공학석사학위논문

6MapNet: Characterizing Soccer Players via a Deep Embedding of GPS Data

GPS 데이터 임베딩을 통한 축구선수 특성 표현

2021 년 2 월

서울대학교 대학원

산업공학과

김 현 성

6MapNet: Characterizing Soccer Players via a Deep Embedding of GPS Data

GPS 데이터 임베딩을 통한 축구선수 특성 표현

지도교수 조 성 준

이 논문을 공학석사 학위논문으로 제출함

2020 년 12 월

서울대학교 대학원

산업공학과

김 현 성

김현성의 공학석사 학위논문을 인준함

2021 년 1 월

위 원 장 이 재 욱

부위원장 조 성 준

위 원 박 종 헌

Abstract

6MapNet: Characterizing Soccer Players via a Deep Embedding of GPS Data

Hyunsung Kim
Department of Industrial Engineering
The Graduate School
Seoul National University

The values of individual players have become astronomical with the growth of the football industry. Nevertheless, subjective judgments have taken a big part to this day due to a lack of data. Recently, there have been new attempts to quantitatively grasp players' styles using video-based event stream data. However, only containing on-the-ball actions of players, data collected from a single match were not enough to represent each player's tendency. In this paper, we propose a deep learning approach that can effectively capture the movement styles of players using in-game GPS data. Without any information of soccer-specific actions, simply each player's locations and velocities are used to generate two types of heatmaps. Our deep architecture then maps these heatmap pairs into embeddings whose similarity corresponds to the actual similarity of playing styles. In this study, each player's off-the-ball movements and roles assigned per match are included in the model, which were not considered in previous studies. We experimentally show that players can be effectively identified

with only a small number of matches. Considering the difficulty of obtaining a lot of data of unknown players in real-world scouting, this result implies that our method is more practical and scalable than previous ones.

Keywords: data mining, sports data analytics, soccer player representation, wearable tracking system, siamese neural networks

Student Number: 2018-23740

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Although accurate evaluations of players have become more and more important with the growth of the soccer industry, subjective judgments have taken a big part [1] due to a lack of data. The absence of objective criteria for evaluating or characterizing players caused huge gaps between the transfer fees and the actual values of players. It was not uncommon that a cheaply scouted player hit the jackpot, or a player signed with a huge amount of transfer fee just ended in failure. Thus, scouts have spent a lot of time and efforts on watching games of unknown players to find hidden gems without wasting money.

Meanwhile, video-based event tagging has been a solution to enrich soccer data. Companies such as Opta Sports [19] and StatsBomb [23] have collected event stream data from numerous games. Some studies used these event streams to analyze team styles [2, 9, 10, 12, 16]. Others focused on individual players, and made models to quantify their performances [4, 8, 11, 20] or identify their playing styles [6, 7, 15, 21].

However, event tagging has limitations in terms of scalability due to the large amount of manual work and the sparsity of resultant data. In general, event stream data consist of several attributes including agents, types, and locations of almost 2,000 events per match. That is, the data collection requires a lot of manual work by

human annotators. Moreover, data from a single match is not sufficient for figuring out a player's style. Actually, most studies to characterize players [6, 7, 15, 21] use the seasonal accumulation of each player. They have difficulties meeting the demand of real-world scouts who don't have an unknown player's full-season data. Especially, this lack of scalability become critical for most scouts who want to find hidden talents in youth or lower leagues.

Then, wearable technologies were introduced to the world of soccer. Companies such as Catapult Sports [5], STATSports [24], and Fitogether [14] started to use wearable trackers to collect players' locations during trainings and unofficial matches. After 2 years of the trial period, FIFA approved the use of electronic performance tracking systems (EPTS) on players' upper back areas in official matches by establishing International Match Standard (IMS) in 2017 [13]. Since wearable trackers could automatically record players' locations with high frequency [18], enough data were collected for effective analyses of each player just by attaching a tracker for a few matches.

In this study, we find an effective representation of players' styles using wearable GPS data. Without any information about soccer-specific actions such as passes, dribbles, and shots, simply the time-series of each player's locations and velocities are used to compute his location and direction heatmaps, respectively. To label the data, we quantify 10 roles for each phase in a match using the method proposed by Bialkowski et al. [2]. We label a single player's phase-by-phase data as the same, if they corresponds to similar roles. Inspired by Schroff et al. [22], we then build a deep siamese architecture minimizing the triplet loss so that the Euclidean similarities between the embeddings implies the actual similarity between the players in terms

of playing style. We evaluate our method by performing a player identification task proposed by Decroos et al. [6] that identifies the anonymized embeddings by their known neighbors.

The main contributions of our paper are as follows.

(a) Instead of event stream data, automatically collected locations and velocity vectors are used in our study. Thus, in a sense, our approach is more scalable than the previous ones in that no manual tagging of soccer-specific actions is needed to collect the data.

(b) We take players' roles into account, which is more reasonable in that a player's movements highly depend on the role given for the match. Role vectors of each player's phase-by-phase data are derived using the relative locations to the teammates, and this is possible because multiple players' data are collected at the same time.

(c) We also consider players' off-the-ball movements, which accounts for most of their movements in soccer matches. This is possible because every moment of each player is recorded during a match.

(d) Due to the density of GPS data, a player can be identified with only a small number of matches. In real-world scouting, there are many difficulties in getting full season data of every unknown player. Therefore, a system that can characterize players with only a small number of matches is more practical.

Chapter 2

Related Works

As large amounts of data were collected in soccer matches, some recent studies have tried to quantify the playing style of soccer players using those data. Most studies used video-based event stream data, which is a time-series containing the agents, types, start locations, and end locations of soccer-specific actions such as passes, dribbles, and shots. They define a playing style as a group of locational tendencies of carrying out certain types of actions [7], and try to find feature vectors that effectively represent playing styles.

Gyarmati et al. [15] used 660,848 movement vectors collected from 2012/13 La Liga to derive a feature vector representing each player. They clustered all movement vectors in the dataset by their start and end locations using mini-batch K-means clustering. Then for each player, they constructed a feature vector by simply counting the movement vectors per cluster. However, they did not take into account the difference of action type. (For example, both passes and dribbles are counted as *movements*.) Also, they did not suggest a method to quantitatively evaluate their approach.

Decroos et al. [6] made a more sophisticated approach. They selected pass, dribble, cross, and shot as action types relevant to the player's attacking style, and found

the principal components by applying nonnegative matrix factorization (NMF) to the heatmaps for each action type. Each player's 18-dimensional feature vector was then constructed by concatenating the weights multiplied on the principal components to reconstruct the original heatmaps. The resulting vector named *player vector* was shown to be an interpretable summary of the player's style. Also, they proposed an evaluation method of identifying anonymized players, which we also perform in Section 5.2.

Decroos et al. [7] also proposed another method named *SoccerMix* that uses a soft clustering instead of the absolute counting in the previous researches (*i.e.* counting movements in each cluster for Gyarmati et al. [15], and in each grid cell for Decroos et al. [6]). They fit the probability distribution of each action type as a combination of finite Von Mises distributions, each of which describes a prototypical action of a certain type, location, and direction. Then, they represent each action as the vector of responsibilities of the prototypical actions (*i.e.* Von Mises distributions) on the action. SoccerMix was shown to have a better performance in identifying anonymized players than player vectors [6] mentioned above.

