Deep Learning-based Activity Recognition in Soccer Using IMUs

Bachelor's Thesis in Medical Engineering

submitted
by

Yannis Maag

born 03.11.2001 in Ochsenfurt

Written at

Machine Learning and Data Analytics Lab
Department Artificial Intelligence in Biomedical Engineering
Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU)

in Cooperation with

adidas AG

Advisors:	Maike Stoeve, M.Sc., Rebecca Lennartz, M.Sc., Burkhard Duemler (adidas AG), Prof. Dr. Bjoern Eskofier
Started:	02.05.2024
Finished:	11.11.2024

Erlangen, den 11.11.2024

Übersicht

Detaillierte Leistungsanalysen werden immer wichtiger in vielen Sportarten, insbesondere im Fußball. Derzeit werden dafür vor allem sehr teure Geräte wie GPS-Westen oder Hochgeschwindigkeitskameras eingesetzt. Diese Technologien sind jedoch für Amateurspieler oft zu teuer. Es stellt sich daher die Frage, wie auch Sportler mit einem geringeren finanziellen Budget vergleichbare Einblicke in ihre Leistungsdaten erhalten können [Cup22]. Die vorliegende Arbeit befasst sich mit dieser Frage und nutzt einen auf Inertialsensorik basierenden Lösungsansatz, bei dem jeweils ein Inertial Measurement Unit (IMU) in die Sohle der Fußballschuhe integriert wird.
Bestehende Modelle können derzeit meist nur allgemeine Aktivitäten wie Pässe, Schüsse oder Rennen unterscheiden und sind gleichzeitig nur eingeschränkt auf reale Spielsituationen anwendbar [Kon22; Alo18]. Daher besteht ein wachsendes Interesse an spezifischeren Leistungsanalysen, die auch außerhalb von Laborbedingungen funktionieren. Es wurde gezeigt, dass traditionelle Machine Learning (ML) Modelle in der Lage sind zwischen diesen allgemeinen Aktivitäten zu unterscheiden [Kon22; Alo18]. Ansätze aus dem Bereich Deep Learning (DL) haben im Vergleich jedoch deutlich bessere Ergebnisse erzielt [Sto21; Cup22]. Das Ziel dieser Studie ist es daher, einen DL-Ansatz zu entwickeln, der auch in realistischen Spielsituationen in der Lage ist, fußballspezifische Aktivitäten zu klassifizieren.

Es werden Daten von über 800 Spielern verwendet. Dabei sollen Aktivitäten wie Dribbling, die Art von Pässen und die Art von Schüssen klassifiziert werden. Zudem wird erfasst, ob diese Aktivitäten mit dem linken oder rechten Fuß ausgeführt werden. Die Aktivitäten werden mithilfe eines Algorithmus zur Spitzenerkennung extrahiert. Ein kombinierter Ansatz aus Random Undersampling und Adaptive Synthetic Sampling Approach (ADASYN) wird eingesetzt, um den stark unausgewogenen Datensatz zu balancieren. Zwei DL-Modelle, ein Convolutional Neural Network (CNN) und ein CNN-Long Short-Term Memory (LSTM), werden implementiert und optimiert, mit besonderem Fokus auf die Leistungsbewertung unter Labor- und realen Spielbedingungen.
Die Ergebnisse zeigen, dass das CNN mit einem gewichteten F1-Score von 92,54 % für die genauere Aktivitätserkennung und 97,53 % für die Unterscheidung, mit welchem Fuß die Aktivität ausgeführt wird, die beste Leistung erzielt.

Aufbauend auf diesen Ergebnissen bietet diese Arbeit wertvolle Einblicke in die detailliertere Spieleranalyse.

Abstract

Detailed performance analysis is becoming increasingly important in many sports, especially in soccer. Currently, very expensive devices such as GPS vests or high-speed cameras are mainly used for this purpose. However, these technologies are often too expensive for amateur players. Therefore, the question arises of how athletes with a lower financial budget can also gain comparable insights into their performance data [Cup22]. The present thesis deals with this question and uses a solution based on inertial sensor technology, in which an Inertial Measurement Unit (IMU) is integrated into the sole of each soccer shoe.
Existing models can currently only differentiate between general activities such as passing, shooting or running and at the same time can only partially be applied to real game situations [Kon22; Alo18]. Therefore, there is a growing interest in more specific performance analyses that work outside of laboratory conditions. It has been shown that traditional Machine Learning (ML) models can distinguish between these general activities [Kon22; Alo18]. However, approaches from the field of Deep Learning (DL) have achieved better results in comparison [Sto21; Cup22]. Therefore, this study aims to develop a DL approach that can classify soccer-specific activities even in realistic game situations.

Data from over 800 players is used. The aim is to classify activities such as dribbling, the type of pass and the type of shots. In addition, it is determined whether these activities are performed with the left or right foot. The activities are extracted using a peak detection algorithm. A combined approach of random undersampling and Adaptive Synthetic Sampling Approach (ADASYN) is used to balance the highly imbalanced data set. Two DL models, a Convolutional Neural Network (CNN) and a CNN-Long Short-Term Memory (LSTM), have been implemented and optimized, with a particular focus on performance evaluation under laboratory and real game conditions.
The results show that the CNN achieves the best performance with a weighted F1 score of 92.54 % for activity recognition and 97.53 % for distinguishing by which foot the activity is performed.

