0.51	0.49	0.49		
Decision Tree	48.36	40.255	0.5	0.46	0.46	Correlation-based feature subset selection using Best First Search Method	54
Gaussian Naïve Bayes	32.37		0.3	0.3	0.29		
RandomForest	26.21		0.35	0.33	0.26		
Logistic Regression	49.83		0.49	0.42	0.42		
MLP	41.1		0.36	0.31	0.29		
Decision Tree	48.21	37.28	0.44	0.42	0.41	Consistency Subset Evaluator using Best First Search Method	32
Gaussian Naïve Bayes	43.81		0.42	0.4	0.4		
RandomForest	29.91		0.27	0.27	0.27		
Logistic Regression	29.29		0.28	0.3	0.26		
MLP	46.72		0.45	0.39	0.39		

An interesting result is the performance of the Gaussian Naive Bayes classification model that achieved an accuracy of 10.41% via seven thousand one hundred and sixty-seven movement patterns, the same algorithm now achieved an accuracy of 32.37% via fifty-four movement patterns identified by correlation-based feature subset selection method and an accuracy of 43.81% via thirty-two movement patterns identified by the consistency subset evaluator feature selection method.

A similar performance is that of Decision tree model that had a 33.99% accuracy using seven thousand one hundred and sixty-seven movement patterns as predictor variables but had an increased accuracy of 48.21% accuracy using the thirty-two key movement patterns (identified by consistency subset evaluator feature selection method) and its best model with an increased accuracy of 48.36% using fifty-four key movement patterns (identified by correlation-based feature subset selection method). These improvements suggest that the reduced size of key movement patterns as predictor variables removed correlated variables that hindered the performances of both classification models.

Given the performances of the classifiers on the *main* and the *key movement patterns* original “Rel_Freq” input datasets, the averaged classification accuracy scores of all six classification models peaked at 42% on the *main* original “Rel_Freq” dataset. Nonetheless, it was closely followed by the averaged classification accuracy of 40.25% on the *key movement patterns* original “Rel_Freq” dataset with fifty-four key movement patterns identified by the correlation-based feature subset feature selection method.

Oversampled Relative Frequency dataset

The application of filter-based feature subset selection method on the oversampled “Rel_Freq” tabular datasets also resulted in the identification of varying and the different total number of key movement patterns. The correlation-based feature subset using the best first search method identified seventy-two (72) key movement patterns out of the oversampled “Rel_Freq” seven thousand one hundred and sixty-seven movement patterns, for the classification of elite rugby league players into playing positions. This subset of seventy-two key movement patterns had the highest merit score of 0.228 among the five hundred and forty-eight thousand nine hundred and thirty-three (548,933) evaluated subsets (See Appendix B.3). These identified key movement patterns are approximately 1% of the total number of movement patterns as predictor

variables in the oversampled “Rel_Freq” dataset.

The consistency-based feature subset using the best first search method identified sixteen (16) key movement patterns out of the oversampled “Rel_Freq” seven thousand one hundred and sixty seven (7,167) movement patterns, for the classification of elite Rugby Football League players into playing positions. This subset of sixteen key movement patterns had the highest score of 1.0, among the one hundred and fifty thousand three hundred and three (150303) evaluated subsets (See Appendix B.4). Also, these identified key movement patterns are approximately 0.22% of the total number of movement patterns as predictor variables in the original “Rel_Freq” dataset.

The Random Forest classification model fitted on the oversampled “Rel_Freq” dataset with seven thousand one hundred and sixty-seven movement patterns as predictor variables still achieved the highest accuracy of 73.41%, 0.74 recall, f1-score and precision scores of 0.73 respectively (Table 6.5).

The Logistic Regression classification model fitted on the *key movement patterns* “Rel_Freq” dataset with seventy-two key movement patterns (identified by the correlation-based feature subset selection using the best first search method) achieved an accuracy of 67.9% and f1-score, precision and recall scores of 0.68 respectively (Table 6.5). The Gaussian Naive Bayes classification model fitted on the *key movement patterns* “Rel_Freq” dataset with sixteen key movement patterns identified by the consistency subset evaluator using the best first search method achieved an accuracy of 61.27%, 0.59 f1-score, 0.61 precision and 0.62 recall scores (Table 6.5).

An interesting result is the performance of the Decision Tree classification model that achieved an accuracy of 45.51% via seven thousand one hundred and sixty-seven movement patterns, the same algorithm achieved an increased accuracy of 61.07% via thirty-two movement patterns identified by the consistency subset evaluator feature selection method and another increased accuracy of 63.6% via seventy-two movement patterns identified by correlation-based feature subset selection method. Similarly, the performance of the Gaussian Naive Bayes model on the data with seven thousand one hundred and sixty-seven movement patterns as predictor variables was increased from 22.09% accuracy to 42.3% accuracy using only seventy-two key movement patterns and later increased to 61.27% accuracy using only sixteen key movement patterns.

Table 6.5: Classification results on oversampled and its key movement patterns datasets based on Relative Frequency Values (Adeyemo, 2023)

Classifier	Accuracy (%)	Average Accuracy (%)	Precision	Recall	F1-score	Feature selection Method	No of Features
Decision Tree	45.51	56.31	0.46	0.46	0.46	-	7167
Gaussian Naïve Bayes	22.09		0.38	0.23	0.18		
RandomForest	73.41		0.73	0.74	0.73		
Logistic Regression	60.37		0.6	0.61	0.6		
MLP	72.85		0.73	0.73	0.73		
Decision Tree	63.6	51.96	0.66	0.65	0.61	Correlation-based feature subset selection using Best First Search Method	72
Gaussian Naïve Bayes	42.3		0.43	0.43	0.42		
RandomForest	34.72		0.39	0.36	0.32		
Logistic Regression	67.9		0.68	0.68	0.68		
MLP	44.48		0.43	0.45	0.43		
Decision Tree	61.07	39.87	0.62	0.61	0.61	Consistency Subset Evaluator using Best First Search Method	16
Gaussian Naïve Bayes	61.27		0.61	0.62	0.59		
RandomForest	36.22		0.37	0.37	0.37		
Logistic Regression	29.51		0.3	0.3	0.29		
MLP	54.34		0.54	0.55	0.54		

Given the performances of all classifiers on the *main* and the *key movement patterns* oversampled “Rel_Freq” dataset, the averaged classification accuracy of all six classification models peaked at 56% on the *main* oversampled “Rel_Freq” tabular dataset. This peaked averaged accuracy was followed by the averaged classification accuracy of 52% on the *key movement patterns* oversampled “Rel_Freq” dataset with seven two key movement patterns identified by the correlation-based feature subset feature selection method.

These results strongly reinforced the application of feature selection methods to solve the problem of high dimensional data problem in sport-related classification modelling. The implemented filter-based feature subset selection methods were able to identify key movement patterns for the classification of elite Rugby League Football players into nine playing positions. Although the key movement patterns could not help to fitting classification models with improved accuracy, however, it greatly reduced the number of required movement patterns used as predictor variables without much accuracy loss.

In a real-time application, given a moderate number of observations, fifty-four key movement patterns are usable as predictor variables for classifying rugby league players into nine playing positions at a Logistic Regression accuracy of 49.83% rather than seven thousand one hundred and sixty-seven movement patterns at 53.53% accuracy. Similarly, given a large number of observations, seven two key movement patterns can be used as predictor variables at a Logistic regression of 67.9% accuracy instead of seven thousand one hundred and sixty-seven movement patterns at a Random Forest 73.41% accuracy. In both cases, the memory consumption rate and execution runtime will be greatly reduced by using the key movement patterns rather than the original set of unique movement patterns used as predictor variables. More so, practitioners will be able to understand the classification models’ predictive prowess based on the key movement patterns, unlike the original set of unique movement patterns.

Feature Importance and Contribution

Although the feature selection technique was able to identify key movement patterns as predictor variables for the classification of elite rugby league players into playing positions, the Random Forest classification model fitted on the oversampled relative frequency data with all unique movement patterns identified by LCCspm algorithm

Table 6.6: Top Fifty Unique Movement Patterns and Classification Importance Scores (Adeyemo, 2023)

Patterns	Importance Scores	Patterns	Importance Scores
ij	0.006057	bbb	0.003567
ijj	0.005951	vu	0.003559
ijj	0.005204	vuvv	0.003531
ii	0.005152	uuv	0.00353
iii	0.005027	iiij	0.003521
jji	0.004937	vuuv	0.003516
jji	0.004933	uuuu	0.003502
HH	0.004788	uu	0.003496
iji	0.004746	no	0.003487
jj	0.00453	bb	0.003432
ji	0.004528	qu	0.00343
HG	0.004409	vv	0.003426
ff	0.004348	iu	0.003414
GH	0.004184	ej	0.003403
mq	0.00388	bba	0.003388
jjj	0.003749	uuu	0.003386
GG	0.003727	uuvu	0.003371
fb	0.003716	GGG	0.003348
vuv	0.003711	fff	0.003347
uv	0.003702	eff	0.003337
vvu	0.003639	qq	0.003328
uvv	0.003632	nn	0.003315
jjj	0.003599	abb	0.003312
ab	0.003594	jf	0.003305
ef	0.003589	aa	0.00329

had the highest accuracy of 73.14%, 0.73 f1-score and precision scores respectively and a 0.74 recall. Hence, the analysis of the feature importance of that model was conducted.

Feature Importance

The LCCspm algorithm extracted a total of seven thousand one and sixty-seven unique movement patterns across playing positions. Only two thousand two hundred and ninety-nine of those unique movement patterns were considered important by the Random Forest classification Model. This is approximately 32% of the total number of unique movement patterns used as predictor variables. Other unique movement patterns had importance scores of zero respectively. The most important movement pattern as considered by the Random Forest model is “ij” with an importance score of 0.006057 while the least important movement pattern is “feeeeff” with an importance score of 0.0000000375. The top fifty most important unique movement patterns are presented in Table 6.6.

The fifty most important unique movement patterns are characterised by two to four lengths of closed contiguous sequences of movement units. These top important movement patterns used in classifying rugby league players are characterised by unique combinations of movement units “i” “j”, “H”, “G”, “m”, “q”, “u”, “v”, “u”, “f”, “b”, “e”, “n” and “o”.

Thirty-one unique movement patterns were ranked by the Random Forest classification model as important with scores ranging between 0.00351 - 0.0061. Among these ranked movement patterns are “jji, HG, ff, mq, ab” (in this non-sequential order) which represent different occurrences of completed external load.

Feature Contribution

The SHAP values contribution of each unique movement pattern used as predictor variables were obtained and each unique movement pattern was ranked based on the values. The SHAP values row for each observation was selected from the SHAP value matrix of each playing position and collated into a new matrix. With the new matrix, the top fifty contributing unique movement patterns were identified and illustrated in Figure 6.3a.