One thing the above approaches didn't consider is that a player can take multiple roles (*i.e.* 'positions' in terms of soccer tactics) that affects the his or her locational tendency. Since they aggregate all the data of each player in a season regardless of his or her roles, a player's instrinsic style is confused with the role he or she is assigned to. Also, they only consider on-the-ball actions which is just a small part of each player's movement. This sparsity of the data leads that a few matches are not sufficient for figuring out a player's style. As such, they used the seasonal accumulation of each player to construct a feature vector.

In this paper, we propose a deep learning approach using GPS data that overcomes these drawbacks. Since GPS data collect every player's every moment, all the movements occured in a match can be taken into account. Taking advantage of the simultaneous data collection of all the players in the team, we find information of varying roles throughout a match and use this as labels of our model. Due to the density of GPS data, our proposed model effectively characterize players even with a few matches of data.

Chapter 3

Definition of Terms

In this section, we define terms that are frequently used in this paper.

First, the term *match* is defined differently. A confusing point of our dataset is that both teams' data are collected for some matches, and only one teams' data are collected for others. Thus, we halve each match as two separate 'matches' each of whose subject is a single team. That is, a match for whom the both teams are measured is counted twice.

Also, we refer to Bialkowski et al. [2] to define *role* and *formation*.

Definition 3.1 (**role**). *A role within a team is an area where each player is assigned responsibility relative to the other teammates. A role is represented by a probability distribution on the 2D soccer pitch, and a 2D role vector is then calculated as the mean of the distribution.*

Definition 3.2 (**formation**). *A formation F is a set of role vectors $\{r_1, \cdots, r_M\}$ respectively assigned to the players simultaneously playing in a time interval in a match.*

Definition 3.3 (**phase**). *A phase T in a match is a connected time interval in which the formation of a team does not change.*

Bialkowski et al. assumed that a team's formation is consistent within a half of a match. That is, they assumed that a match (without overtime) consisted of exactly two phases (the first half and the second half).

On the other hand, we add a domain intuition to divide a match into more number of phases. Considering that most tactical changes occur when a new session is started or player composition is changed, we split each match so that a new phase begins at the start of the second half, player substitution and player sent-off. Now, each player's data can be identified by the phases he or she played.

Definition 3.4 (**player-phase entity**). *A player-phase entity is a player in a single phase. That is, if a player p have played in phases $T_1, \cdots, T_N$ during a season, there exist N player-phase entities $(p, T_1), \cdots, (p, T_N)$ for the player.*

In Section 4.2 and 4.3, a role vector is assigned for each player-phase entity. Then, the role vectors corresponding to each player are partitioned into 1 to 3 clusters. Players who have more than one role cluster is considered to have multiple identities as follows.

Definition 3.5 (**player-role entity**). *A player-role entity is a player's identity determined by the corresponding role cluster. That is, if a player's role vectors $r_1, \cdots, r_N$ respectively assigned to player-phase entities $(p, T_1), \cdots, (p, T_N)$ are partitioned into K clusters $R_1, \cdots, R_K$, then p is considered to have K different identities $(p, R_1), \cdots, (p, R_K)$ and each identity is called a player-role entity.*

These player-role entity labels are used to train the deep architecture in Section 4.6. Table 3.1 shows an example of player-phase entities and player-role entities.

Table 3.1: An example of player-phase entities and player-role entities. A player-phase entity is identified by a player and a phase, and a player-role entity is identified by a player and one of his or her role clusters. We assume that every player's role doesn't change during a phase.

Match	Phase	Player	Role cluster
Match 1	T_1	p_A	R_1
		p_B	R_4
	T_2	p_A	R_2
		p_D	R_7
	T_3	p_A	R_1
		p_D	R_7
Match 2	T_4	p_A	R_3
		p_B	R_5
		p_C	R_6
	T_5	p_A	R_3
		p_C	R_6
		p_D	R_7

$\longrightarrow$

Player	Player-role entity	Player-phase entity
p_A	(p_A, R_1)	(p_A, T_1)
		(p_A, T_3)
	(p_A, R_2)	(p_A, T_2)
	(p_A, R_3)	(p_A, T_4)
		(p_A, T_5)
p_B	(p_B, R_4)	(p_B, T_1)
	(p_B, R_5)	(p_B, T_4)
p_C	(p_C, R_6)	(p_C, T_4)
		(p_C, T_5)
p_D	(p_D, R_7)	(p_D, T_2)
		(p_D, T_3)
		(p_D, T_5)

Chapter 4

Methodology

In this chapter, we illustrate the process of finding player embeddings from their movement data one by one.

First, we describe the data collection and preprocessing steps in Section 4.1. As defined in Chapter 3, we divide each match into several phases in each of which the players' roles and the team's formation is assumed to be consistent. In Section 4.2, we find the formation of the team in each phase using the method of Bialkowski et al. [2], and assign a single dominant role to each player-phase entity. In Section 4.3, we partition the role vectors of each player into 1 to 3 role clusters. Player-phase entities corresponding to different role clusters are then considered as different identities.

In Section 4.4, we describe how to generate location and direction heatmaps for each player-phase entity. In Section 4.5, after splitting the heatmap dataset into training, validation, and test sets, we augment each dataset by simply accumulating the heatmaps of the same identities. These augmented heatmap pairs are used as inputs of 6MapNet introduced in Section 4.6. A subnetwork of 6MapNet is trained to take a heatmap pair and maps it to an embedding that effectively characterize the identity. Figure 4.1 shows an overview of the whole procedure.