Building on these results, this paper provides valuable insights into more detailed player analysis.

Contents

Chapter 1

Introduction

In recent years, data analysis has become a pillar of modern team-based sports, fundamentally transforming how teams approach both preparation and in-game strategies [Sar21; Lee22; Jha22; Mic23; Nic24]. This development is particularly evident in soccer, where performance analysis has become essential for evaluating players and matches. In professional soccer, advanced tools such as high-speed cameras, GPS vests, and video analytics systems are used to monitor key performance indicators such as player positions, distance covered, speed, power, intensity, and heart rate. These insights are crucial for optimizing performance, player health for injury prevention, and refining tactical approaches for future matches [Cup22; Kon22; Jha22]. For example, during the 2010 FIFA World Cup, it was identified that England's opponents were increasingly utilizing long passes, which exposed weaknesses in the English defense. The German team effectively used this information in their round of sixteen match against England, where goalkeeper Manuel Neuer assisted a goal by delivering a long pass over the English defense to Miroslav Klose, who scored the opening goal [Per13].
Given the considerable financial resources available in professional soccer, professional clubs can afford to invest in this expensive equipment [Kon22]. However, for amateur athletes with limited budgets, the question arises: How can they benefit from similar insights without the need for big investments into expensive technology?

A common method for performance and activity analysis is the use of data from an Inertial Measurement Unit (IMU). IMUs are ideal for this purpose due to their small size, low cost, and high accuracy [Cup22]. These sensors allow an easy-to-use, unobtrusive data acquisition for Human Activity Recognition (HAR) [Man23]. HAR means the automatic recognition of physical human activities [Sch19]. It has been shown that IMU data recorded during a soccer drill can be used to identify different activities. When the sensors are placed on a player's foot or hand, it is possible to distinguish between jogging, sprinting, passing, shooting, and jumping [Cup22; Kon22; Sto21; Hos17].

The intersection of HAR and soccer performance analysis opens up new opportunities to better understand player behaviour on the field. Beyond general activity classification, HAR techniques can provide insights into more differentiated actions, such as the type of pass or whether a player is dribbling while jogging [Cup22].
Traditional Machine Learning (ML) methods have achieved considerable accuracy in classifying these activities, with reported accuracies of up to 87 % [Alo18]. In contrast, more advanced Deep Learning (DL) approaches have shown even greater potential, achieving accuracies of up to 98.3 % [Cup22]. However, current methods often lack the granularity required for detailed individual assessment, and many approaches still rely on shallow classification structures, making the development process manual, subjective and time-consuming [Cup22]. DL has been used to address the issues associated with traditional ML, improving accuracy and overcoming some of its limitations. Nevertheless, despite achieving notable results, DL approaches typically focus on general activities and make limited use of real game data. This restricts their overall applicability [Sch19; Sto21].

Despite these advancements, a gap remains in the combination of using DL models to classify more specific metrics, incorporating real game data where the complexity and variability of movements are higher. Therefore, the aim of this work is to develop a DL approach capable of classifying more specific metrics such as dribbling, type of pass, type of shot, and automatic left and right foot recognition using two IMU sensors embedded in the sole of each soccer shoe.
The DL model seeks to accurately classify these metrics, even in game-like situations, thus providing detailed insights into player performance and activities.

Chapter 2

Related Work

2.1 Overview of IMU-based Activity Recognition

In recent years, IMU data has emerged as a powerful tool for HAR, owing to its versatility and practical applications. Integrating IMUs into various devices has expanded their utility. For instance, IMUs are now commonly found in smartwatches and sports equipment, such as the balls used in the 2024 European Championship in Germany [Mic23; Kon22; adi23]. Their application extends beyond these devices to textiles like clothing and shoes, or even as standalone sensors attached to specific body regions [Kru24; Sch19; Lar23]. This widespread integration highlights the flexibility of IMUs, making them increasingly popular for recognizing a broad spectrum of activities. From everyday tasks such as sitting, lying down, walking, cycling, and dishwashing to more complex sport-specific movements, IMU-based HAR provides valuable insights [Sch19; Hsu18]. Consequently, this technology has found applications across numerous sports, including rugby, tennis, boxing, swimming, and soccer [Nic24; Mic23; Jay24; Che23; Cup22].

Given the broad applicability of IMU data, both traditional ML methods and DL techniques have been extensively employed for activity classification. Existing literature has predominantly focused on distinguishing between a wide range of soccer activities that vary in movement patterns and intensity, such as shooting, passing, heading, dribbling, jogging, sprinting, changing direction, and jumping [Alo18; Rei21; Sch19; Kon22; Hal23; Lar23; Cup22; Sto21; Hos17]. As a consequence, classification models have often achieved good to excellent accuracy in these scenarios [Cup22; Alo18]. However, they primarily focus on more predictable and repetitive soccer activities, often recorded in controlled laboratory settings. In contrast, real game scenarios involve fast, irregular, and non-repetitive movements, such as ball contacts under opponent pressure. Furthermore, players execute these movements differently based on whether the ball contact is with their dominant or weaker foot, which adds another layer of complexity [Alo18].

Given this complexity, the accurate classification of soccer-specific activities based on IMU data requires refined approaches. Therefore the following section provides a comprehensive overview of the current literature on soccer-specific activity recognition using IMU signals, highlighting the various traditional ML and DL approaches that have been explored.