For each playing position, the top fifty contributing unique movement patterns were also identified and illustrated in Figures 6.5b, 6.4 and 6.5. Besides the first eight ranked movement patterns, other movement patterns’ ranks varied across playing positions. This suggests that each unique movement pattern contributed more or less information depending on the rugby league players’ playing positions. For example, the movement pattern “mq” is the twelfth most contributing movement pattern for classifying players into the centre playing position. However, it is the sixteenth most contributing movement pattern for classifying players into the five-eighth and half-back playing position. The same movement pattern is the thirteenth most contributing movement pattern for classifying players into full-back and second-row playing positions. For classifying players into hookers or wingers, the same movement pattern is the tenth most contributing movement pattern while being the fifteenth most contributing movement pattern for classifying players into prop-forwards.

6.2 Results and Discussion

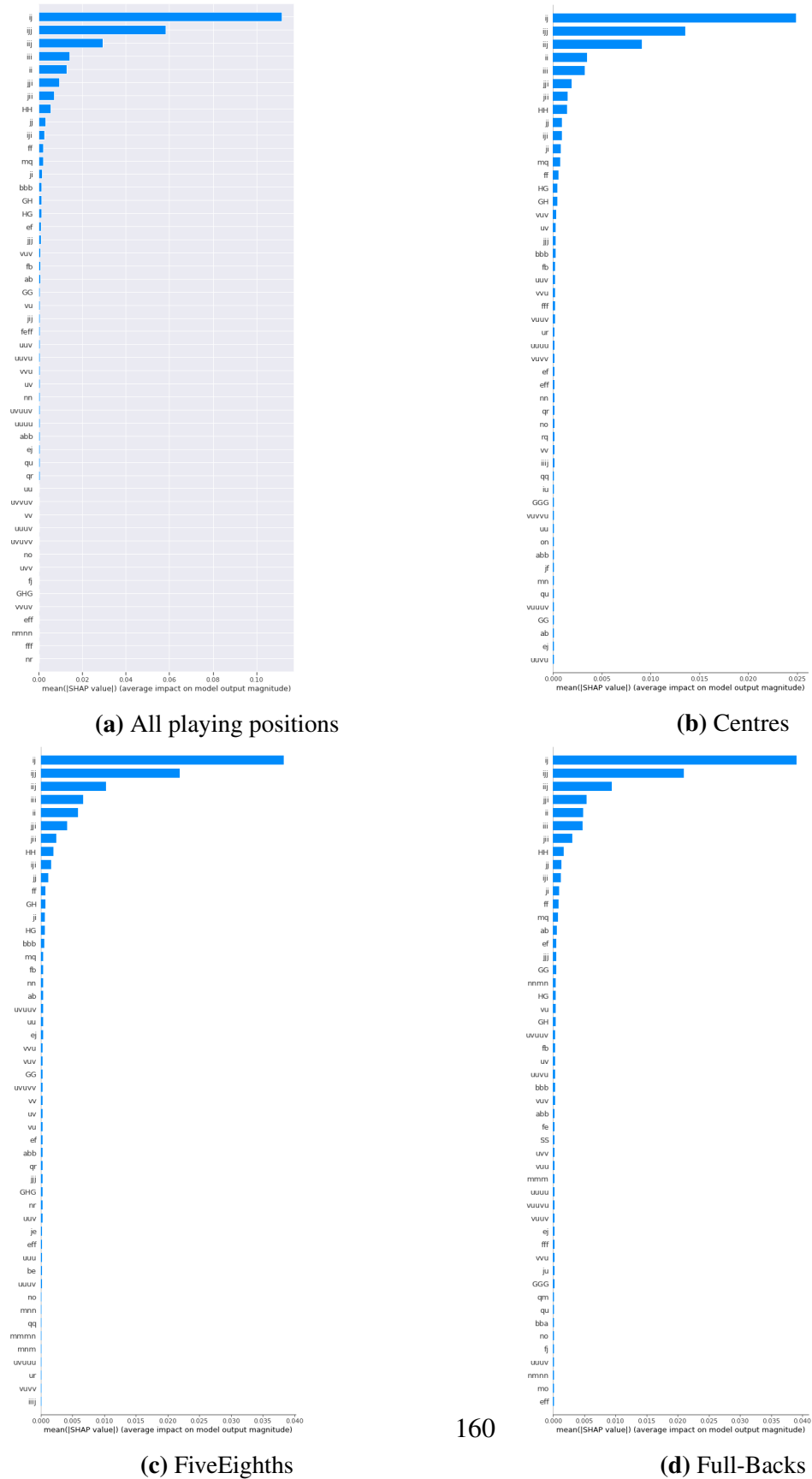


Figure 6.3: SHAP values of Top Fifty Movement Patterns Across and per Playing Positions (Adeyemo, 2023)

6.2 Results and Discussion

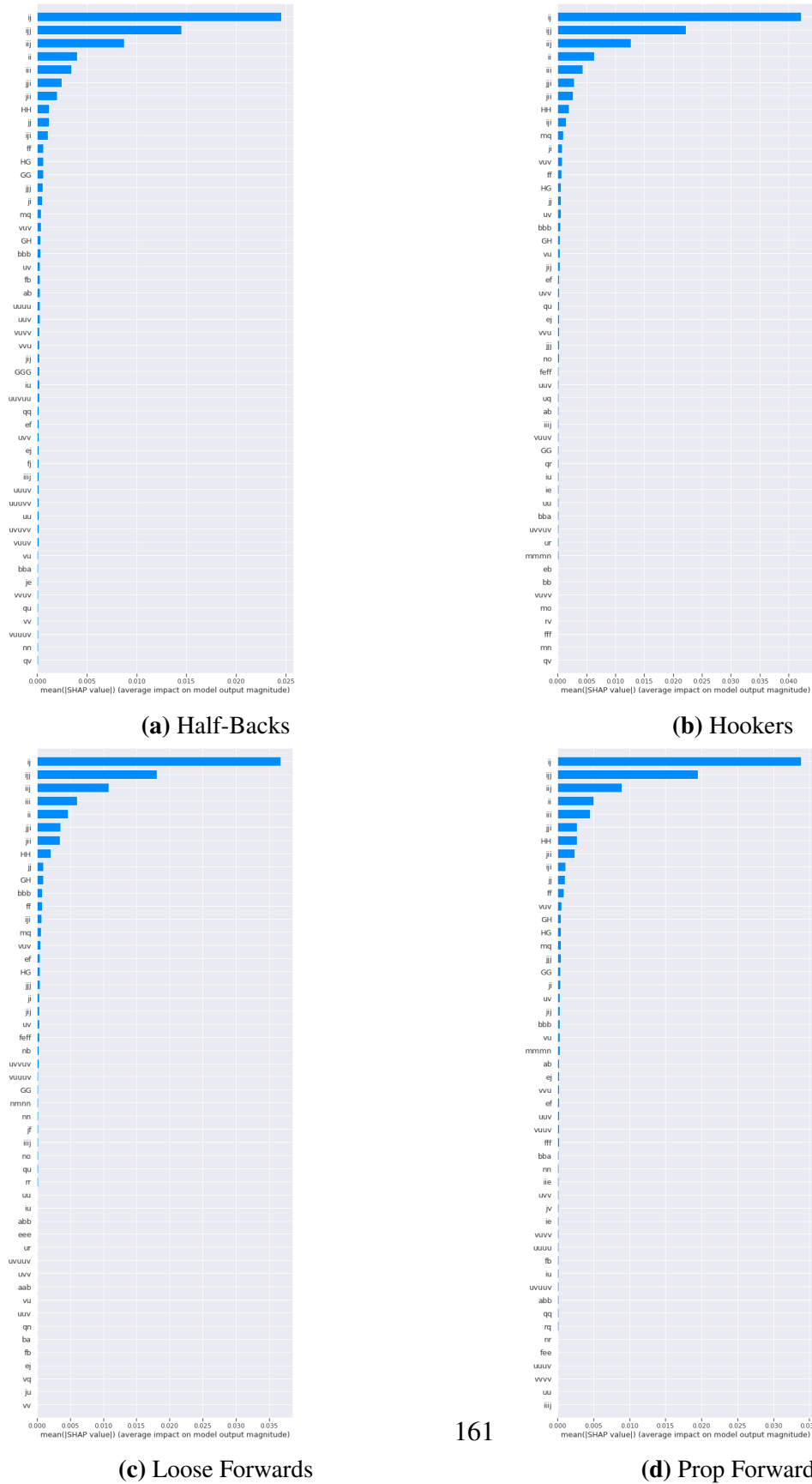


Figure 6.4: SHAP values of Top Fifty Movement Patterns per Playing Positions (Cont'd) (Adeyemo, 2023)

Figure 6.6 illustrates the performance variability of a centre rugby league player based on the selected movement performance indicators. The player participated in forty-eight matches and the rate (i.e., relative frequency) at which each movement performance indicator was performed per match was obtained and visualised. Based on “SS” (Sprint Acceleration Straight, Sprint Acceleration Straight) movement performance indicator, the player performed this movement performance indicator in sixteen of forty-eight matches at an average of 0.056 (i.e., 5.63% of all completed external load per match).

Figure 6.7 illustrates the performance variability of a rugby league five-eighth based on the selected movement performance indicators. The player participated in fifty-six matches and the rate (i.e., relative frequency) at which each movement performance indicator was performed per match was obtained and visualised. Based on “SS” (Sprint Acceleration Straight, Sprint Acceleration Straight) movement performance indicator, the player performed this movement performance indicator in six of fifty-six matches at an average of 0.06 (i.e., 6% of all completed external load per match).

Figure 6.8 illustrates the performance variability of a rugby league full-back based on the selected movement performance indicators. The player participated in fifty-two matches and based on “SS” (Sprint Acceleration Straight, Sprint Acceleration Straight) movement performance indicator, the player performed this movement performance indicator in forty-four of fifty-two matches at an average of 0.06 (i.e., 6.05% of all completed external load per match).

Figure 6.9 illustrates the performance variability of a rugby league half-back based on the selected movement performance indicators. The player participated in fifty-three matches and the rate (i.e., relative frequency) at which each movement performance indicator was performed per match was obtained and visualised. Based on “SS” (Sprint Acceleration Straight, Sprint Acceleration Straight) movement performance indicator, the player performed this movement performance indicator in two of fifty matches at an average of 0.055 (i.e., 5.5% of all completed external load per match).