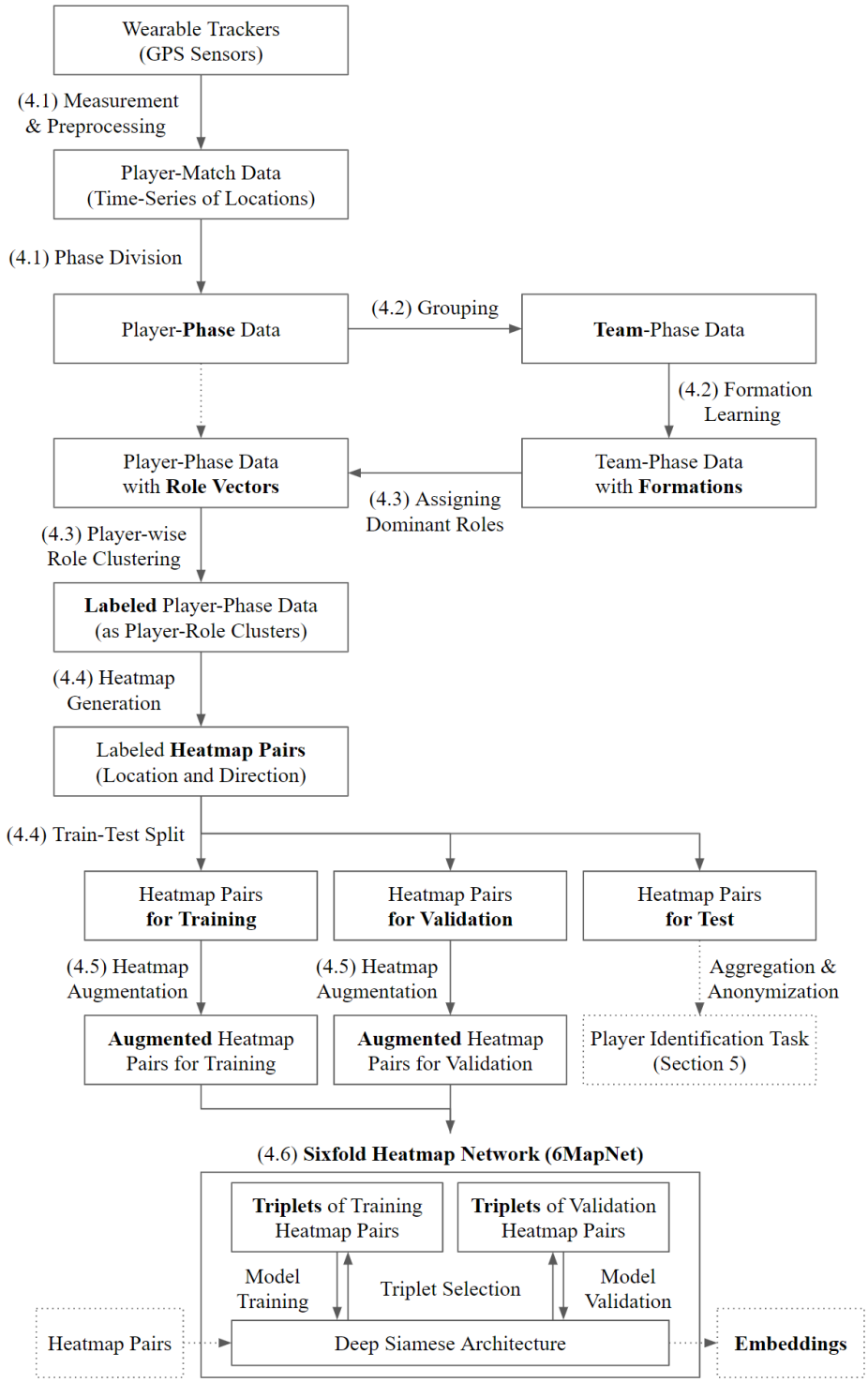


Figure 4.1: Overview of the whole procedure.

4.1 Data Preparation

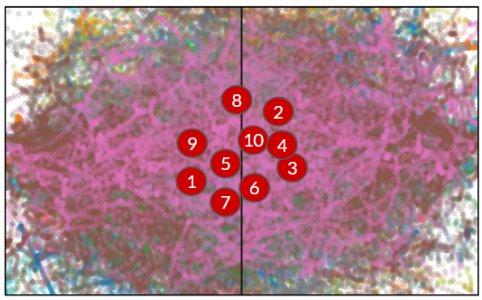
During soccer matches, a wearable GPS device (OhCoach Cell B) developed by Fitogether [14] is put on each player's upper back area. Latitudes, longitudes, and speeds are measured at 10 Hz. The devices are approved by FIFA to be used in official soccer matches. According to the result of FIFA's quality test in 2019, the tracking accuracy of OhCoach Cell B is well-above the industry standard. The devices have successfully collected the spatiotemporal data from 360 matches of 2019 K League 1 and K League 2, the first and the second division of the South Korean professional soccer league system.

The relative locations and velocities are computed from the raw data as follows. Location data are converted to points on 2D planar coordinates relative to the soccer pitch. To explain, each pair of latitude and longitude is transformed to x and y coordinates in meters, whose origin is at the bottom left corner of the pitch. Different transformations are applied to the data from each half so that the team always attacks along the positive direction of the x-axis. Also, 2-dimensional velocity vectors are then calculated by differentiating the local coordinates.

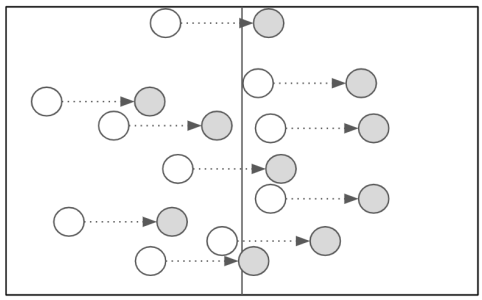
As defined in Chapter 3, we split each match so that a new phase begins at the start of the second half, player substitution and player sent-off. In addition, phases with a length of 10 minutes or less were absorbed into adjacent phases to ensure the minimum length of each phase. As a result, each match is divided into 2 to 6 phases, and 13,558 player-phase entities were generated for the total of 360 matches. Each player-phase entity has a time-series of tuples each of which contains a location vector (s_x, s_y) in meters and a velocity vector (v_x, v_y) in m/s.

4.2 Learning Phase-by-Phase Formations

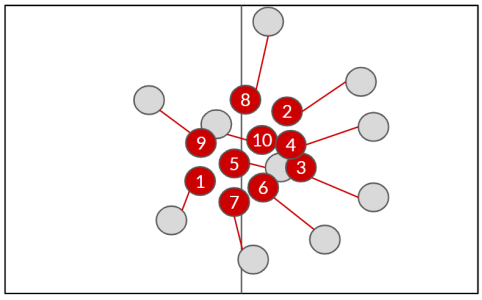
In this section, the phase-by-phase formations are learned by the method of Bialkowski et al. [2]. Player-phase entities of each player are then classified into one or several player-role entities by clustering the the player's role vectors.



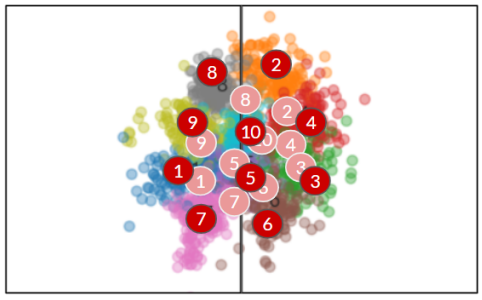
(a) Assign a unique role label (for example from 1 to 10) to each field player and set each of 10 initial role vectors as the mean location of the corresponding player.



(b) For each moment, translate the player locations by the team average to reduce the situational variation.



(c) **E-step:** Use the Hungarian algorithm for each moment in the phase to reassign a unique role to each player so that the sum of distances between players and their respective role vectors is minimized.



(d) **M-step:** Recompute role vectors by updating the mean location of each role.

Figure 4.2: The learning procedure of phase-by-phase formations based on the method of Bialkowski et al. [2].

Bialkowski et al. proposed an EM algorithm that dynamically assigned each

player a role at every moment. This algorithm is analogous to that of K-means, except using the Hungarian algorithm in E-step to satisfy the constraint that no two players take the same role at the same time. Figure 4.2 visually shows the specific procedure of the method. (We have slightly modified the details of the original method for brevity.)

After each EM iteration, the moments where the assignment cost was too large were considered outliers such as the set piece situation and thus deleted. The algorithm was terminated after 3-4 iterations until there was no improvement in the overall assignment cost. The resulting role assignment satisfies the followings.