2.2 Traditional Machine Learning Approaches

Alobaid and colleagues conducted a study using a smartphone's accelerometer to classify activities such as sprinting, passing, heading, shooting, and dribbling. The study achieved an accuracy of 87 % using a Support Vector Machine (SVM) when all features were considered simultaneously. Data were collected from 16 players aged 18 to 35, who performed only the specific activities to be classified. The smartphone, equipped with an integrated accelerometer, was secured in a belt-mounted phone pouch for data collection [Alo18]. However, the study did not evaluate the practicality or feasibility of this approach in real-world settings, such as during training sessions or actual games, leaving it unclear whether wearing a sensor in the abdominal area and using a smartphone is the most effective method for data collection in such environments [Kon22].

While Alobaid and colleagues focused on classifying specific activities using a smartphone-based approach, Reilley and colleagues introduced a new metric, called Change of Direction (COD), aimed at providing a more comprehensive analysis of player dynamics during a game. This metric extends traditional physiological measurements like sprinting and jumping by focusing on how often and when a player changes direction. Such insights can offer a deeper understanding of player load and support the development of personalized training and recovery strategies for individual athletes. The study utilized data from a sensor that provides both GPS and IMU data, from which a new variable, the GPS-COD angle, was derived. The sensor was placed in specially designed vest pockets located on the players' backs between their shoulder blades. Data were collected from 23 Premier League academy soccer players over the course of 10 competitive matches. Using this data, the researchers trained a random forest classifier to automatically detect CODs greater than 45 degrees. The model achieved an accuracy of 84 % [Rei21].

Building on the study of general player dynamics and loading, Schuldhaus concentrated on a more detailed soccer-specific analysis. His doctoral thesis focused on distinguishing between two specific soccer actions, the full-step kick and the side-foot kick while identifying the event leg used in each action. To collect data, Schuldhaus embedded an IMU sensor in the sole of an Adidas F50 adizero shoe. 11 male participants, aged between 20 and 30 years, were recruited to perform 14 predefined exercises. The data gathered from these exercises was then used to train a ML model. In addition to this controlled data, real match data was collected from 17 other male players during an 11-a-side soccer match, with only one of these players having participated in the initial data collection phase.
To distinguish between the soccer-specific actions, Schuldhaus developed two algorithms: one based on a hierarchical architecture and the other focused on analyzing the phases of the kicking action. A k-Nearest Neighbours (k-NNs) algorithm was employed for the classification of the event leg, achieving an accuracy of over 99 % in both approaches. Using both Naive Bayes (NB) and SVM classifiers, an accuracy exceeding 94 % was achieved in distinguishing between full-instep and sidefoot kicks, regardless of the classification approach applied.
The hierarchical model was further tested on real match data, representing actual game scenarios. Since these real match conditions were not included during the training phase, the model's accuracy in detecting key events from the sensor data decreased by 5 %. The decline in accuracy and the rise in false positive rates suggest that the variability of real game conditions introduced factors not covered during training. As a result, Schuldhaus recommends integrating real match data into the training process of future systems to improve their overall performance and reliability in authentic match conditions [Sch19].

While Schuldhaus' work provided deep insights into specific soccer actions and their recognition under real game conditions, Kondo and colleagues' paper shifted the focus to the evaluation of sensor placement and its impact on the classification of different soccer activities. They include metrics such as shooting, passing, heading, running, dribbling, and performing kick tricks. Their research compared three different sensor placements, ultimately determining that positioning the sensor inside the ankle yielded the most accurate results. By employing an ensemble bagged tree classification method, the researchers achieved an accuracy rate of 78.7 % when classifying all six metrics simultaneously. The data were collected from 10 right-footed players aged 20 to 23 years, although data from only seven players were usable for analysis. Importantly, the data were recorded under controlled experimental conditions with no real game data being used in the study [Kon22].

The presented papers demonstrate how well traditional ML algorithms perform across various individual metrics. It becomes evident that a differentiated recognition of activities is generally feasible. However, these papers also highlight the limitations of traditional ML, as the classifiers tend to perform less effectively when distinguishing between multiple metrics simultaneously. Additionally, most models were trained primarily on laboratory data, which further underscores these limitations.

2.3 Deep Learning Approaches

In contrast to the previously discussed approaches, Stoeve and colleagues employed a DL methodology to distinguish between shots and passes in soccer using data collected from an IMU sensor. This sensor was also embedded in the sole of an Adidas F50 adizero shoe, just as in Schuldhaus' doctoral research. The study utilized a comprehensive dataset collected from 836 players, categorized according to German-defined age groups, ranging from U12 to adult athletes. The data collection process was extensive, incorporating both predefined exercises conducted in controlled laboratory settings and real-world game scenarios, thus yielding a highly diverse dataset.
Three different types of neural networks were developed in the study and their performance was then evaluated against a SVM classifier already established in another paper [Sch19]. The evaluation process involved testing the classifiers progressively on data that transitioned through three stages, each stage increasingly reflecting real-game conditions rather than controlled laboratory environments. The results showed that the SVM was no longer able to reliably distinguish between the three classes. In contrast, the DL models performed consistently very well, with the Convolutional Neural Networks (CNNs) achieving the highest effectiveness, as evidenced by a weighted F1 score of 93 % [Sto21].