Table 6.7: A set of Movement Performance Indicators and decoded Movement units ([Adeyemo, 2023](#))

Movement Performance Indicator	Decoded Movement Pattern
SS	Sprint Acceleration Straight, Sprint Acceleration Straight
iiij	[Walk Acceleration Straight]x2, Walk Acceleration Acute-Change
fb	Walk Neutral Acute-Change, Walk Deceleration Acute-change
vuuvu	Jog Acceleration Acute-Change, Jog Acceleration Straight, [Jog Acceleration Acute-Change]x2, Jog Acceleration Straight
on	Jog Deceleration Large-Change, Jog Deceleration Acute-Change
bbb	[Walk Deceleration Acute-Change]x3
ej	Walk Neutral Straight, Walk Acceleration Acute-Change
jjij	[Walk Acceleration Acute-Change]x3
mo	Jog Deceleration Large-Change, Jog Deceleration Large-Change
HH	[Run Acceleration Acute-Change]x2
mq	Jog Deceleration Straight, Jog Neutral Straight
jjij	Walk Acceleration Acute-Change, Walk Acceleration Straight, Walk Acceleration Acute-Change
vuuv	[Jog Acceleration Acute-Change]x2, Jog Acceleration Straight, Jog Acceleration Acute-Change.
ijj	Walk Acceleration Straight, [Walk Acceleration Acute-Change]x2
ij	Walk Acceleration Straight, Walk Acceleration Acute-Change
jjj	[Walk Acceleration Acute-Change]x2, Walk Acceleration Straight

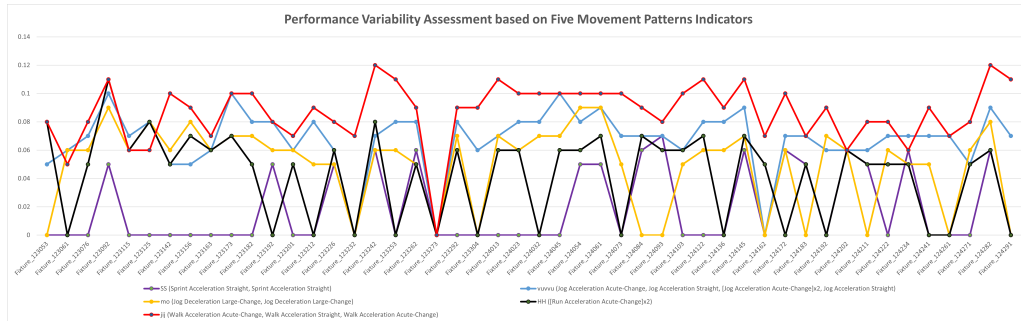


Figure 6.6: A Centre Player Performance Variability Assessment (Adeyemo, 2023)

Figure 6.10 illustrates the performance variability of a rugby league hooker based on the selected movement performance indicators. The player participated in forty-seven matches and the rate (i.e., relative frequency) at which each movement performance indicator was performed per match was obtained and visualised. Based on “SS” (Sprint Acceleration Straight, Sprint Acceleration Straight) movement performance indicator, the player performed this movement performance indicator in three of forty-seven matches at an average of 0.0044 (i.e., 0.44% of all completed external load per match).

Figure 6.11 illustrates the performance variability of a rugby league loose-forward based on the selected movement performance indicators. The player participated in forty-seven matches and the rate (i.e., relative frequency) at which each movement performance indicator was performed per match was obtained and visualised. The player does not perform the movement performance indicator “SS” in any of the participated matches. Based on “vuvvu” (Jog Acceleration Acute-Change, Jog Acceleration Straight, [Jog Acceleration Acute-Change]x2, Jog Acceleration Straight) movement performance indicator, the player performed this movement performance indicator in thirty-six of forty-seven matches at an average of 0.0667 (i.e., 6.67% of all completed external load per match).

Figure 6.12 illustrates the performance variability of a rugby league prop-forward based on the selected movement performance indicators. The player participated in forty-three matches and the rate (i.e., relative frequency) at which each movement performance indicator was performed per match was obtained and visualised. The player performed the movement performance indicator “SS” only in one match at the relative frequency of 0.13 (i.e., 13% of all completed external load in the match). Based on

6.2 Results and Discussion

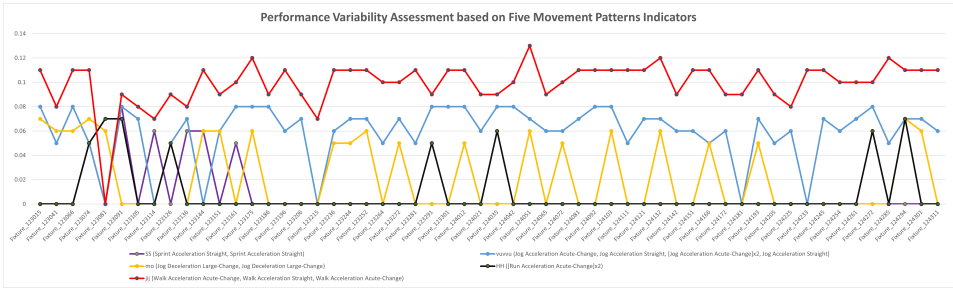


Figure 6.7: A Winger Player Performance Variability Assessment (Adeyemo, 2023)

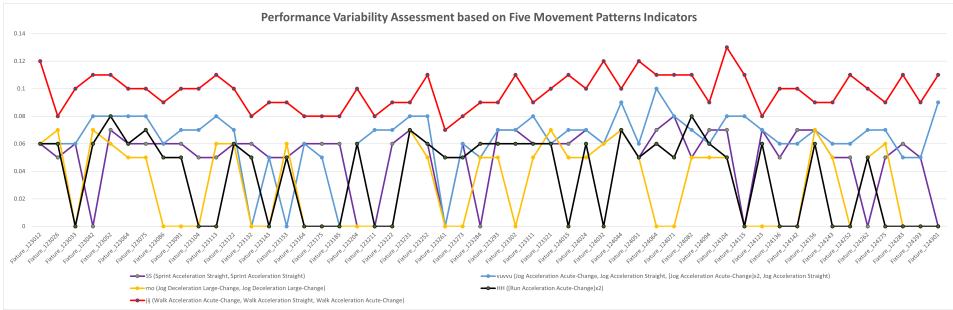


Figure 6.8: A Full-Back Player Performance Variability Assessment (Adeyemo, 2023)

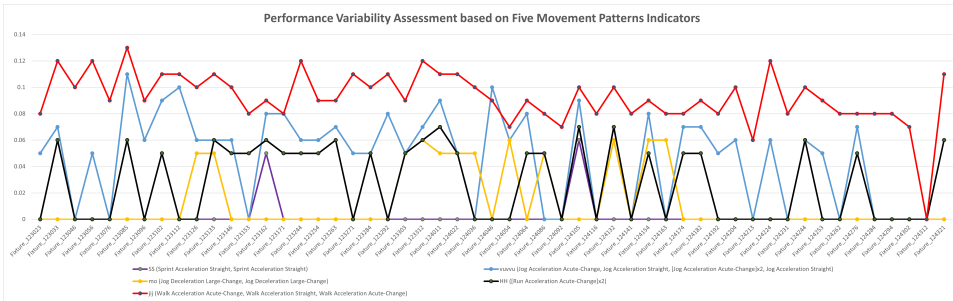


Figure 6.9: A Half-Back Player Performance Variability Assessment (Adeyemo, 2023)

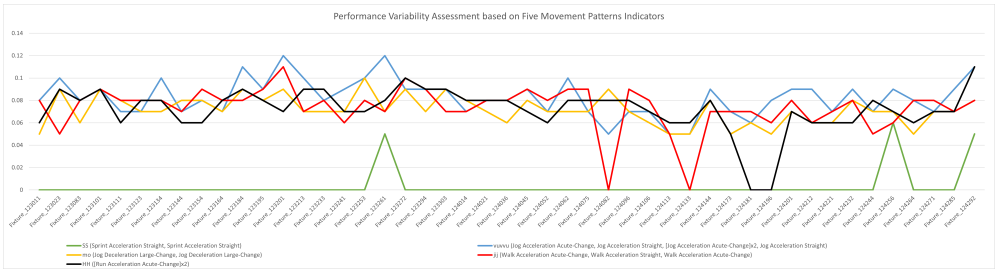


Figure 6.10: A Hooker Player Performance Variability Assessment (Adeyemo, 2023)

6.2 Results and Discussion

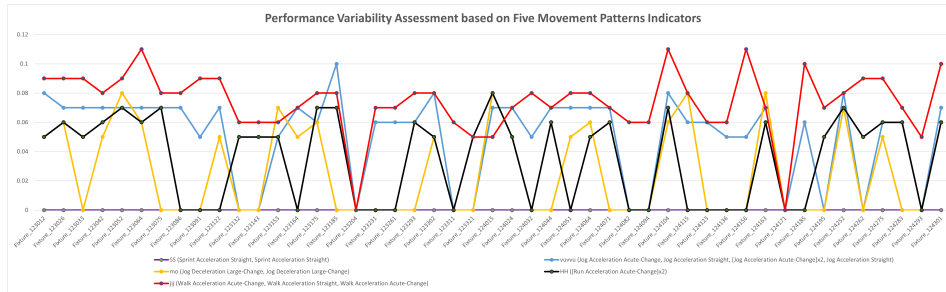


Figure 6.11: A Loose-Forward Player Performance Variability Assessment (Adeyemo, 2023)

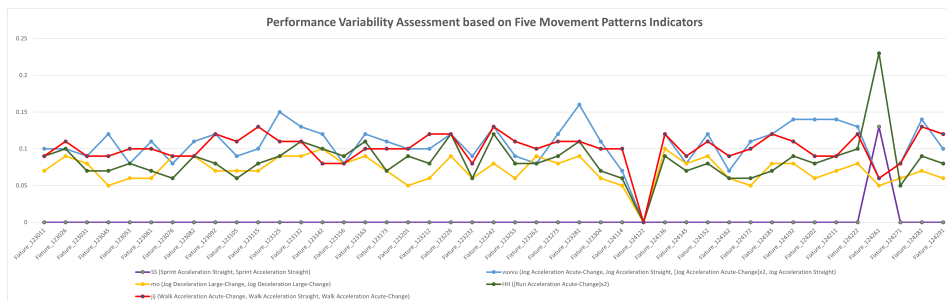


Figure 6.12: A Prop-Forward Player Performance Variability Assessment (Adeyemo, 2023)

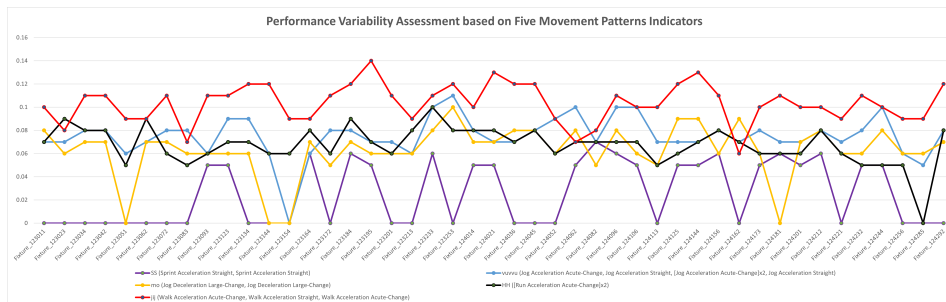


Figure 6.13: A Second-Row Player Performance Variability Assessment (Adeyemo, 2023)

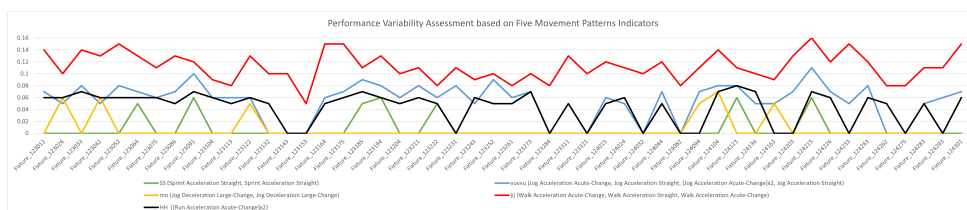


Figure 6.14: A Winger Player Performance Variability Assessment (Adeyemo, 2023)

“vuvvu” (Jog Acceleration Acute-Change, Jog Acceleration Straight, [Jog Acceleration Acute-Change]x2, Jog Acceleration Straight) movement performance indicator, the player performed this movement performance indicator in forty-two of forty-three matches at an average of 0.108 (i.e., 10.79% of all completed external load per match).