(a) Each player's role varies throughout the phase.

(b) No two players in a team have the same role at the same time.

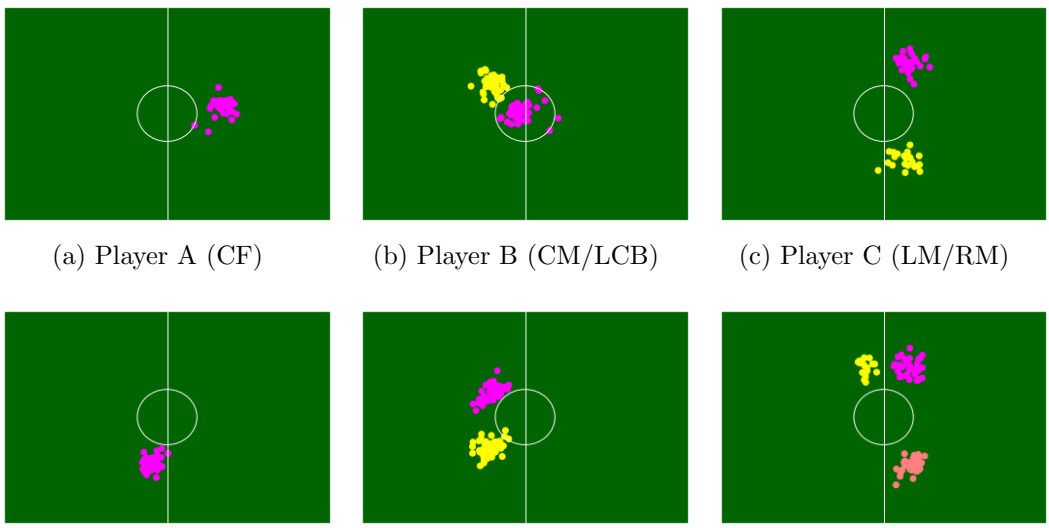
(c) The role-by-role probability distributions exhibit minimal overlap.

(d) The role vectors represent the actual 'positions' (in terms of soccer tactics) well. That is, role vectors corresponding to the same position are similar.

4.3 Player-wise Role Vector Clustering

In this section, for the purpose of data labeling, individual clustering task is performed for each player's data based on the learned role vectors. First, the result of Section 4.2 exhibits too frequent role changes to be used in labeling. Therefore, assuming that the player's actual role does not change within a phase, we assign a single dominant role to each player-phase entity. Since a single player mostly plays from 1 to 3 roles throughout a season, the role vectors of each player are partitioned well into 1 to 3 discriminative clusters.

K-means clustering is performed with the number of clusters that maximizes the silhouette score. If all the silhouette scores are less than 0.6, all of the player's data are considered to belong to a single cluster.



(a) Player A (CF) (b) Player B (CM/LCB) (c) Player C (LM/RM)

(d) Player D (RB) (e) Player E (LCB/RCB) (f) Player F (LM/LB/RM)

Figure 4.3: Role vectors of individual players colored per cluster. The caption of each subfigure indicates the player index and role name that we intuitively give to each cluster. The first role name (*i.e.* LM in (f) for example) corresponds to the pink cluster, the second does to the yellow, and the third does to the orange.

Figure 4.3 shows some examples of the players with their clustered role vectors. Player A and D have played only one role throughout the season. Meanwhile, the role vectors of each of B, C, and E are partitioned into 2 clusters with silhouette score higher than 0.6. These players are considered to have two different identities (*i.e.* player-role entities). Likewise, player F is considered to have 3 roles during the season and thus have 3 identities.

Each time-series (*i.e.* data from each player-phase entity) is then labeled as the pair of the player name and the corresponding role cluster. That is, if the role vectors of the same player's two time-series belong to a single cluster, the two time-series are labeled as the same identity. On the other hand, if those of the same player belong to different clusters, they are labeled as different identities. Using these labels, triplets are generated and a deep siamese architecture minimizing the triplet loss is constructed in Section 4.6.

4.4 Generating Location and Direction Heatmaps

For the labeled time-series, location and direction heatmaps are generated by making grids on the fixed domains and counting data points in each grid cell.

First, we compute each player-phase entity's location heatmap using a grid of size 35×50 overlaid on the soccer pitch (Figure 4.4a). The x value of each location is translated by team average to reduce the variation by tides of or tactics for the match. (Otherwise, a player's location is highly influenced by these factors, which are irrelevant to his characteristics. For instance, if a team is on the defensive in a match, the players' locations tend to be distributed on the back side of the pitch.) Specifically, we compute heatmaps using

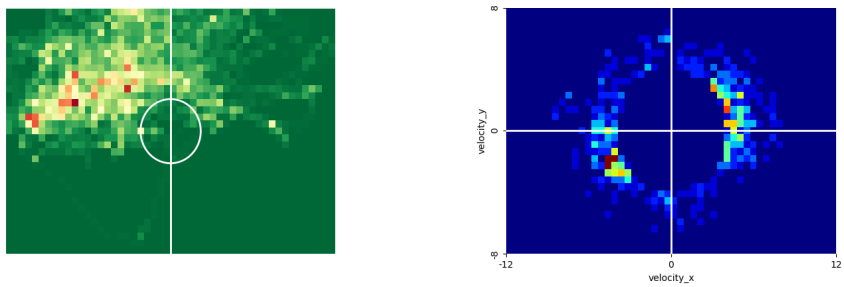
$$((s_x - \bar{s}_x) \cdot 0.8 + L_x/2, s_y) \tag{4.1}$$

where $s = (s_x, s_y)$ is the location at each time, $\bar{s} = (\bar{s}_x, \bar{s}_y)$ is the average team location in the phase, and L_x is the length of the soccer pitch. A constant 0.8 is multiplied to the translated x values, so that each heatmap is compressed not to be out of bounds.

What the location heatmaps cannot cover is the direction of each locomotion. Especially, a player's running direction during a game can be an important feature for player characterization. Thus, we also compute heatmaps for velocity vectors whose speed is higher than a threshold speed (Figure 4.4b). Similar to the above, we overlay a grid of size 35×50 on the rectangle

$$\{(v_x, v_y) | -12\text{m/s} \leq v_x \leq 12\text{m/s}, -8\text{m/s} \leq v_y \leq 8\text{m/s}\}, \tag{4.2}$$

and count the end points of the velocity vectors in each grid cell.



(a) Raw location heatmap by making a grid on the soccer pitch, and counting the locations in each grid cell.

(b) Raw direction heatmap by making a grid on the 2D plane where the start points of velocity vectors are on the origin, and counting the end points in each grid cell.

Figure 4.4: The location and direction heatmaps of a player-phase entity.

The grid sizes are set to be the same (*i.e.* 35×50) for the location and direction heatmaps, since these are the inputs of the convolutional neural networks in Section 4.6 with the same shape. The boundary values of the x-axis and y-axis of the direction heatmap (*i.e.* $\pm$12m/s and $\pm$8m/s, respectively) are determined so that the shape of each grid cell is close to a square. The threshold speed is empirically set to be 4m/s. For lower speed thresholds, data points are concentrated near the origin without any tendency. Also, for higher speed thresholds, the number of data points is not enough for effective analysis.