Building on Stoeve and colleagues' use of DL to classify soccer-specific activities, Hossain took a slightly different approach by placing the sensors on the wrist rather than in the shoe. With this wrist-worn sensor setup, he wanted to distinguish between multiple metrics such as passing, shooting, running, standing and dribbling, extending Stoeve's range of metrics. In this study, dribbling was defined as performing a specific trick, either a cut-over or step-over move. A Restricted Boltzmann Machine (RBM) was utilized for classification, achieving an overall accuracy of 86.54 % when all classes were classified simultaneously. Notably, high accuracy was observed for the walking, running, and standing classes, while the lowest accuracy was recorded for dribbling events.

Data collection was conducted using an IMU sensor attached to the player's non-dominant wrist. The study exclusively used acceleration data measured in the x, y, and z directions. The dataset was gathered from six players, each participating in ten one-hour sessions of a five-a-side game [Hos17].

In contrast to Hossain's wrist-based sensor approach, Larsen's study employed a broader DL methodology to classify various soccer actions, including passing, first touch, dribbling, positioning, and head scanning. For this purpose, players were equipped with 17 IMU sensors, although only seven of these sensors were used for the final analysis. The sensors were strategically positioned on the head, sternum, right wrist, right calf, right foot, left calf, and left foot, reflecting common sensor placements used by major player tracking companies in soccer.
The data utilized included 13 raw outputs from the sensors, which comprised accelerometer, gyroscope, and magnetometer readings (X, Y, Z) as well as quaternion outputs (W, X, Y, Z). From these raw outputs, manual features were extracted to serve as inputs for the classification algorithms. The classification process was divided into two distinct parts: the first part utilized Long Short-Term Memory (LSTM) networks to differentiate among passing, first touch, dribbling, and positioning, while the second part employed a Deep Neural Network (DNN) to predict head scanning.
The LSTM model achieved an accuracy of 75 % for the first part, while the DNN model achieved an accuracy of 78 % for head scanning. The analysis revealed that sensor placement on the right calf provided the best results for the LSTM-based classification of soccer actions, whereas sensor placement on the head was most effective for the DNN-based head scanning classification. The dataset consisted of recordings from 17 players aged 12 to 26, who participated in sessions within a Goal Station Focus 360° environment, with no real match data being recorded [Lar23].

In another paper in the field of soccer activity classification, Cuperman made remarkable progress by applying DL techniques by distinguishing between five specific soccer activities: jogging, sprinting, passing, shooting and jumping. By employing a combination of CNN and LSTM networks, the model achieved an accuracy of up to 98.3 %.
Data for this study were collected from 11 players, each equipped with five IMU sensors placed on the pelvis, right thigh, left thigh, right shank, and left shank. These sensors recorded accelerometer and gyroscope data. The experiments were carefully designed to simulate conditions similar to those in real soccer matches.
Before classifying specific activities, the model first distinguished between high-activity and low-activity periods. Classification into specific activities was only performed during high-activity periods, enhancing the model's accuracy. Additionally, participants were required to walk or stand

still briefly before and after each activity; these low-activity intervals were removed from the dataset before training the model.
Cuperman explored various model architectures, including those based solely on CNNs, Recurrent Neural Networks (RNNs), and combinations of both. The highest accuracy was achieved with a model combining CNN and bidirectional LSTM networks [Cup22].

The research findings highlight that DL algorithms have substantially greater potential for HAR than traditional methods. Unlike traditional algorithms, DL can effectively utilize real game data and achieve equal or even better accuracy. This advantage makes DL particularly well-suited for handling complex and varied metrics. This alignment with our classification goals emphasizes the benefits of DL in delivering more accurate and promising results.

Chapter 3

Theoretical Background

Given the frequent references to ML and DL in the preceding section, it is essential to distinguish between these fundamental concepts clearly. This section will provide a concise overview of the hierarchical relationship between ML and DL, followed by an introduction to both approaches. Special attention will be given to the DL models employed in this study, specifically CNNs, LSTMs, and their combination.

3.1 Hierarchical Relationship between ML and DL

Artificial Intelligence (AI) encompasses a broad range of methods and techniques to enable machines to perform tasks that typically require human intelligence [Jan21].
Within this spectrum, ML plays a central role. ML focuses on developing algorithms and models that allow computer systems to enhance their performance on specific tasks through experience. In this context, manually created features are often used as input for the classifiers to optimize model performance [Jan21]. Traditional ML techniques include algorithms such as SVMs, Decision Trees (DTs), and k-NNs, which are often categorized under ”Traditional” ML [Cha18]. These algorithms excel in tasks such as credit scoring and natural language processing by identifying patterns in high-dimensional data without the need for explicit programming [Jan21].
DL is a specialized subset of ML that focuses on Artificial Neural Networks (ANNs). More particularly, DNNs are used here, representing a specific type of ANNs. Unlike traditional ML algorithms, DL models operate with multiple processing layers that enable them to automatically extract and learn complex features directly from raw data [Cha18]. This integrated approach enhances performance in object and speech recognition [Cup22]. Despite these advanced ca-

pabilities, DL models still encounter limitations, especially in areas requiring strong AI, such as tasks involving literal understanding and intentionality [Jan21]. Figure 3.1 further illustrates this relationship.