Figure 6.13 illustrates the performance variability of a rugby league second-row based on the selected movement performance indicators. The player participated in forty-five matches and the rate (i.e., relative frequency) at which each movement performance indicator was performed per match was obtained and visualised. Based on “SS” (Sprint Acceleration Straight, Sprint Acceleration Straight) movement performance indicator, the player performed this movement performance indicator in twenty-one of forty-five matches at an average of 0.0543 (i.e., 5.43% of all completed external load per match).

Figure 6.14 illustrates the performance variability of a rugby league winger based on the selected movement performance indicators. The player participated in fifty matches and the rate (i.e., relative frequency) at which each movement performance indicator was performed per match was obtained and visualised. Based on “SS” (Sprint Acceleration Straight, Sprint Acceleration Straight) movement performance indicator, the player performed this movement performance indicator in six of fifty matches at an average of 0.05 (i.e., 5.5% of all completed external load per match).

6.3 Chapter Summary

In the previous chapter, LCCspm algorithm was validated as the pattern mining algorithm but with some limitations. This PhD study, through this chapter, resolved those limitations by applying the LCCspm algorithm for the extraction and identification of frequent movement patterns from sets of movement sequences of elite Rugby Football League players to profile and classify players into all nine rugby league playing positions. The relative frequency as predictor variables value was ascertained as the better type of value. Also, significant movement patterns were identified by using filter feature subset selection methods as well as SHAP value feature contribution methods. A set of Movement Performance Indicators was identified and demonstrated for players’ performance variability assessment.

Signature movement patterns for each rugby league playing position were discov-

ered. The signature movement characterised unique combinations of closed contiguous movement units for each position and the number of identified signature movements varied from one playing position to the other. Both Five-Eighth and Loose-Forward playing positions had the least number of signature movement patterns (i.e. thirteen movement patterns). Meanwhile, the Centre playing position had the highest number of identified signature movement patterns.

For the classification of elite rugby league players into all nine playing positions, the Random Forest classification model fitted on the oversampled “Rel_Freq” dataset had the highest accuracy of 73.41%. These results indicate that the relative frequency of performed movement patterns is better and more appropriate as predictor variable values for classifying rugby players into individual playing positions. Also, the results suggest that the separation of rugby league players into playing positions requires a significant number of observations per position and the development of a fairly balanced dataset with respect to playing position observations.

Due to the high number of extracted movement patterns used for classifying elite rugby league into all nine playing positions, key movement patterns were identified by applying the correlation-based feature subset selection using the best-first search method as a feature selection technique. The dataset with the relative frequency of performed movement patterns as value and its oversampled variant was considered for key movement pattern identification analysis. Fifty-four (54) key movement patterns of 7,167 movement patterns were identified by the correlation-based feature subset-selection technique on the “Rel_Freq” dataset while thirty-two (32) key movement patterns were identified by the consistency subset evaluator technique. Seventy-two (72) key movement patterns were identified by the correlation-based feature subset-selection technique on the oversampled “Rel_Freq” dataset while sixteen (16) key movement patterns were identified by consistency subset evaluator technique. For all classification models fitted on datasets for classification based on key movement patterns, the Logistic Regression classification model fitted on the oversampled “Rel_Freq” dataset with seventy-two key movement patterns had the highest accuracy of 67.9%.

A set of Movement Performance Indicators was identified for players’ performance variability assessment. Sixteen movement patterns were identified as Movement Performance Indicators. This set of Movement Performance Indicators was identified across playing positions through the integration of the feature selection method and

feature contribution results. Performance variability assessment of elite rugby league players was demonstrated using nine players (one from each playing position). Each player's performance differs based on each Movement Performance Indicator. Also, players belonging to different playing positions recorded different rates of completing the external load represented by a Movement Performance Indicator.

Based on these results, this PhD study through this chapter achieved its fourth objective and claims that user-defined length of frequent closed contiguous movement patterns extracted by the LCCspm algorithm can be used to identify signature movement patterns for profiling elite rugby league players into playing positions. Also, LCCspm algorithm extracted frequent closed contiguous movement pattern that is usable as predictor variables to classify elite rugby league players into nine playing positions and relative frequency is the appropriate predictor variable value. Also, the LCCspm algorithm extracts movement patterns (through the integration of other advanced analytics techniques) set(s) of significant movement patterns for assessing elite rugby league players' performance variability across playing positions.

Conclusions

The main discourse of this PhD study is that movement patterns can be identified from rugby football league players' GPS data to quantify players' external load and uncover players' behavioural movement patterns during matches for various applications.

Existing frameworks for player movement profiling (Sweeting et al., 2014, 2017; White et al., 2021) are limited by the algorithms implemented to find and extract movement patterns. Four criteria (i.e., user-specified maximal length, item contiguousness, user-defined frequency threshold and frequent pattern lossless compression) of a suitable algorithm for player movement profiling were identified. A review of other existing sequential pattern mining algorithms (with respect to the first objective of this PhD study) revealed that the state-of-the-art algorithm for extracting frequent closed contiguous movement patterns (i.e., CCSpan) (Zhang et al., 2015) satisfied only three of the four criteria of a suitable algorithm for player movement profiling beside suffering other limitations (Abboud et al., 2017; Bermingham and Lee, 2020) such as being unable to scale well on a large and lengthy set of sequences that characterise sport big data. Chapter 4 outlined the rationale for a user-defined length of frequent closed contiguous movement patterns for player movement profiling. A sequential pattern mining algorithm was developed (with respect to this PhD study's second objective) based on a snippet-growth method for candidate pattern generation. The *inverse characteristic* of frequent contiguous patterns was explored to identify the frequent closed contiguous patterns. An implementation of *l*-length closed contiguous sequential pattern mining algorithm (LCCspm) incorporating all these attributes was described.

LCCspm was empirically tested for efficiency and effectiveness using natural sport-related big data pattern mining datasets. The experiment on rugby league and football datasets showed that LCCspm can extract patterns faster, uses lower memory and scales better while extracting patterns from any size of a given set of sequences than CCSpan. Additionally, LCCspm effectively extracts patterns even from a given set of sequences with extremely long sequences. Results of LCCspm (Time-Optimised) further reinforced the efficiency, effectiveness and applicability of LCCspm to extract patterns from a set of sequences with a faster execution runtime. The LCCspm (Time-Optimised), however, consumes more computer memory for extracting patterns. Both LCCspm (Memory and Time - Optimised) variants do not fail to extract user-defined lengths of frequent closed contiguous patterns given any support parameter. Both LCCspm algorithms, developed by this PhD study, outperformed the state-of-the-art (i.e., CCSpan) algorithm (Zhang et al., 2015) for frequent closed contiguous sequential pattern mining in terms of scalability and execution runtime while providing more functionality.

The best type of movement patterns useful for player movement profiling (in the context of player position separation) was identified in Chapter 5 through an experimental process. The sequential but non-consecutive movement patterns of the existing SMP framework (White et al., 2021) and the frequent closed non-consecutive and non-sequential movement patterns of AprioriClose algorithm (Pasquier et al., 1999) were compared against the sequential and consecutive movement patterns of LCCspm (with respect to this PhD study's third objective) to profile elite RFL players into two rugby league playing positions (i.e. hookers and wingers) known to be tactically different. Based on obtained results from three step-wise analyses, LCCspm was validated as the algorithm that produced the best type of movement patterns to profile players' movement.

Consequently, in Chapter 6, practical applications of LCCspm movement patterns (with respect to this PhD study's fourth objective) to identify sets of signature movement patterns of each RFL playing position, classify players into nine playing positions and establish "Movement Performance Indicators" across RFL playing positions was carried out. These practical applications are immensely useful for talent identification and recruitment (Vaeyens et al., 2008; Williams and Reilly, 2000), training customisation and players' pathway development (Adeyemo et al., 2022; Whitehead et al.,

2021) and assessment of players' performance variability (Hrovat et al., 2015) among others. Varying numbers of signature movement patterns were identified across playing positions, with centres having the highest number of signature patterns (i.e. 1241). Classification of RFL players into playing positions can be accurately carried out at 73.41% using a Random Forest model with 7,167 movement patterns as predictor variables. However, a Logistic Regression model was developed using only 72 movement patterns as predictor variables, with a classification accuracy of 67.9% to classify RFL players into playing positions. The relative frequency of movement patterns is the appropriate value useful to complete such classification analysis. Also, a set of 16 "Movement Performance Indicators" were established and used to assess players' performance variability. Each player's performance differs based on each Movement Performance Indicator. Also, players belonging to different playing positions recorded different rates of completing the external load represented by a Movement Performance Indicator.

7.1 Practical Implications

The research findings offer valuable insights and applications that can benefit practitioners and the sports industry as a whole. The practical implications of this study are highlighted as follows:

1. Advancements in Computing and Sports Analytics:

- The LCCspm algorithm and the introduced techniques contribute to the advancement of pattern mining algorithms and sports analytics.
- The LCCspm algorithm can be applied in other sports domains, allowing for more accurate analysis and insights into player behaviour and team dynamics.

2. Tactical Decision-Making and Team Composition:

- The successful application of LCCspm in classifying rugby league players into tactically different positions provides a valuable tool for team composition decisions.

- Coaches and team management can leverage this information to optimize team strategies, match player strengths with specific positions, and enhance overall team performance.
3. Signature Movement Patterns and Game Understanding:
 - The identification of signature movement patterns specific to rugby league players enhances the understanding of player behaviour within the game.
 - Practitioners can use this knowledge to develop targeted training programs, improve tactical decision-making, and enhance team performance.
 4. Enhanced Player Evaluation and Performance Optimization:
 - The LCCspm algorithm and the developed Movement Performance Indicators (MPIs) enable a comprehensive assessment of players' performance variability.
 - Practitioners can utilize these insights to identify areas of improvement, optimize training programs, and tailor individualized coaching strategies leading to more informed strategies and player management.
 5. Generalization to Other Domains:
 - The insights and methodologies gained from this research can be extended beyond rugby league and applied to other sports or domains where the sequential quantification of activities plays a crucial role.
 - This widens the potential impact of the study in areas such as team sports, individual athletics, and human motion analysis.

The findings provide actionable insights for coaches, trainers, analysts, and the sports industry, ultimately leading to improved performance outcomes and a deeper understanding of player behaviour in competitive sports.

7.2 Summary of Contributions

1. This PhD study contributes to the body of knowledge through the proposal, formulation, development, optimisation and performance evaluation of a novel pat-

tern mining algorithm that serves player movement profiling. This fulfilled the second objective of this PhD study and also make a contribution to the body of knowledge.