Before the data augmentation in Section 4.5, the dataset is split into training, validation, and test sets. In order to perform the player identification experiment in Section 5.2, the test dataset is constructed by sampling the same number of heatmap pairs (*i.e.* pair of location and direction heatmaps of a player-phase entity) for each identity. Similarly, the validation dataset is sampled from the rest of the data.

4.5 Data Augmentation by Accumulating Heatmaps

One of the good properties of players' heatmaps is their 'additivity'. In other words, we can compute the accumulated heatmaps of the multiple time intervals by simple pixel-wise additions. Formally speaking, let $s_p : \mathcal{T} \longrightarrow \mathbb{R}^2$ and $v_p : \mathcal{T} \longrightarrow \mathbb{R}^2$ be the location and velocity of a player p, respetively defined on the time axis $\mathcal{T}$. Also, let $\mathbf{h} : \mathcal{P}\left(\mathbb{R}^2\right) \longrightarrow \mathbb{Z}_*^{35\times 50}$ be the heatmap calculated for sets of 2-dimensional vectors. Then for disjoint time-intervals $T_1, \cdots, T_m \subset \mathcal{T}$,

$$\mathbf{h}\left(\bigcup_{i=1}^{m} s_p(T_i)\right) = \sum_{i=1}^{m} \mathbf{h}(s_p(T_i)) \tag{4.3}$$

$$\mathbf{h}\left(\bigcup_{i=1}^{m} v_p(T_i)\right) = \sum_{i=1}^{m} \mathbf{h}(v_p(T_i)) \tag{4.4}$$

Using this additivity, we augment the dataset by accumulating several heatmaps of the same identities (*i.e.* same player-role entities).

The augmentation task is performed separately for each of the training, validation, and test dataset so that data in different sets are not mixed with each other. 3 heatmaps are sampled from the set of each identity's heatmaps and accumulated to be a single augmented heatmap by a pixel-wise addition. In this way, we repeat sampling without duplicates to make a large number of augmented heatmaps. Figure 4.5 shows an example of heatmap augmentation.

A different 3-set (*i.e.* a subset composed of 3 heatmaps) extraction method is applied between training and validation/test data. Since the validation and test datasets have the same number of heatmap pairs per player-role entity, 3-sets are extracted through the exhaustive combination. That is, if there are exactly k pairs for

each player-role entity in the dataset, $\binom{k}{3}$ accumulated heatmap pairs are generated per player-role entity. On the other hand, the number of heatmap pairs for each player-role entity varies in the training dataset, and some players even have more than 100 pairs. In this case, the exhaustive combination is computationally expensive and deepens the class imbalance. Therefore, to ensure that the size of augmented training dataset does not exceed m times the original training data size, we randomly sample 3-sets by $m \cdot n_p$ times for each player-role entity p with n_p heatmap pairs and remove the duplicates.

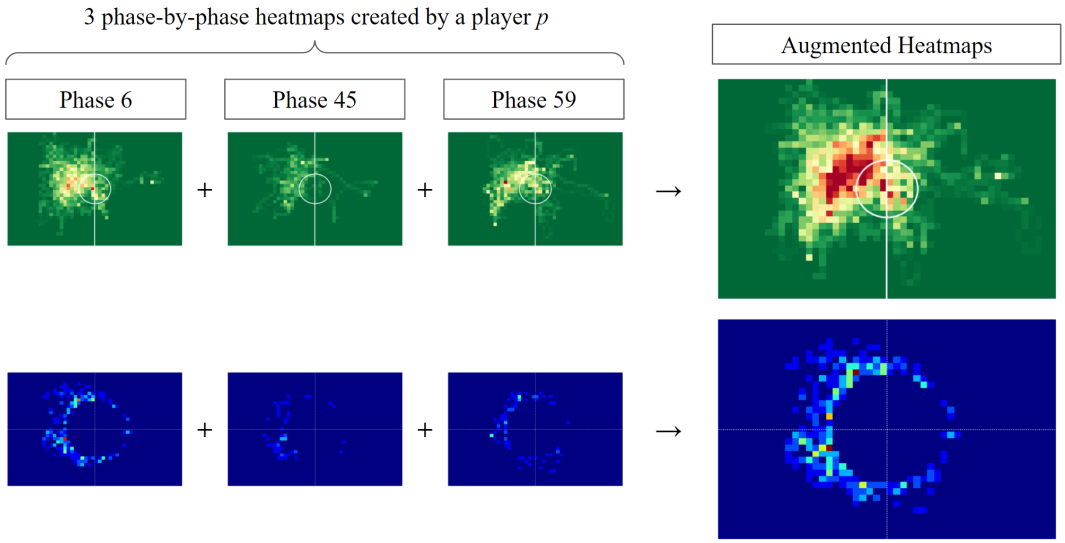


Figure 4.5: An example of heatmap augmentation.

The average duration of one phase is about 30 minutes, which is not enough to understand a player's movement tendency. Heatmap augmentation ensures that the individual augmented heatmap is of sufficient duration to indicate the player's movement tendency. Also, it enrich the input data to the deep neural network introduced in Section 4.6.

4.6 Building the Sixfold Heatmap Network

In this section, a deep neural network named *sixfold heatmap network* (*6MapNet*) is built to learn a Euclidean embedding for each heatmap pair. If a mapping f embeds heatmap pairs of the same identity to similar feature vectors, then we can expect that those of the identities whose actual playing style is similar also have small distances from each other. Therefore, we aim to make embeddings of the same player-role entities close to each other.

To do this, the idea of FaceNet by Schroff et al [22] is used. FaceNet takes a triplet composed of $\mathbf{x}^a$ (anchor), $\mathbf{x}^p$ (positive), and $\mathbf{x}^n$ (negative) as an input, where $\mathbf{x}^a$ and $\mathbf{x}^p$ are images of the same person and $\mathbf{x}^n$ is that of another person. Then it learns an embedding $f(\mathbf{x})$ from each image $\mathbf{x}$ into a feature space $\mathbb{R}^d$ that minimizes a triplet loss defined as

$$\mathcal{L} = \sum_i \left[\|f(\mathbf{x}_i^a) - f(\mathbf{x}_i^p)\|_2^2 - \|f(\mathbf{x}_i^a) - f(\mathbf{x}_i^n)\|_2^2 + \alpha \right]_+ \tag{4.5}$$

By minimizing $\mathcal{L}$, it tries to make $f(\mathbf{x})$ satisfy

$$\|f(\mathbf{x}_i^a) - f(\mathbf{x}_i^p)\|_2^2 + \alpha \leq \|f(\mathbf{x}_i^a) - f(\mathbf{x}_i^n)\|_2^2 \tag{4.6}$$

for every possible triplet $(\mathbf{x}_i^a, \mathbf{x}_i^p, \mathbf{x}_i^n)$, which means that faces of the same identity have smaller distances than those of distinct people.