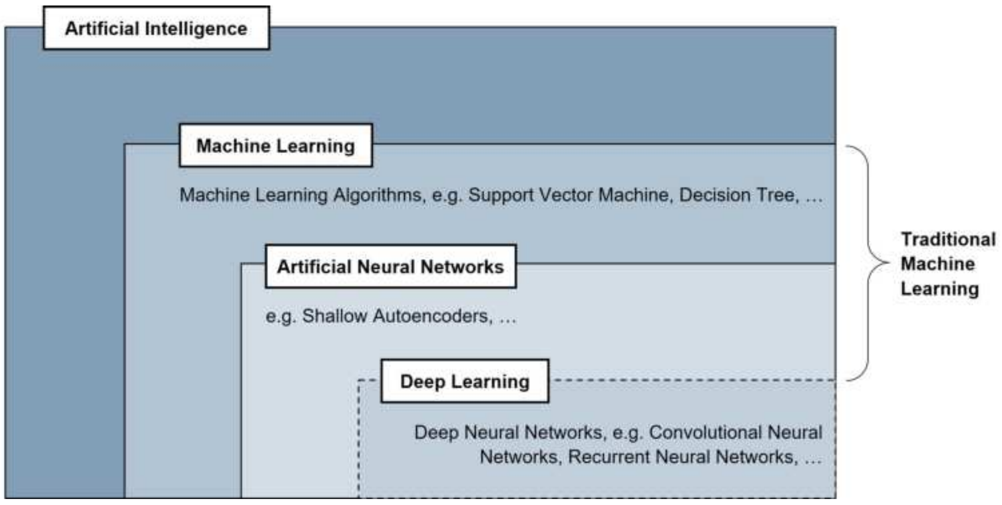

Artificial Intelligence
Machine Learning
Machine Learning Algorithms, e.g. Support Vector Machine, Decision Tree, ...
Artificial Neural Networks
e.g. Shallow Autoencoders, ...
Deep Learning
Deep Neural Networks, e.g. Convolutional Neural Networks, Recurrent Neural Networks, ...
Traditional Machine Learning

Figure 3.1: Visualization of the hierarchical relationship between ML and DL. Adapted from [Jan21].

3.2 Traditional Machine Learning

Traditional ML algorithms are typically categorized into supervised and unsupervised learning. Within supervised learning, tasks are further divided into classification and regression [Cha18]. Classification is concerned with predicting discrete class labels, or the category to which the data belongs, such as gender or a type of fruit. In contrast, regression focuses on estimating continuous outcomes by modelling how strong the relationship between input characteristics and outcomes is, such as property prices or profits. Formally, regression can be represented as a function $y = f(X)$, where X represents the input variables and y the continuous output variable [Pap20]. Common ML models for classification include SVM, Discriminant Analysis, and Naive Bayes Classifiers. For regression, typical models include SVMs with regression capabilities, and DTs. Unsupervised learning methods, such as K-means Clustering and Principal Component Analysis, focus on discovering patterns and dimensionality reduction without labelled outcomes. These approaches are used to tackle association and clustering problems [Cha18].

Unfortunately, the efficient processing of raw data often requires manual feature extraction, which involves deriving features from the time and frequency domain. This process includes calculations such as minimum, maximum, mean, standard deviation, variance, and percentiles. More advanced methods such as the Fourier transform and the discrete cosine transform are usually used in the frequency domain [Alo18]. However, these traditional machine learning methods have notable limitations. The use of "hand-crafted features" can be both subjective and time-consuming, as each feature must be carefully selected and adjusted [Cup22]. This can lead to a loss of information, as features with higher information content are selected, while other features with lower information content are not taken into account. In addition, a considerable level of understanding and expertise is essential [Cha18].

Consequently, traditional ML methods are increasingly being outperformed by DL models, which automatically learn and optimize features directly from raw data [Shi23; Sto21]. However, traditional ML models still offer advantages in terms of lower complexity and reduced data requirements for training and development, as they do not require high-performance computing units or access to large datasets. Additionally, they are more interpretable, allowing people to understand and follow the decision-making process of the algorithms. Therefore, some HAR classifications in soccer continue to rely on these methods [Ahm23].

3.3 Deep Learning

DL involves learning hierarchical representations of data using architectures with multiple hidden layers. In DL models, input data is processed through several layers, where each layer gradually extracts more complex features and passes relevant information to the next layer. The initial layers capture basic, low-level features, while the deeper layers combine these to form more complex and abstract representations [Shi23].

Over time, numerous methods and model architectures have been developed within the field of DL. These models can generally be categorized into two main types: discriminative (supervised) and generative (unsupervised) approaches. Prominent examples of generative models include Generative Adversarial Networks (GANs) and Autoencoders. In the realm of supervised learning, CNNs and RNNs are widely recognized, with LSTMs networks playing a particularly important role [Shi23]. In this work, CNNs and LSTMs will be used. Therefore, the following sections will focus on these two structures and their functionalities.

Before delving into these advanced architectures, it is essential first to understand the structure and functionality of an ANN that serves as a foundational architecture for deep learning [Shi23]. Therefore, the next section will overview a ANN's basic components and operations.