2. This PhD study also contributes to the body of knowledge through the identification of (and thus validates) the best pattern mining algorithm for player movement profiling in the context of playing position. This fulfilled the third objective of this PhD study.
3. This PhD study conducted the first attempt to classify elite rugby league players' into playing positions based on movement patterns as predictor variables, ascertained the appropriate predictor variable values and identified key movement patterns useful for such classification. This partially fulfilled the fourth objective of this PhD study and also make a contribution to the body of knowledge.
4. The discovery of signature movement patterns of each elite rugby league playing position (i.e., only performed by players belonging to a specific playing position) is also among the significant contributions of this PhD study. This partially fulfilled the fourth objective of this PhD study
5. Also, the identification of a set of significant movement patterns as "Movement Performance Indicators" and its use to assess and visualise players' performance variability over participated competitive matches is also considered to be another contribution of this PhD study. This partially fulfilled the fourth objective of this PhD study.

7.3 Limitations and Future Works

The greatest limitation of identifying frequent movement patterns from rugby league GPS data is that external loads of players per match were represented as a set of discrete movement sequences. This representation affects the computed frequency of extracted movement patterns. An ideal solution is to represent a set of discrete movement sequences of players per match as a single sequence. This will affect the support of extracted movement patterns and will influence the movement patterns identified as frequent for various applications.

Another limitation of the PhD study design is that the lowest granularity for analysing GPS data (i.e. player per fixture level) was considered. However, other granularity such as player level or playing position level (per match or across matches) can be considered which may yield different results.

The optimised variant of LCCspm is limited in its consumption of large computer memory. Although computer memory is becoming a disposable resource in this modern time, the implementation of another efficient data structure to store the vertical representation of candidate patterns and index pairs should be considered in the future. Also, other interestingness measures can be integrated into both LCCspm and its optimised variant algorithms in addition to the frequency and maximal length interestingness measures.

It would be interesting to apply LCCspm to discover movement patterns leading to injuries in rugby league, such as non-contact and soft tissue, hamstring and or anterior cruciate ligament (ACL) injuries. Rugby league is an intense contact team sport where players experience high levels of physical stress. A body of a typical elite rugby league player undergoes and responds to enormous stress and fatigue. Current research on injury prevention or detection in sports at large considers aggregated physical indicators (over a period) to assess injury in players. The use of movement patterns to assess injury risk in players will provide a more detailed granular level of information for injury risk assessment, injury prediction and or detection.

This PhD study focused more on player movement profiling based on GPS data, however, match events data are also collected during competitive matches. It would equally be interesting to analyse match events data with respect to the sequential nature of match events. This can be useful for technical-tactical analysis such as opposition analysis or attacking tactics analysis, especially in a team or individual sports where sequential order of match events is important.

References

- Rugby league playing positions. <https://www.usarl.org/development/the-rules-of-rugby-league/>. [Online; accessed 22-November-2022].
- <https://statsbomb.com/resource-centre/>. [Online; accessed 01-October-2021].
- l-length frequent contiguous subsequence (i.e., lfcs) explorer. <https://share.streamlit.io/arhvel/lfcs/main/LFCSstreamlit.py>, 2020. [Online; accessed 27-January-2023].
- Random forest. <https://dinhanhthi.com/random-forest/>, 2020. [Online; accessed 22-June-2022].
- Yacine Abboud, Anne Boyer, and Armelle Brun. Ccpm: a scalable and noise-resistant closed contiguous sequential patterns mining algorithm. In *International Conference on Machine Learning and Data Mining in Pattern Recognition*, pages 147–162. Springer, 2017.
- Victor E Adeyemo, Abdullateef O Balogun, Hammed A Mojeed, Noah O Akande, and Kayode S Adewole. Ensemble-based logistic model trees for website phishing detection. In *International Conference on Advances in Cyber Security*, pages 627–641. Springer, 2020.
- Victor Elijah Adeyemo. l-length closed contiguous sequential pattern mining algorithm. <https://github.com/arhvel/LCCspm/>, 2021a. [Online; accessed 22-June-2021].

REFERENCES

- Victor Elijah Adeyemo. l-Length Closed Contiguous Sequential Pattern Mining Algorithm. <https://github.com/arhvel/MovementPatternsForClassification>, 2021b. [Online; accessed 24-August-2022].
- Victor Elijah Adeyemo. Own figures and tables, 2023.
- Victor Elijah Adeyemo, Anna Palczewska, and Ben Jones. Lccspm: l-length closed contiguous sequential patterns mining algorithm to find frequent athlete movement patterns from gps. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 455–460. IEEE, 2021.
- Victor Elijah Adeyemo, Anna Palczewska, Ben Jones, Dan Weaving, and Sarah Whitehead. Optimising classification in sport: a replication study using physical and technical-tactical performance indicators to classify competitive levels in rugby league match-play. *Science and Medicine in Football*, pages 1–8, 2022.
- Victor Elijah Adeyemo, Anna Palczewska, Ben Jones, and Dan Weaving. Identification of pattern mining algorithm for rugby league players positional groups separation based on movement patterns. *arXiv preprint arXiv:2302.14058*, 2023.
- Rakesh Agrawal and Ramakrishnan Srikant. Mining sequential patterns. In *Proceedings of the eleventh international conference on data engineering*, pages 3–14. IEEE, 1995.
- Rakesh Agrawal, Ramakrishnan Srikant, et al. Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB*, volume 1215, pages 487–499. Citeseer, 1994.
- Aqib Ali, Salman Qadri, Wali Khan Mashwani, Samir Brahim Belhaouari, Samreen Naeem, Sidra Rafique, Farrukh Jamal, Christophe Chesneau, and Sania Anam. Machine learning approach for the classification of corn seed using hybrid features. *International Journal of Food Properties*, 23(1):1110–1124, 2020.
- Damien J Austin and Stephen J Kelly. Positional differences in professional rugby league match play through the use of global positioning systems. *The Journal of Strength & Conditioning Research*, 27(1):14–19, 2013.

REFERENCES

- Francisco Ayala, Alejandro López-Valenciano, Jose Antonio Gámez Martín, Mark De Ste Croix, Francisco J Vera-Garcia, Maria del Pilar Garcia-Vaquero, Iñaki Ruiz-Pérez, and Gregory D Myer. A preventive model for hamstring injuries in professional soccer: learning algorithms. *International journal of sports medicine*, 40(05): 344–353, 2019.
- Jay Ayres, Jason Flannick, Johannes Gehrke, and Tomi Yiu. Sequential pattern mining using a bitmap representation. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 429–435, 2002.
- ABDULLATEEF O Balogun, SHUIB Basri, SAID J Abdulkadir, VICTOR E Adeyemo, ABDULLAHI A Imam, and AMOS O Bajeh. Software defect prediction: analysis of class imbalance and performance stability. *J. Eng. Sci. Technol*, 14 (6):3294–3308, 2019.
- Abdullateef O Balogun, Shuib Basri, Saipunidzam Mahamad, Said J Abdulkadir, Malek A Almomani, Victor E Adeyemo, Qasem Al-Tashi, Hammed A Mojeed, Abdullahi A Imam, and Amos O Bajeh. Impact of feature selection methods on the predictive performance of software defect prediction models: an extensive empirical study. *Symmetry*, 12(7):1147, 2020.
- Gary Benson, Avivit Levy, and B Riva Shalom. Longest common subsequence in k length substrings. In *International Conference on Similarity Search and Applications*, pages 257–265. Springer, 2013.
- Luke Bermingham and Ickjai Lee. Mining distinct and contiguous sequential patterns from large vehicle trajectories. *Knowledge-Based Systems*, 189:105076, 2020.
- Shashank Bhargav, Shruti Kaushik, Varun Dutt, et al. A combination of decision trees with machine learning ensembles for blood glucose level predictions. In *Proceedings of International Conference on Data Science and Applications*, pages 533–548. Springer, 2022.
- Adel Binbusayyis and Thavavel Vaiyapuri. Identifying and benchmarking key features for cyber intrusion detection: an ensemble approach. *IEEE Access*, 7:106495–106513, 2019.

REFERENCES

- Paul S Bradley and Jack D Ade. Are current physical match performance metrics in elite soccer fit for purpose or is the adoption of an integrated approach needed? *International Journal of Sports Physiology and Performance*, 13(5):656–664, 2018.
- John Brewer and Jackie Davis. Applied physiology of rugby league. *Sports Medicine*, 20(3):129–135, 1995.
- Rory Bunker, Keisuke Fujii, Hiroyuki Hanada, and Ichiro Takeuchi. Supervised sequential pattern mining of event sequences in sport to identify important patterns of play: an application to rugby union. *PloS one*, 16(9):e0256329, 2021.
- Peter Calhoun, Richard A Levine, and Juanjuan Fan. Repeated measures random forests (rmrf): Identifying factors associated with nocturnal hypoglycemia. *Biometrics*, 77(1):343–351, 2021.
- Chih-Wen Chen, Yi-Hong Tsai, Fang-Rong Chang, and Wei-Chao Lin. Ensemble feature selection in medical datasets: Combining filter, wrapper, and embedded feature selection results. *Expert Systems*, 37(5):e12553, 2020a.
- Shenglei Chen, Geoffrey I Webb, Linyuan Liu, and Xin Ma. A novel selective naïve bayes algorithm. *Knowledge-Based Systems*, 192:105361, 2020b.
- Neil Collins, Ryan White, Anna Palczewska, Dan Weaving, Nick Dalton-Barron, and Ben Jones. Moving beyond velocity derivatives; using global positioning system data to extract sequential movement patterns at different levels of rugby league match-play. *European Journal of Sport Science*, (just-accepted):1–23, 2022.
- Carmen ME Colomer, David B Pyne, Mitch Mooney, Andrew McKune, and Benjamin G Serpell. Performance analysis in rugby union: a critical systematic review. *Sports Medicine-Open*, 6(1):1–15, 2020.
- Nicholas Dalton-Barron, Sarah Whitehead, Gregory Roe, Cloe Cummins, Clive Beggs, and Ben Jones. Time to embrace the complexity when analysing gps data? a systematic review of contextual factors on match running in rugby league. *Journal of sports sciences*, 38(10):1161–1180, 2020.