To apply the architecture of FaceNet to our heatmap dataset, we build 3 identical subnetworks (*i.e.* subnetworks sharing weights, and thus called *siamese* neural networks [3]), each of which has two branch convolutional neural networks (CNNs)

(Figure 4.6). Each subnetwork takes an heatmap pair $\mathbf{x}_i = (\mathbf{h}(s_i), \mathbf{h}(v_i))$ as an input, and each branch CNN (one for location heatmaps $\mathbf{h}(s_i)$ and the other for direction heatmaps $\mathbf{h}(v_i)$) maps the corresponding heatmap into a d'-dimensional vector. These two d'-dimensional vectors are then concatenated to a single $2d'$-dimensional vector used as features for the final fully connected (FC) layer. We use the d-dimensional output of the FC layer as an embedding $f(\mathbf{x}_i) = f(\mathbf{h}(s_i), \mathbf{h}(v_i))$of each input heatmap pair $\mathbf{x}_i = (\mathbf{h}(s_i), \mathbf{h}(v_i))$ to compute the triplet loss of anchor, positive, and negative inputs.

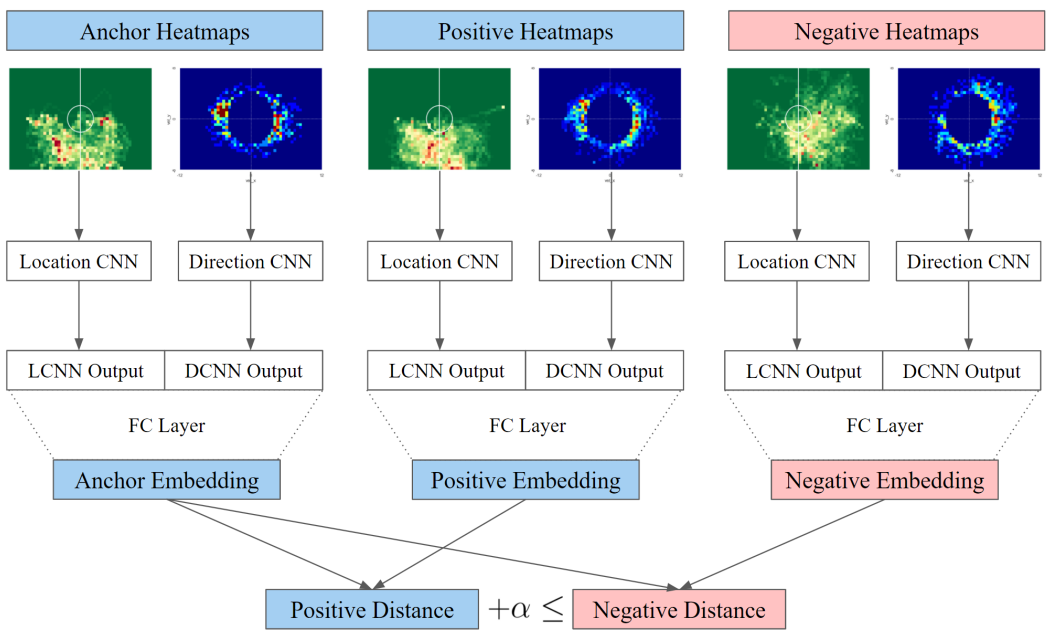


Figure 4.6: 6MapNet overview: The anchor and positive heatmaps are of the same player-role entity, while the negative heatmaps are of another. The three location CNNs are identical (*i.e.* share weights with each other), and the three direction CNNs are also identical. We divide each embedding by its norm as in FaceNet so that every embedding has a unit norm.

In summary, the whole architecture of 6MapNet receives 6 heatmaps

$$(\mathbf{x}_i^a, \mathbf{x}_i^p, \mathbf{x}_i^n) = (\mathbf{h}(s_i^a), \mathbf{h}(v_i^a), \mathbf{h}(s_i^p), \mathbf{h}(v_i^p), \mathbf{h}(s_i^n), \mathbf{h}(v_i^n)), \tag{4.7}$$

return a triplet of d-dimensional embeddings $(f(\mathbf{x}_i^a), f(\mathbf{x}_i^p), f(\mathbf{x}_i^n))$ and use Eq. 4.5 as a loss function to minimize. If 6MapNet returns embeddings that the Euclidean similarities between them correspond to the similarities between the players' actual movements, then we can say that the embedding $f(\mathbf{x})$ is an effective representation of playing styles of an identity who made $\mathbf{x}$ during soccer matches.

Chapter 5

Experiments

5.1 Implementation Detail

In this study, 360 matches of data are initially used and split into 1,263 phases. Phases with more than 8 measured players are used for role vector learning in Section 4.2 and player-wise clustering in Section 4.3. As a result, 276 players are divided into 382 player-role entities, which is used as labels for the 9,317 player-phase entities.

In Section 4.4 and 4.5, we only use 311 player-role entities who have more than 10 labeled player-phase entities. In training-validation-test split, the test set is constructed by sampling 5 player-phase entities per player-role entity, and the validation set is done by sampling 5 for each of 229 player-role entities who have more than 15 labeled player-phase entities. The augmentation task (*i.e.* repeatedly sampling 3 heatmaps for each player-role entity and accumulating them by a pixel-wise addition) is independently performed for each of the training, validation, and test dataset. Augmented training data of size 23,518 (heatmap pairs), validation data of size 2,290, and test data of size 3,110 are generated as a result. (Note that since all the 3-combinations are used to augment validation and test data, the resulting dataset size is $\binom{5}{3} = 10$ times the number of corresponding player-role entities.)

As in FaceNet [22], the choice of triplets for 6MapNet in Section 4.6 has a great

influence on the success or failure of the learning. For each heatmap pair as an anchor, we select a random positive and a hard negative. Specifically, we do the triplet selection below for every 10 or fewer epochs:

(a) Sample 5 heatmap pairs for each identity to form a candidate set S to be used as positive or negative.

(b) Use every heatmap pair in the dataset as an anchor $\mathbf{x}^a$ and combine it with the corresponding 5 positive candidates $\mathbf{x}^p \in S^a = \{\mathbf{x}^p \in S | y^p = y^a\}$ (*i.e.* $\mathbf{x}^p \in S$ that is of the same identity as $\mathbf{x}^a$) to make a 5 positive pairs $(\mathbf{x}^a, \mathbf{x}^p)$.

(c) For each positive pair $(\mathbf{x}^a, \mathbf{x}^p)$, pick one hard negative $\mathbf{x}^n \in S - S^a$ that does not satisfy Eq. 4.6. If there is no such negative, randomly choose one of the 10 negatives with the smallest distance from the anchor.

In particular, the rate of positive pairs without hard negatives in (c), *i.e.*

$$\left|\{(\mathbf{x}^a, \mathbf{x}^p) \in P : \|f(\mathbf{x}_i^a) - f(\mathbf{x}_i^p)\|_2^2 + \alpha \leq \|f(\mathbf{x}_i^a) - f(\mathbf{x}_i^n)\|_2^2 \ \forall \mathbf{x}^n \in S - S^a\}\right| / |P| \tag{5.1}$$

(where P is the set of all positive pairs selected in (b)) is used as validation accuracy during the training. As a result, 69,408 triplets for the training data and 6,183 for the validation data are selected for every 10 or fewer epochs.