3.3.1 ANNs

ANNs are modelled after the functioning of biological neurons. The core components of ANNs are artificial neurons, which are linked by weighted connections [Zay18].
The operation of a single neuron is depicted in Figure 3.2. Essentially, a neuron receives one or more input signals, each multiplied by an individual weight. These weighted inputs are then summed within the neuron. The bias b_k affects the net input to the activation function, either increasing or decreasing it depending on whether it is positive or negative. The output is then generated through an activation function. This function primarily limits the amplitude range of the output signal to a finite value, while also introducing non-linearity [Zay18]. This non-linearity enables the ANN to detect and model more complex patterns [Ahm23].

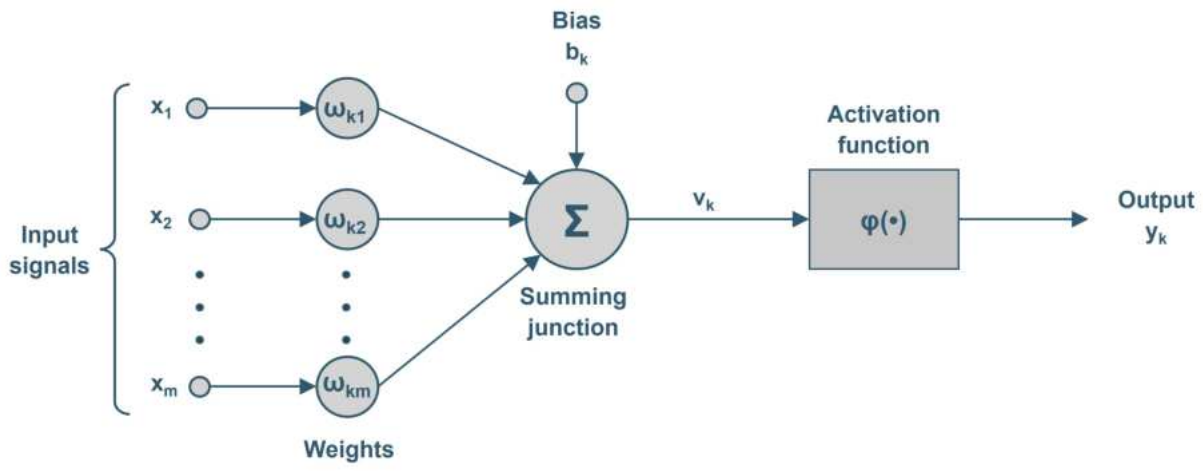


Figure 3.2: Basic principle of a artificial neuron: The input x is multiplied by the weights w. The weighted inputs and bias b are summed and then transformed through an activation function to produce the final output y. Adapted from [Zay18].

Typical activation functions are the sigmoid function, which scales the output between 0 and 1 [Zay18], and the Rectified Linear Unit (ReLU) function, returning the input if it is greater than 0, otherwise it outputs 0 [Che23]. The activation functions are shown in the Figures 3.3 and 3.4, respectively.

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

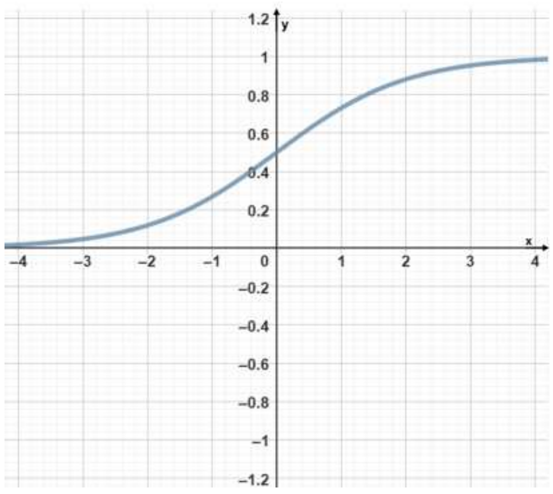


Figure 3.3: Sigmoid activation function. The values are scaled between 0 and 1.

$$\mathrm{ReLU}(x) = \begin{cases} 0 & \text{if } x \leq 0, \\ x & \text{if } x > 0 \end{cases}$$

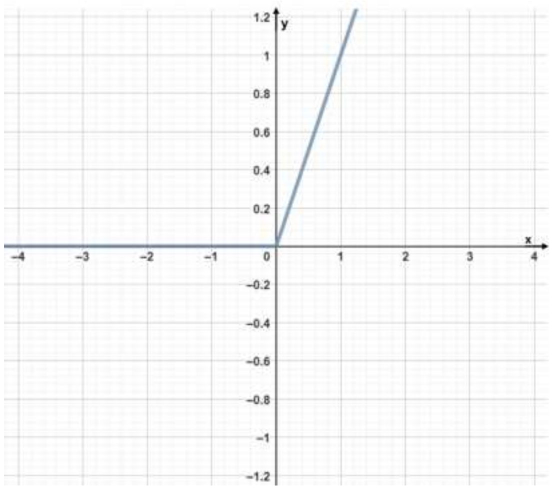


Figure 3.4: ReLU activation function. Outputs the input value if greater than 0, otherwise 0.

The output y on the neuron k can mathematically be described as

$$y_k = \varphi\left(\sum_{i=1}^{m} x_i w_{ki} + b_k\right)$$

where

- $x_1, x_2, x_3, \ldots, x_m$ are the input's signals,
- $w_{k1}, w_{k2}, w_{k3}, \ldots, w_{km}$ are the respective weights of neuron,
- b_k is the bias, and
- φ is the activation function [Zay18].