REFERENCES

- Nicholas Dalton-Barron, Anna Palczewska, Shaun J McLaren, Gordon Rennie, Clive Beggs, Gregory Roe, and Ben Jones. A league-wide investigation into variability of rugby league match running from 322 super league games. *Science and Medicine in Football*, 5(3):225–233, 2021.
- Tom Decroos, Jan Van Haaren, and Jesse Davis. Automatic discovery of tactics in spatio-temporal soccer match data. In *Proceedings of the 24th acm sigkdd international conference on knowledge discovery & data mining*, pages 223–232, 2018.
- Carolyn A Emery and Kati Pasanen. Current trends in sport injury prevention. *Best Practice & Research Clinical Rheumatology*, 33(1):3–15, 2019.
- Syeda Farzana Zerin and Byeong-Soo Jeong. A fast contiguous sequential pattern mining technique in dna data sequences using position information. *IETE Technical Review*, 28(6):511–519, 2011.
- Philippe Fournier-Viger, Antonio Gomariz, Ted Gueniche, Espérance Mwamikazi, and Rincy Thomas. Tks: efficient mining of top-k sequential patterns. In *International Conference on Advanced Data Mining and Applications*, pages 109–120. Springer, 2013a.
- Philippe Fournier-Viger, Cheng-Wei Wu, and Vincent S Tseng. Mining maximal sequential patterns without candidate maintenance. In *International Conference on Advanced Data Mining and Applications*, pages 169–180. Springer, 2013b.
- Philippe Fournier-Viger, Antonio Gomariz, Manuel Campos, and Rincy Thomas. Fast vertical mining of sequential patterns using co-occurrence information. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 40–52. Springer, 2014a.
- Philippe Fournier-Viger, Cheng-Wei Wu, Antonio Gomariz, and Vincent S Tseng. Vmsp: Efficient vertical mining of maximal sequential patterns. In *Canadian conference on artificial intelligence*, pages 83–94. Springer, 2014b.
- Philippe Fournier-Viger, Jerry Chun-Wei Lin, Rage Uday Kiran, Yun Sing Koh, and Rincy Thomas. A survey of sequential pattern mining. *Data Science and Pattern Recognition*, 1(1):54–77, 2017.

REFERENCES

- Dmitriy Fradkin and Fabian Mörchen. Mining sequential patterns for classification. *Knowledge and Information Systems*, 45(3):731–749, 2015.
- Fabio Fumarola, Pasqua Fabiana Lanotte, Michelangelo Ceci, and Donato Malerba. Clofast: closed sequential pattern mining using sparse and vertical id-lists. *Knowledge and Information Systems*, 48(2):429–463, 2016.
- Tim Gabbett, David Jenkins, and Bruce Abernethy. Physical collisions and injury during professional rugby league skills training. *Journal of Science and Medicine in Sport*, 13(6):578–583, 2010.
- Tim J Gabbett. Incidence, site, and nature of injuries in amateur rugby league over three consecutive seasons. *British journal of sports medicine*, 34(2):98–103, 2000.
- Tim J Gabbett. Training injuries in rugby league: an evaluation of skill-based conditioning games. *The Journal of Strength & Conditioning Research*, 16(2):236–241, 2002.
- Tim J Gabbett. Science of rugby league football: a review. *Journal of sports sciences*, 23(9):961–976, 2005.
- Tim J Gabbett. Influence of playing standard on the physical demands of professional rugby league. *Journal of Sports Sciences*, 31(10):1125–1138, 2013.
- Tim J Gabbett, David G Jenkins, and Bruce Abernethy. Relationships between physiological, anthropometric, and skill qualities and playing performance in professional rugby league players. *Journal of sports sciences*, 29(15):1655–1664, 2011.
- Tim J Gabbett, David G Jenkins, and Bruce Abernethy. Physical demands of professional rugby league training and competition using microtechnology. *Journal of science and medicine in sport*, 15(1):80–86, 2012.
- Chuancong Gao, Jianyong Wang, Yukai He, and Lizhu Zhou. Efficient mining of frequent sequence generators. In *Proceedings of the 17th international conference on World Wide Web*, pages 1051–1052, 2008.

REFERENCES

- Andrew Gardner, Grant L Iverson, Christopher R Levi, Peter W Schofield, Frances Kay-Lambkin, Ryan MN Kohler, and Peter Stanwell. A systematic review of concussion in rugby league. *British journal of sports medicine*, 49(8):495–498, 2015.
- Júlio Garganta. Trends of tactical performance analysis in team sports: bridging the gap between research, training and competition. *Revista Portuguesa de Ciências do desporto*, 9(1), 2009.
- Tom Geeson-Brown, Ben Jones, Kevin Till, Sarah Chantler, and Kevin Deighton. Body composition differences by age and playing standard in male rugby union and rugby league: A systematic review and meta-analysis. *Journal of sports sciences*, 38(19): 2161–2176, 2020.
- Daniel J Glassbrook, Tim LA Doyle, Jacqueline A Alderson, and Joel T Fuller. The demands of professional rugby league match-play: a meta-analysis. *Sports medicine-open*, 5(1):1–20, 2019.
- FR Goes, LA Meerhoff, MJO Bueno, DM Rodrigues, FA Moura, MS Brink, MT Elferink-Gemser, AJ Knobbe, SA Cunha, RS Torres, et al. Unlocking the potential of big data to support tactical performance analysis in professional soccer: A systematic review. *European Journal of Sport Science*, 21(4):481–496, 2021.
- Antonio Gomariz, Manuel Campos, Roque Marin, and Bart Goethals. Clasp: An efficient algorithm for mining frequent closed sequences. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 50–61. Springer, 2013.
- Sini Guo, Xiang Li, Wai-Ki Ching, Ralescu Dan, Wai-Keung Li, and Zhiwen Zhang. Gps trajectory data segmentation based on probabilistic logic. *International Journal of Approximate Reasoning*, 103:227–247, 2018.
- Mark A Hall. *Correlation-based feature selection for machine learning*. PhD thesis, The University of Waikato, 1999.
- Shona L Halson, Rich D Johnston, Renee N Appaneal, Margot A Rogers, Liam A Toohey, Michael K Drew, Charli Sargent, and Gregory D Roach. Sleep quality in elite athletes: Normative values, reliability and understanding contributors to poor sleep. *Sports Medicine*, 52(2):417–426, 2022.

REFERENCES

- Jiawei Han, Jian Pei, Behzad Mortazavi-Asl, Qiming Chen, Umeshwar Dayal, and Mei-Chun Hsu. Freespan: frequent pattern-projected sequential pattern mining. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 355–359, 2000.
- Jiawei Han, Jian Pei, Behzad Mortazavi-Asl, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Meichun Hsu. Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *proceedings of the 17th international conference on data engineering*, pages 215–224. Citeseer, 2001.
- Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data mining and knowledge discovery*, 8(1):53–87, 2004.
- Goran Hrovat, Iztok Fister Jr, Katsiaryna Yermak, Gregor Stiglic, and Iztok Fister. Interestingness measure for mining sequential patterns in sports. *Journal of Intelligent & Fuzzy Systems*, 29(5):1981–1994, 2015.
- Mike Hughes and Ian M Franks. The essentials of performance analysis. *London: E. & FN Spon*, 2008.
- Franco M Impellizzeri, Ermanno Rampinini, and Samuele M Marcora. Physiological assessment of aerobic training in soccer. *Journal of sports sciences*, 23(6):583–592, 2005.
- Md Islam, CM Saifullah, Zarrin Tasnim Asha, Rezoana Ahamed, et al. Chemical reaction optimization for solving longest common subsequence problem for multiple string. *Soft Computing*, 23(14):5485–5509, 2019.
- Jingquan Jin and Xin Lin. Web log analysis and security assessment method based on data mining. *Computational Intelligence and Neuroscience*, 2022, 2022.
- Rich D Johnston, Tim J Gabbett, and David G Jenkins. Applied sport science of rugby league. *Sports medicine*, 44(8):1087–1100, 2014.
- Md Rezaul Karim, Md Mamunur Rashid, Byeong-Soo Jeong, and Ho-Jin Choi. An efficient approach to mining maximal contiguous frequent patterns from large dna sequence databases. *Genomics & informatics*, 10(1):51–57, 2012.

- Thomas Kempton, Anita C Sirotic, and Aaron J Coutts. A comparison of physical and technical performance profiles between successful and less-successful professional rugby league teams. *International journal of sports physiology and performance*, 12(4):520–526, 2017.
- Shufen Kuo and George R. Cross. An improved algorithm to find the length of the longest common subsequence of two strings. In *ACM Sigir Forum*, volume 23, pages 89–99. ACM New York, NY, USA, 1989.
- Natalie Kupperman and Jay Hertel. Global positioning system–derived workload metrics and injury risk in team-based field sports: a systematic review. *Journal of athletic training*, 55(9):931–943, 2020.
- Hoang Thanh Lam, Fabian Mörchén, Dmitriy Fradkin, and Toon Calders. Mining compressing sequential patterns. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 7(1):34–52, 2014.
- Guillaume Lemaître, Fernando Nogueira, and Christos K Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *The Journal of Machine Learning Research*, 18(1):559–563, 2017.
- Vladimir I Levenshtein et al. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union, 1966.
- Hua-Fu Li. A sliding window method for finding top-k path traversal patterns over streaming web click-sequences. *Expert Systems with Applications*, 36(3):4382–4386, 2009.
- Huan Liu, Rudy Setiono, et al. A probabilistic approach to feature selection-a filter solution. In *ICML*, volume 96, pages 319–327. Citeseer, 1996.
- David Lo, Siau-Cheng Khoo, and Jinyan Li. Mining and ranking generators of sequential patterns. In *Proceedings of the 2008 SIAM International Conference on Data Mining*, pages 553–564. SIAM, 2008.
- S Lu and C Li. Aprioriadjust: An efficient algorithm for discovering the maximum sequential patterns. In *Proc. Intern. Workshop knowl. Grid and grid intell*, 2004.

REFERENCES

- Claudio Lucchese, Salvatore Orlando, and Raffaele Perego. Fast and memory efficient mining of frequent closed itemsets. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):21–36, 2005.
- Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.
- MA Mabayoje, AO Balogun, AO Ameen, and VE Adeyemo. Influence of feature selection on multi-layer perceptron classifier for intrusion detection system. *Computing, Information System Development Informatics & Allied Research Journals*, 7(4):87–94, 2016.
- Nizar R Mabroukeh and Christie I Ezeife. A taxonomy of sequential pattern mining algorithms. *ACM Computing Surveys (CSUR)*, 43(1):1–41, 2010.
- João Cláudio Machado, João Ribeiro, Carlos Ewerton Palheta, Chellsea Alcântara, Daniel Barreira, José Guilherme, Júlio Garganta, and Alcides José Scaglia. Changing rules and configurations during soccer small-sided and conditioned games. how does it impact teams’ tactical behavior? *Frontiers in Psychology*, 10:1554, 2019.
- Christina D Mack, Peter Meisel, Mackenzie M Herzog, Lisa Callahan, Eva E Oakkar, Taylor Walden, Joseph Sharpe, Nancy A Dreyer, and John DiFiori. The establishment and refinement of the national basketball association player injury and illness database. *Journal of athletic training*, 54(5):466–471, 2019.
- Rudi Meir, Robert Newton, Edgar Curtis, Matthew Fardell, and Benjamin Butler. Physical fitness qualities of professional rugby league football players: determination of positional differences. *The Journal of Strength & Conditioning Research*, 15(4):450–458, 2001.
- Carl H Mooney and John F Roddick. Sequential pattern mining—approaches and algorithms. *ACM Computing Surveys (CSUR)*, 45(2):1–39, 2013.
- Elia Morgulev, Ofer H Azar, and Ronnie Lidor. Sports analytics and the big-data era. *International Journal of Data Science and Analytics*, 5(4):213–222, 2018.