We start the training with randomly initialized weigths. Adam is used as the optimizer with initial learning rate 0.01, which is decayed during the training procedure. We fix mini-batches of size 1,000, so the weights are updated 70 times per epoch. The model is trained with a maximum of 10 epochs for a selected triplets. If there is no significant increase in validation loss, the triplet is newly selected. During

each triplet selection task, the learning performance is evaluated with the validation accuracy given in Eq. 5.1. If there is no improvement in the validation accuracy, the whole learning procedure is finished. Table 5.1 shows the structure of a single branch CNN (*i.e.* each of the location CNN and direction CNN).

Table 5.1: The locational and directional branch CNNs of 6MapNet have the same structure as in this table. There are a little changes from a standard CNN structure in padding or kernel sizes such as in Conv1a and Conv3a, in order to handle the odd dimension of inputs. Batch normalization is done after every layer, and dropouts with probatility 0.25 are done after some layers.

Layer	Input	Padding	Kernel	Activation	Output
Conv1a	$35 \times 50 \times 1$	1×0	$2 \times 3 \times 32$, 1	ReLU	$36 \times 48 \times 32$
Conv1b	$36 \times 48 \times 32$	1×1	$3 \times 3 \times 32$, 1	ReLU	$36 \times 48 \times 32$
MaxPool1	$36 \times 48 \times 32$	-	2×2, 2	-	$18 \times 24 \times 32$
Conv2a	$18 \times 24 \times 32$	1×1	$3 \times 3 \times 64$, 1	ReLU	$18 \times 24 \times 64$
Conv2b	$18 \times 24 \times 32$	1×1	$3 \times 3 \times 64$, 1	ReLU	$18 \times 24 \times 64$
MaxPool2	$18 \times 24 \times 64$	-	2×2, 2	-	$9 \times 12 \times 64$
Conv3a	$9 \times 12 \times 64$	1×1	$2 \times 3 \times 128$, 1	ReLU	$10 \times 12 \times 128$
Conv3b	$10 \times 12 \times 128$	1×1	$3 \times 3 \times 128$, 1	ReLU	$10 \times 12 \times 128$
MaxPool3	$10 \times 12 \times 128$	-	2×2, 2	-	$5 \times 6 \times 128$
Conv4a	$5 \times 6 \times 128$	1×1	$3 \times 3 \times 256$, 1	ReLU	$5 \times 6 \times 256$
Conv4b	$5 \times 6 \times 256$	1×1	$3 \times 3 \times 256$, 1	ReLU	$5 \times 6 \times 256$
FC1	7680	-	-	ReLU	128
FC2	128	-	-	ReLU	32

The training goes very quickly. It starts with nearly 0% accuracy and reaches its peak performance after dozens of epochs. The best performance is obtained when α in Eq. 4.5 is set to 0.15. We used the model with 92.6% validation accuracy (in terms of Eq. 5.1) in the experiments of the next sections.

5.2 Identifying Anonymized Players

There is no existing standard for evaluating our method, since the playing style is a subjective concept. Thus, we intuitively assume that small distances between the same identities (*i.e.* player-role entities) imply the small distances between similar identities (in terms of playing styles). Based on this assumption, we perform an experiment proposed by Decroos et al. [6].

What they proposed is a player identification task to evaluate a model for characterizing soccer players. Given an anonymized player's data, they check how the model can identify each player using the data from the previous season. In the succeeding study [7], they perform this task both for player vectors (NMF) [6] and Soccermix [7]. They prepared 193 players who had played at least 900 minutes in both of 2017/2018 and 2018/19 seasons of English Premier League. Then they anonymized player-by-player feature vectors from the 2018/19 season, and check how close the feature vector of own in 2017/18 season is to each one. The exact results taken from the paper is attached in Table 5.2.

Table 5.2: Top-k results out of 193 players for the player identification task by Decroos et al. [7]. Full season data for each anonymized player are used to find the right player.

Method	Top-1	Top-3	Top-5	Top-10
Player Vectors [6]	36.5%	53.2%	66.5%	83.2%
SoccerMix [7]	48.2%	62.7%	71.5%	80.8%

Meanwhile, we perform the same task but using much less matches anonymized data. The test dataset generated in Section 4.4, containing 5 phase-by-phase heatmaps per player-role entity, is used as anonymized data. Specifically, we compute the em-

beddings of the heatmap pairs aggregated by player-role entity, and compare them to those from the training dataset (which also consists of aggregations by player-role entity).

Table 5.3 shows the results of an experiment. We first perform the task with all the 311 players in the test set, and then with sampled 193 player-role entities to compare the performance to the previous methods. We see that the identifying performance is not much less than the previous methods, and even higher for the top-5 and top-10 accuracies.

Table 5.3: Top-k results out of 311 and 193 player-role entities for the player identification task. For the latter, 193 identities are sampled randomly from the test dataset for the comparison with the above methods. Only 5 phases (with average of 150 minutes) are used to find the right identity.

#. players	Top-1	Top-3	Top-5	Top-10
311	24.1%	46.0%	59.8%	78.8%
193	31.1%	60.1%	74.1%	86.0%

This player identification task by Decroos et al. can be applied to help player scouting more objective and less time-consuming. For example, a scout can figure out an unknown player's style by seeing his or her neighbors whose styles are already known. Also, the scout can easily find alternatives who have similar styles to that of a key player leaving the team. In this sense, the above results show that our method is more practical in that it can identify players with only a small number of matches.

Chapter 6

Conclusion

In this study, we propose a deep learning approach to find effective representation of playing styles from soccer players' in-game GPS data. Without using any information of soccer-specific actions, we simply use each player's locations and velocities to generate two types of heatmaps. Our deep architecture then maps these heatmap pairs into embeddings whose similarity corresponds to the actual similarity of playing styles.

To our knowledge, this is the first study to characterize soccer players' styles using tracking data instead of event stream data. Our approach overcomes some drawbacks, such as the sparsity of event stream data.

In real-world scouting, there are many constraints to get full season data of every unknown player. Therefore, a system that can characterize players with only a small number of matches is more practical. If a scout gets only a few matches of a target player's data, the scout can figure out the player's style by seeing the known players whose embeddings are close to the target player.

Bibliography

[1] C. Anderson and D. Sally, *The Numbers Game: Why Everything You Know About Football is Wrong*, 2013.

[2] A. Bialkowski, P. Lucey, P. Carr, Y. Yue, S. Sridharan, and I. Matthews, *Large-scale analysis of soccer matches using spatiotemporal tracking data*, in IEEE International Conference on Data Mining, 2014.

[3] J. Bromley, I. Guyon, Y. LeCun, E. Sachkinger, and R. Shah, *Siamese verification using a "siamese" time delay neural network*, International Journal of Pattern Recognition and Artificial Intelligence, (1993).

[4] J. Brooks, M. Kerr, and J. Guttag, *Developing a data-driven player ranking in soccer using predictive model weights*, in ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2016.

[5] Catapult Sports. https://www.catapultsports.com/. Accessed: 2021-02-05.

[6] T. Decroos and J. Davis, *Player vectors: Characterizing soccer players' playing style from match event streams*, in European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, 2019.

[7] T. Decroos, M. V. Roy, and J. Davis, *SoccerMix: Representing soccer actions with mixture models*, in European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, 2020.