These neurons are organized into layers: a typical ANN includes an input layer, a hidden layer, and an output layer. The function of each neuron varies depending on the layer in which it is located. For example, the neurons in the input layer transmit information to the hidden layers, while the neurons in the hidden layers transmit information to the output layer [Han18]. The hidden layers can extract higher-level features that are eventually passed to the output layer. The output layer, which represents the number of possible outputs, is finally responsible for categorization tasks [Che21].

Nevertheless, a key question remains: How does the ANN learn? Training involves calibrating the weights and biases, which are the connections between neurons [Zay18]. Low weights weaken the signal, while high weights strengthen it. For example, a weight of 0 would result in no signal being forwarded, thereby having no real impact on the network. The challenge is to determine the appropriate weights to achieve the desired output [Han18].
To do this, an error function is used to calculate how much the final outputs of the network deviate from their target values [Lil20]. A common choice is to use the categorical cross-entropy loss:

$$Loss(H(y, p_i)) = -\sum_{i=1}^{N} y_i \log(p_i)$$

where p_i is the predicted value, y_i the corresponding real result and N the number of possible output classes [Mos21]. In this context, backpropagation is a method for calculating the gradient of this error with respect to each weight. This is done based on the current architecture of the network. It starts with the output units by calculating the derivative of the error function. The error signals then flow backwards through the network, layer by layer. For each non-output layer, the error signal at the neuron is calculated as a function of the error signals in the following layer and the synaptic weights. In this way, the network can iteratively adjust each weight to minimize the total error [Lil20].
While the backpropagation calculates the gradients of the error with respect to the weights, the actual minimization of the error requires an optimization method. For this purpose, the gradient descent method is usually used [Han18].

The gradient descent method is a technique for finding the lowest point of a cost function, which represents the difference between the predicted value and the actual value. The machine begins with any weight value and gradually adjusts it to minimize the cost, moving down the graph until the cost reaches a minimum. This minimizes the error between the predicted value and the actual output, without complicated mathematical calculations, thereby validating or adjusting the initial weight [Han18]. This learning process of ANNs is illustrated in Figure 3.5.

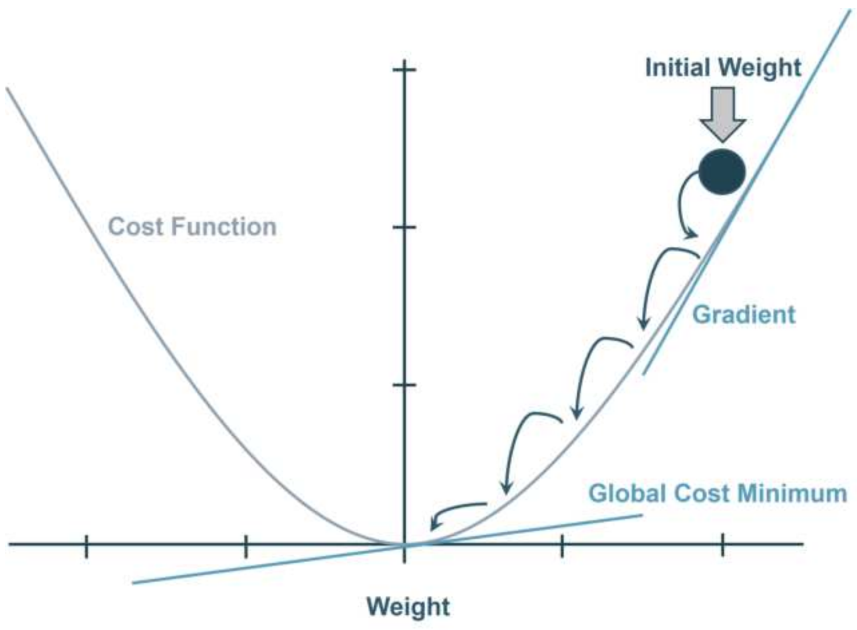


Figure 3.5: Schematic illustration of the gradient descent method. Initially, an arbitrary value for the weight is selected. The weighting is then adjusted iteratively in order to converge towards the minimum of the cost function successively. Adapted from [Han18].

3.3.2 CNNs

CNNs were originally developed for tasks in artificial vision and image processing [Cup22]. However, they have also proven to be highly effective in other domains such as object recognition, speech recognition, and time-series tasks. Fundamentally, CNNs are feedforward Neural Networks (NNs) that utilize convolutional structures to learn, identify, and extract features from data automatically. The primary components of a CNN include the convolutional layer, pooling layer, and fully connected layer and can be seen in Figure 3.6 [Shi23].

Convolutional Layers

The convolutional layer is essential for extracting various features from input data through convolution operations. In this process, the input data, structured as a matrix, is combined with small filter matrices known as convolutional kernels. Multiple convolutional layers are typically employed to merge a broad range of features into a single feature set, which is then passed on to subsequent layers for further processing [Shi23].
Convolutional kernels, which are weight matrices usually sized 3x3, 5x5, or 7x7, slide across the entire input matrix, applying convolution operations to each subregion, or patch [Shi23; Liv20].