REFERENCES

- Siegfried Nijssen. *Pattern Mining*, pages 1661–1666. Springer New York, New York, NY, 2013. ISBN 978-1-4419-9863-7. doi: 10.1007/978-1-4419-9863-7_600. URL https://doi.org/10.1007/978-1-4419-9863-7_600.
- Victor Chazan Pantzalis and Christos Tjortjis. Sports analytics for football league table and player performance prediction. In *2020 11th International Conference on Information, Intelligence, Systems and Applications (IISA)*, pages 1–8. IEEE, 2020.
- Adrian Ionut Pascu. Data mining. concepts and applications in banking sector. *Annals of 'Constantin Brancusi' University of Targu-Jiu. Economy Series*, (1), 2018.
- Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal. Discovering frequent closed itemsets for association rules. In *International Conference on Database Theory*, pages 398–416. Springer, 1999.
- Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.
- Charles Perin, Romain Vuillemot, Charles D Stolper, John T Stasko, Jo Wood, and Sheelagh Carpendale. State of the art of sports data visualization. In *Computer Graphics Forum*, volume 37, pages 663–686. Wiley Online Library, 2018.
- Thi-Thiet Pham, Jiawei Luo, Tzung-Pei Hong, and Bay Vo. Msgps: a novel algorithm for mining sequential generator patterns. In *International Conference on Computational Collective Intelligence*, pages 393–401. Springer, 2012.
- Johan Pion, Andreas Hohmann, Tianbiao Liu, Matthieu Lenoir, and Veerle Segers. Predictive models reduce talent development costs in female gymnastics. *Journal of sports sciences*, 35(8):806–811, 2017.
- David Pizarro, Alba Práxedes, Bruno Travassos, Fernando del Villar, and Alberto Moreno. The effects of a nonlinear pedagogy training program in the technical-tactical behaviour of youth futsal players. *International Journal of Sports Science & Coaching*, 14(1):15–23, 2019.

REFERENCES

- Huseyin Polat, Hoday Danaei Mehr, and Aydin Cetin. Diagnosis of chronic kidney disease based on support vector machine by feature selection methods. *Journal of medical systems*, 41(4):1–11, 2017.
- Wei Qu, Yuanyuan Jia, and Michael Jiang. Pattern mining of cloned codes in software systems. *Information Sciences*, 259:544–554, 2014.
- Md Mamunur Rashid, Md Rezaul Karim, Byeong-Soo Jeong, and Ho-Jin Choi. Efficient mining of interesting patterns in large biological sequences. *Genomics & informatics*, 10(1):44, 2012.
- Nazim Razali, Aida Mustapha, Faiz Ahmad Yatim, and Ruhaya Ab Aziz. Predicting player position for talent identification in association football. In *IOP Conference Series: Materials Science and Engineering*, volume 226, page 012087. IOP Publishing, 2017.
- Dale Read, Daniel Weaving, Padraic Phibbs, Joshua Darrall-Jones, Gregory Roe, Jonathon Weakley, Sharief Hendricks, Kevin Till, and Ben Jones. Movement and physical demands of school and university rugby union match-play in england. *BMJ open sport & exercise medicine*, 2(1):e000147, 2017.
- Ian Renshaw and Jia-Yi Chow. A constraint-led approach to sport and physical education pedagogy. *Physical Education and Sport Pedagogy*, 24(2):103–116, 2019.
- Thuraiappah Sathyan, Richard Shuttleworth, Mark Hedley, and Keith Davids. Validity and reliability of a radio positioning system for tracking athletes in indoor and outdoor team sports. *Behavior research methods*, 44(4):1108–1114, 2012.
- Agnese Sbröllini, Micaela Morettini, Elvira Maranesi, Ilaria Marcantoni, Amnah Nasim, Roberta Bevilacqua, Giovanni R Riccardi, and Laura Burattini. Sport database: Cardiorespiratory data acquired through wearable sensors while practicing sports. *Data in brief*, 27:104793, 2019.
- Jivitesh Sharma, Charul Giri, Ole-Christoffer Granmo, and Morten Goodwin. Multi-layer intrusion detection system with extratrees feature selection, extreme learning machine ensemble, and softmax aggregation. *EURASIP Journal on Information Security*, 2019(1):1–16, 2019.

REFERENCES

- Kelvin Sim, Jinyan Li, Vivekanand Gopalkrishnan, and Guimei Liu. Mining maximal quasi-bicliques: Novel algorithm and applications in the stock market and protein networks. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 2 (4):255–273, 2009.
- Ramakrishnan Srikant and Rakesh Agrawal. Mining sequential patterns: Generalizations and performance improvements. In *International conference on extending database technology*, pages 1–17. Springer, 1996.
- Alice J Sweeting, Stuart Morgan, Stuart Cormack, and Robert J Aughey. A movement sequencing analysis of team-sport athlete match activity profile. 2014.
- Alice J. Sweeting, Robert J. Aughey, Stuart J. Cormack, and Stuart Morgan. Discovering frequently recurring movement sequences in team-sport athlete spatiotemporal data. *Journal of Sports Sciences*, 35(24):2439–2445, 2017. doi: 10.1080/02640414.2016.1273536. URL <https://doi.org/10.1080/02640414.2016.1273536>. PMID: 28282752.
- Heidi R Thornton, Jace A Delaney, Grant M Duthie, and Ben J Dascombe. Importance of various training-load measures in injury incidence of professional rugby league athletes. *International journal of sports physiology and performance*, 12(6):819–824, 2017.
- Kevin Till, Stephen Copley, John O’Hara, Chris Chapman, and Carlton Cooke. A longitudinal evaluation of anthropometric and fitness characteristics in junior rugby league players considering playing position and selection level. *Journal of Science and Medicine in Sport*, 16(5):438–443, 2013.
- Kevin Till, Sean Scantlebury, and Ben Jones. Anthropometric and physical qualities of elite male youth rugby league players. *Sports Medicine*, 47(11):2171–2186, 2017.
- Takeaki Uno, Masashi Kiyomi, Hiroki Arimura, et al. Lcm ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets. In *Fimi*, volume 126, 2004.
- Roel Vaeyens, Matthieu Lenoir, A Mark Williams, and Renaat M Philippaerts. Talent identification and development programmes in sport. *Sports medicine*, 38(9):703–714, 2008.

REFERENCES

- Hans Van Eetvelde, Luciana D Mendonça, Christophe Ley, Romain Seil, and Thomas Tischer. Machine learning methods in sport injury prediction and prevention: a systematic review. *Journal of experimental orthopaedics*, 8(1):1–15, 2021.
- Mark Waldron, Craig Twist, Jamie Highton, Paul Worsfold, and Matthew Daniels. Movement and physiological match demands of elite rugby league using portable global positioning systems. *Journal of sports sciences*, 29(11):1223–1230, 2011.
- Jianyong Wang and Jiawei Han. Bide: Efficient mining of frequent closed sequences. In *Proceedings. 20th international conference on data engineering*, pages 79–90. IEEE, 2004.
- Jianyong Wang, Jiawei Han, and Chun Li. Frequent closed sequence mining without candidate maintenance. *IEEE Transactions on Knowledge and Data Engineering*, 19(8):1042–1056, 2007.
- Zheng Wang, Cheng Long, Gao Cong, and Ce Ju. Effective and efficient sports play retrieval with deep representation learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 499–509, 2019.
- Joe H Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.
- Dan Weaving, Thomas Sawczuk, Sean Williams, Tannath Scott, Kevin Till, Clive Beggs, Rich D Johnston, and Ben Jones. The peak duration-specific locomotor demands and concurrent collision frequencies of european super league rugby. *Journal of sports sciences*, 37(3):322–330, 2019.
- Dan Weaving, Damien Young, Andrea Riboli, Ben Jones, and Giuseppe Coratella. The maximal intensity period: Rationalising its use in team sports practice. *Sports Medicine-Open*, 8(1):1–9, 2022.
- Ryan White, Anna Palczewska, Dan Weaving, Neil Collins, and Ben Jones. Sequential movement pattern-mining (smp) in field-based team-sport: A framework for quantifying spatiotemporal data and improve training specificity? *Journal of Sports Sciences*, pages 1–10, 2021.

REFERENCES

- Sarah Whitehead, Kevin Till, Dan Weaving, and Ben Jones. The use of microtechnology to quantify the peak match demands of the football codes: a systematic review. *Sports medicine*, 48(11):2549–2575, 2018.
- Sarah Whitehead, Kevin Till, Dan Weaving, Richard Hunwicks, Rob Pacey, and Ben Jones. Whole, half and peak running demands during club and international youth rugby league match-play. *Science and Medicine in Football*, 3(1):63–69, 2019.
- Sarah Whitehead, Kevin Till, Ben Jones, Clive Beggs, Nicholas Dalton-Barron, and Dan Weaving. The use of technical-tactical and physical performance indicators to classify between levels of match-play in elite rugby league. *Science and Medicine in Football*, 5(2):121–127, 2021.
- David Whiteside, Douglas N Martini, Adam S Lepley, Ronald F Zernicke, and Grant C Goulet. Predictors of ulnar collateral ligament reconstruction in major league baseball pitchers. *The American journal of sports medicine*, 44(9):2202–2209, 2016.
- Sascha Wilkens. Sports prediction and betting models in the machine learning age: The case of tennis. *Journal of Sports Analytics*, 7(2):99–117, 2021.
- A Mark Williams and Tom Reilly. Talent identification and development in soccer. *Journal of sports sciences*, 18(9):657–667, 2000.
- Brian D Williamson, Peter B Gilbert, Marco Carone, and Noah Simon. Nonparametric variable importance assessment using machine learning techniques. *Biometrics*, 77(1):9–22, 2021.
- Ian H Witten and Eibe Frank. Data mining: practical machine learning tools and techniques with java implementations. *Acm Sigmod Record*, 31(1):76–77, 2002.
- Carl T Woods, Anthony S Leicht, Ben Jones, and Kevin Till. Game-play characteristics differ between the european super league and the national rugby league: implications for coaching and talent recruitment. *International Journal of Sports Science & Coaching*, 13(6):1171–1176, 2018a.
- Carl T Woods, Sam Robertson, Wade H Sinclair, Kevin Till, Leesa Pearce, and Anthony S Leicht. A comparison of game-play characteristics between elite youth

REFERENCES

- and senior australian national rugby league competitions. *Journal of Science and Medicine in Sport*, 21(6):626–630, 2018b.
- Carl T Woods, James Veale, Job Fransen, Sam Robertson, and Neil French Collier. Classification of playing position in elite junior australian football using technical skill indicators. *Journal of sports sciences*, 36(1):97–103, 2018c.
- Xifeng Yan, Jiawei Han, and Ramin Afshar. Clospan: Mining: Closed sequential patterns in large datasets. In *Proceedings of the 2003 SIAM international conference on data mining*, pages 166–177. SIAM, 2003.
- Shengwei Yi, Tianheng Zhao, Yuanyuan Zhang, Shilong Ma, and Zhanbin Che. An effective algorithm for mining sequential generators. *Procedia Engineering*, 15: 3653–3657, 2011.
- Mohammed J Zaki. Spade: An efficient algorithm for mining frequent sequences. *Machine learning*, 42(1):31–60, 2001.
- Mohammed J Zaki and Ching-Jui Hsiao. Charm: An efficient algorithm for closed itemset mining. In *Proceedings of the 2002 SIAM international conference on data mining*, pages 457–473. SIAM, 2002.
- Jingsong Zhang, Yinglin Wang, and Dingyu Yang. Ccspan: Mining closed contiguous sequential patterns. *Knowledge-Based Systems*, 89:1–13, 2015.
- Qiankun Zhao and Sourav S Bhowmick. Sequential pattern mining: A survey. *ITechnical Report CAIS Nanyang Technological University Singapore*, 1(26):135, 2003.