[8] T. Decroos, J. Van Haaren, L. Bransen, and J. Davis, *Actions speak louder than goals: Valuing player actions in soccer*, in ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2019.

[9] T. Decroos, J. Van Haaren, and J. Davis, *Automatic discovery of tactics in spatio-temporal soccer match data*, in ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2018.

[10] J. Diquigiovanni and B. Scarpa, *Analysis of association football playing styles: An innovative method to cluster networks*, Statistical Modelling, 19 (2019).

[11] J. Duch, J. S. Waitzman, and L. A. Nunes Amaral, *Quantifying the performance of individual players in a team activity*, PLoS ONE, 5 (2010).

[12] J. Fernandez-Navarro, L. Fradua, A. Zubillaga, P. R. Ford, and A. P. Mcrobert, *Attacking and defensive styles of play in soccer: analysis of Spanish and English elite teams elite teams*, Journal of Sports Sciences, 34 (2016).

[13] FIFA, *Laws of the Game 2017-2018*, 2017.

[14] Fitogether. https://www.fitogether.com/. Accessed: 2021-02-05.

[15] L. Gyarmati and M. Hefeeda, *Analyzing in-game movements of soccer players at scale*, in MIT Sloan Sports Analytics Conference, 2016.

[16] L. Gyarmati, H. Kwak, and P. Rodriguez, *Searching for a unique style in soccer*, (2014).

[17] H. Liu, W. Hopkins, M. A. Gómez, and J. S. Molinuevo, *Inter-operator reliability of live football match statistics from OPTA Sportsdata*, International Journal of Performance Analysis in Sport, 13 (2013).

[18] J. J. Malone, R. Lovell, M. C. Varley, and A. J. Coutts, *Unpacking the black box: Applications and considerations for using GPS devices in sport*, International Journal of Sports Physiology and Performance, 12 (2017).

[19] Opta Sports. https://www.optasports.com/. Accessed: 2021-02-05.

[20] L. Pappalardo, *PlayeRank: Data-driven performance evaluation and player ranking in soccer via a machine learning approach*, ACM Transactions on Intelligent Systems and Technology, 10 (2019).

[21] J. L. Peña and R. S. Navarro, *Who can replace Xavi? A passing motif analysis of football players*, (2015).

[22] F. Schroff, D. Kalenichenko, and J. Philbin, *FaceNet: A unified embedding for face recognition and clustering*, in IEEE Conference on Computer Vision and Pattern Recognition, 2015.

[23] StatsBomb. https://statsbomb.com/. Accessed: 2021-02-05.

[24] STATSports. https://statsports.com/. Accessed: 2021-02-05.

국문초록

축구 산업의 성장과 함께 개별 선수의 가치가 천문학적인 수준이 되었지만, 선수 파악 및 평가는 최근까지도 도메인 전문가들의 주관에 의지해 왔다. 최근, 영상 기반 이벤트 데이터를 이용하여 선수 플레이스타일을 정량적으로 파악하고자 하는 새로운 시도들이 있었다. 그러나 해당 데이터는 공을 잡은 선수의 동작만을 수기로 기록한 것이기 때문에, 한 경기에서 선수의 움직임 경향을 파악할 만큼 충분한 양이 수집되지 못했다. 본 논문에서는 축구 경기 중에 수집된 선수의 움직임 데이터로부터 선수의 플레이스타일을 효과적으로 정량화할 수 있는 딥러닝(deep learning) 기반의 방법론을 제안한다. 수동 태깅 정보 없이 선수의 위치 및 속도만을 이용하여 두 가지 유형의 히트맵을 생성하였으며, 이들을 샴 네트워크(siamese networks)에 통과시킴으로써 각 히트맵 쌍에 대한 임베딩(embedding)을 산출하였다. 실제로 움직임이 비슷한 선수들의 임베딩 벡터간 거리가 가깝도록 학습을 진행했기 때문에, 각 임베딩은 선수의 플레이스타일을 효과적으로 나타낸다고 판단된다. 본 연구에서는 기존 연구들이 고려하지 못한 오프더볼(off-the-ball) 움직임과 선수가 경기별로 부여 받은 전술적 역할을 모델에 포함시켰으며, 적은 수의 경기만으로도 선수를 효과적으로 식별할 수 있음을 실험적으로 확인하였다. 실제 축구 현장에서 생소한 선수의 데이터를 많이 확보하기 힘들다는 점을 고려해 볼 때, 이는 논문에서 제안하는 방법론이 기존 연구들보다 높은 적용 가능성을 가지고 있음을 시사한다.

주요어: 데이터마이닝, 표현 학습, 스포츠 데이터 분석, 플레이스타일, 웨어러블 추적 시스템, 위치 및 방향 히트맵, 샴 네트워크

학번: 2018-23740

감사의 글

저는 학부에서 수학을 전공했지만, 축구를 정말 좋아합니다. 대학 시절부터 K리그와 해외축구를 가리지 않고 전 세계의 축구장을 돌며 경기를 관람했고, 축구를 하다가 양쪽 무릎 전방십자인대가 모두 완전파열되어 병역 의무를 면제 받기도 했습니다. 축구에 대한 이런 저의 열정은 저를 진로 고민의 늪에 빠뜨렸고, 그 결과 데이터마이닝을 축구에 응용하는 것이 수학과 축구의 접점으로서 제가 나아가야 할 방향이라고 판단하고 서울대학교 산업공학과 석사과정에 진학하게 되었습니다.

입학 당시의 저는 타전공자로서 데이터마이닝에 관한 실질적인 지식이 전무한 상태였습니다. 그러나, 2년간의 대학원 생활을 통해 많은 것을 배울 수 있었고, 그간 배운 지식을 축구에 응용하여 부족하게나마 논문으로 남길 수 있게 되었습니다.

먼저, 연구 주제에 관한 아이디어부터 방법론에 이르기까지 좋은 지도를 해 주신 조성준 교수님께 감사의 말씀을 드리고 싶습니다. 교수님께서 축구에 대해 개인적으로도 많은 관심이 있으셨기 때문에, 방법론 뿐 아니라 도메인 측면에서도 좋은 조언을 받을 수 있었습니다. 또한, 연구에 많은 도움을 주신 데이터마이닝 연구실 동료들을 포함한 서울대학교 산업공학과의 모든 식구들께 감사드립니다. 특히 연구 방향 설정에 큰 도움이 되는 조언을 해주신 심재웅 박사과정과 궂은 실험 작업을 열심히 도와주신 김지훈 학부생에게 감사를 전합니다.

마지막으로, 본 연구는 축구 데이터 분석 기업 (주)핏투게더의 지원 하에 진행되었음을 알립니다. (주)핏투게더에서 제공받은 웨어러블 센서 기반 데이터는 전세계 기준으로도 희소성이 있고, 2019년 FIFA에서 주관한 EPTS 측정 정확도 테스트에서 세계 1위를 할 만큼 그 정확성을 보장받았습니다. 이러한 양질의 데이터는 제가 부족한 지식으로도 새로운 연구를 할 수 있었던 발판이 되었습니다. 이에 대하여 윤진성 대표님을 포함한 회사 구성원들께 감사드립니다.