Appendix A

Ethical Approval and Informed Consent

07/01/2021

Research Ethics Online

A A A A

Research Ethics Online

[New Application](#)
[My Applications](#)

[Logout](#)

My Applications

New Application

If you wish to submit a new application, click on 'New Applications' above.

Existing applications

If you wish to edit an existing application prior to submission, click on the Application Title or select the 'Edit/Continue' button.

If you have submitted an application and now need to make changes to it, click on the 'Make Revision/Copy' button. Please add to the title the version number (for example, v2).

10 records per page

Search:

Title	Risk Category	Status	Date Created	Action
Data Management, Mining and Advanced Analytics: An Intensive Application for Sports Performance	Risk Category 2	Approved by LREC	30-JUL-20	<div></div> <div></div> <div></div>
Data Management, Mining, and Analytics: An Intensive Application for Sports Performance	Risk Category 2	Approved by LREC	28-JUL-20	<div></div> <div></div> <div></div>

Showing 1 to 2 of 2 entries

← Previous

1

Next →

Contact Us

+44 (0)113 812 000

Find Us

Leeds Beckett University,
City Campus,
Leeds, LS1 3HE

Connect with Us

f

YouTube

Twitter

in

Contact Us | Disclaimer | Accessibility | Privacy

© Copyright Leeds Beckett University 2018

Figure A.1: University Ethical Approval

Gate Keeper Consent Form

Researcher's Name	ADEYEMO, Victor Elijah [REDACTED]
University	School of Built Environment, Engineering and Computing Leeds Beckett University
Course	PhD
Project Title	Data Management, Mining, and Analytics: An Intensive Application for Sports Performance
Director of Studies	Dr Anna Palczewska [REDACTED]
Supervisor	Dr Abdulrahman Altahhan [REDACTED]
Advisor	Prof Ben Jones [REDACTED]

To: Dave Rotheram,
Chief On-Field Officer,
Rugby Football League (RFL).

This consent form serves as a medium for the researcher to request access, store and make use of the data owned and managed by your organisation for the above named project title towards the completion of his doctoral research for the School of Built Environment, Engineering and Computing, Leeds Beckett University, United Kingdom.

The aim of this project is to optimally manage athletes' data in order to develop and apply novel advanced data mining and analytics techniques for athletes' movement profiling and assessment of athletes' performance in sports. The required objectives to achieve this aim are to:

- 1) design and develop a central database that optimally manages athletes' data (i.e. a backend for data acquisition and integration of various homogeneous data sources and its management. In addition, a front end for generating report and exportation);
- 2) design and develop an algorithm for athletes' movement patterns:
 - a) develop a novel algorithm for L-length frequent consecutive sequences (L-LFCS),
 - b) apply the algorithm developed in objective two (2a) to rugby sports data;
- 3) develop a robust framework for rugby athlete frequent and unique movement profiling (i.e. a novel or hybridized method capable of identifying the unique movement of athletes and even based on position); and
- 4) design and develop a novel method for assessing player performance.

It is expected that this project output the following outcomes that will be beneficial to the club:

- i- highly optimized database schema for rugby sports data;
- ii- An organized central database (proof-of-concept) with front and back ends for swift storage, reporting and analysis enablement;
- iii- a designed and developed novel algorithm for finding L-length frequent consecutive sequences from data strings which must have the aforementioned constraints;
- iv- an implementation of (-iii-) in the form of Python/R libraries which will enable athlete movement sequence analysis.
- v- a designed and developed method for finding frequent movement signatures of rugby sports athletes.

Figure A.2: The Rugby Football League Inform Consent

Appendix B

Feature Selection Results

B.1 Correlation-based Subset Evaluator Technique on original relative frequency Data

=== Run information === Evaluator: weka.attributeSelection.CfsSubsetEval
Search: weka.attributeSelection.BestFirst -D 1 -N 5
Relation: RugbyLeaguePMPatterns_rel-weka.filters.unsupervised.attribute.Remove-R7168,7170
Instances: 4640
Attributes: 7168
list of attributes omitted
Evaluation mode: evaluate on all training data

=== Attribute Selection on all input data ===
Search Method:
Best first.
Start set: no attributes
Search direction: forward
Stale search after 5 node expansions
Total number of subsets evaluated: 421,150
Merit of best subset found: 0.126
Attribute Subset Evaluator (supervised, Class (nominal): 7168 Position):
CFS Subset Evaluator

B.2 Consistency-based Subset Evaluator Technique on original relative frequency Data

Including locally predictive attributes

Selected attributes: 92, 235, 297, 346, 672, 721, 725, 881, 999, 1141, 1163, 1452, 1658, 1687, 1856, 1956, 1976, 2300, 2392, 2422, 2622, 2831, 2966, 2991, 3049, 3070, 3206, 3238, 3308, 3526, 3629, 3681, 3705, 3851, 3996, 4071, 4121, 4314, 4465, 4478, 5002, 5229, 5297, 5310, 5422, 5624, 5717, 5737, 5879, 6101, 6668, 6705, 6897, 7065
: 54

SST, ij, no, iij, jiii, jjj, ijj, bbb, uvuuuvu, jj, ST, HHG, SSS, quuuu, vuuuvv, qv, ijjj, SS, eff, nb, rm, jf, ru, mr, mo, HH, uqu, uuvuvv, vuuvu, uuuv, vvv, vvuuuv, vv, vvuuu, GH, efa, ff, jjjj, vvvuuu, jje, if, jjf, quv, jjii, vuuuvuu, fb, fff, jji, HGH, je, TS, mq, iijj, and GHH.

B.2 Consistency-based Subset Evaluator Technique on original relative frequency Data

=== Run information ===

Evaluator: weka.attributeSelection.ConsistencySubsetEval

Search: weka.attributeSelection.BestFirst -D 1 -N 5

Relation: RugbyLeaguePMPatterns_rel-weka.filters.unsupervised.attribute.Remove-R7168,7170

Instances: 4640

Attributes: 7168

list of attributes omitted

Evaluation mode: evaluate on all training data

=== Attribute Selection on all input data ===

Search Method:

Best first.

Start set: no attributes

Search direction: forward

Stale search after 5 node expansions

Total number of subsets evaluated: 264,519

Merit of best subset found: 0.999

Attribute Subset Evaluator (supervised, Class (nominal): 7168 Position):

B.3 Correlation-based Subset Evaluator Technique on Oversampled Relative frequency Data

Consistency Subset Evaluator

Selected attributes: 225, 284, 297, 346, 621, 701, 721, 725, 975, 1268, 1637, 1713, 1794, 1844, 1850, 1882, 2422, 2831, 2961, 2991, 3049, 3498, 3705, 4513, 4826, 5737, 5762, 5918, 6058, 6226, 6495, 7108 : 32

eef, cb, no, iij, mmm, bc, jjj, ijj, uq, kj, fg, be, ju, jk, fa, vuvvu, nb, jf, HG, mr, mo, vvuuv, vv, oo, fj, jji, jii, on, iie, mb, gf, and vuvuu.

B.3 Correlation-based Subset Evaluator Technique on Oversampled Relative frequency Data

=== Run information ===

Evaluator: weka.attributeSelection.CfsSubsetEval

Search: weka.attributeSelection.BestFirst -D 1 -N 5

Relation: RugbyLeaguePMPatterns_rel_SMOTE

Instances: 6825

Attributes: 7168

list of attributes omitted

Evaluation mode: evaluate on all training data

=== Attribute Selection on all input data ===

Search Method:

Best first.

Start set: no attributes

Search direction: forward

Stale search after 5 node expansions

Total number of subsets evaluated: 548933

Merit of best subset found: 0.228

Attribute Subset Evaluator (supervised, Class (nominal): 7168 Position):

CFS Subset Evaluator

Including locally predictive attributes

Selected attributes: 235, 301, 346, 672, 721, 725, 755, 813, 881, 1163, 1452, 1561, 1637, 1658, 1794, 1850, 1882, 1956, 1976, 2025, 2300, 2422, 2622, 2831, 2884, 2991,

B.4 Consistency-based Subset Evaluator Technique on Oversampled Relative frequency Data

3049, 3051, 3070, 3143, 3157, 3206, 3308, 3584, 3635, 3681, 3682, 3837, 3900, 3951, 4314, 4465, 4478, 4489, 4671, 4773, 4826, 4874, 5002, 5142, 5229, 5297, 5310, 5422, 5595, 5624, 5717, 5737, 5879, 5918, 6099, 6101, 6122, 6311, 6318, 6597, 6668, 6705, 6714, 6736, 6897, 7065 : 72

ij, nq, iij, jiii, jjj, ijj, TSS, mno, bbb, ST, HHG, efff, fg, SSS, ju, fa, vuvvu, qv, ijij, uvuuuvuu, SS, nb, rm, jf, jie, mr, mo, vuuvuu, HH, fi, iv, uqu, vuuvu, vvuvu, vvuv, vvuuuv, effff, uuvvv, nno, uG, jjjj, vvuuu, jje, vuuvvu, jv, jfe, fj, ej, if, ffff, jjf, quv, jjii, vuuvuu, nr, fb, fff, jji, HGH, on, vvuu, je, ej, jij, om, GGH, TS, mq, uvuuuv, ffef, iijj, and GHH.

B.4 Consistency-based Subset Evaluator Technique on Oversampled Relative frequency Data

=== Run information ===

Evaluator: weka.attributeSelection.ConsistencySubsetEval

Search: weka.attributeSelection.BestFirst -D 1 -N 5

Relation: RugbyLeaguePMPatterns_rel_SMOTE

Instances: 6825

Attributes: 7168

list of attributes omitted

Evaluation mode: evaluate on all training data

=== Attribute Selection on all input data ===

Search Method:

Best first.

Start set: no attributes

Search direction: forward

Stale search after 5 node expansions

Total number of subsets evaluated: 150303

Merit of best subset found: 1

Attribute Subset Evaluator (supervised, Class (nominal): 7168 Position):

Consistency Subset Evaluator

B.4 Consistency-based Subset Evaluator Technique on Oversampled Relative frequency Data

Selected attributes: 701, 721, 725, 1141, 1561, 1647, 2771, 3157, 3491, 3635, 3705, 4121, 4778, 4826, 5624, 5737 : 16
bc, jjj, ijj, jj, efff, rv, nmmn, iv, vvvv, vvuv, vv, ff, ur, fj, fb, and jjj.