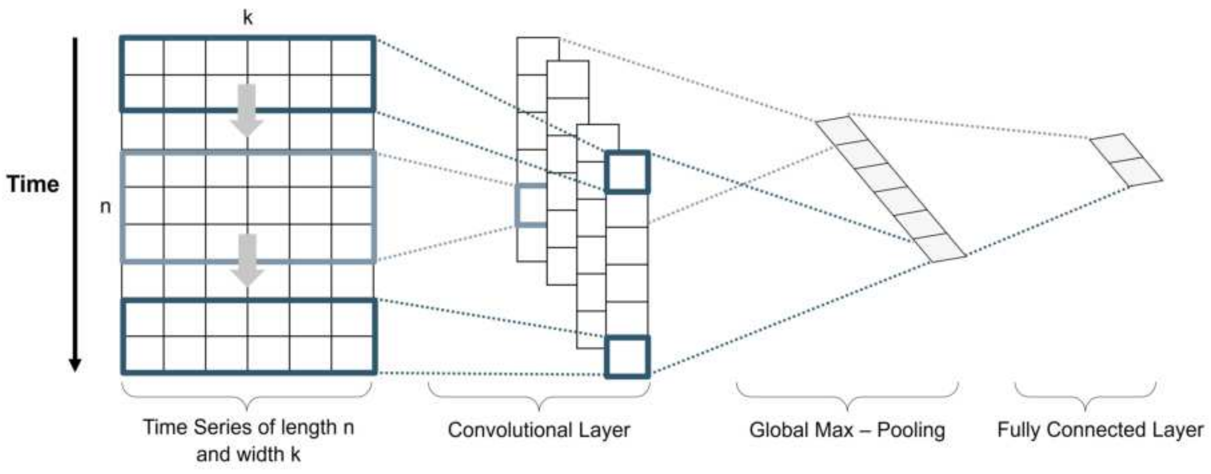


Figure 3.6: Time series classification with CNN. The Figure shows the primary components of a CNN including the convolutional layer, pooling layer, and fully connected layer [Shi23]. Adapted from [Gra19].

This produces a new convolved matrix, where each element reflects a feature value determined by the kernel's size and weights. By utilizing different convolutional kernels, the model can generate multiple convolved features that are often more informative than the raw input data, leading to improvements in the model's performance [Liv20]. Figure 3.7 illustrates the basic convolution process.

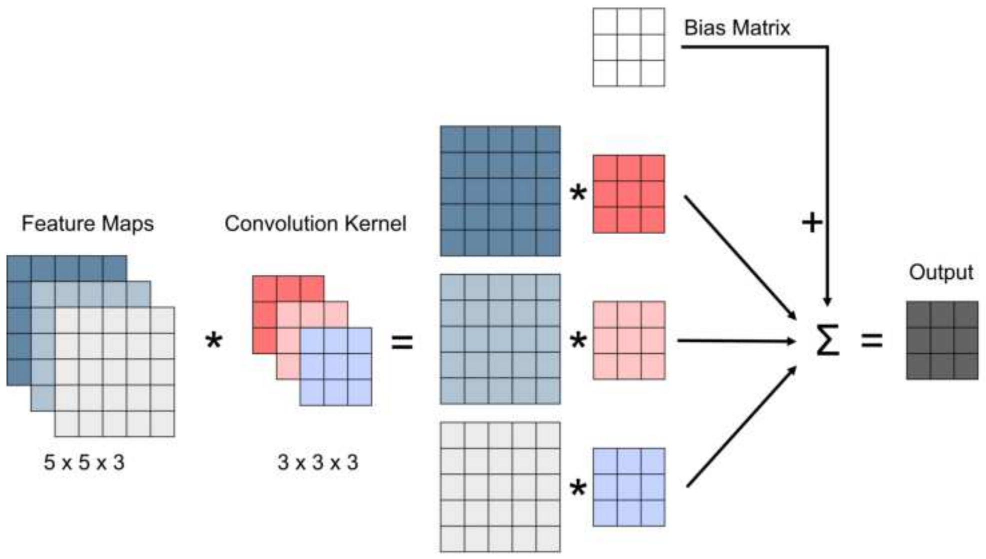


Figure 3.7: Schematic of the convolution process. Each feature map is convolved with a kernel, producing a new output matrix. Adapted from [Che21].

After the convolutional layers, a nonlinear activation function such as ReLU, as in Figure 3.4, is typically applied, as it can enhance network performance by accelerating training during gradient descent optimization [Liv20; Che23]. This is followed by a pooling layer to refine the extracted features further [Liv20].

Pooling Layers

The pooling layer is typically responsible for reducing the number of connections in the network by performing downsampling on the convolved features. By extracting specific values from the input data, this downsampling process reduces the dimensionality of the data that enters the pooling layer. The primary purpose of the pooling layer is to decrease computational complexity and address issues related to overfitting [Shi23].
Similar to the convolutional layer, the pooling layer uses a small sliding window that processes each patch of the convolved features and outputs a new value based on a specific operation defined for the pooling layer [Liv20]. For instance, Max Pooling selects the maximum value, while Average Pooling calculates the average value within each patch. This approach produces output features that are more robust to distortions and errors in individual neurons, as minor changes in the input do not really affect the pooled output values [Liv20; Shi23]. The methods of Max Pooling and Average Pooling are illustrated in Figure 3.8.

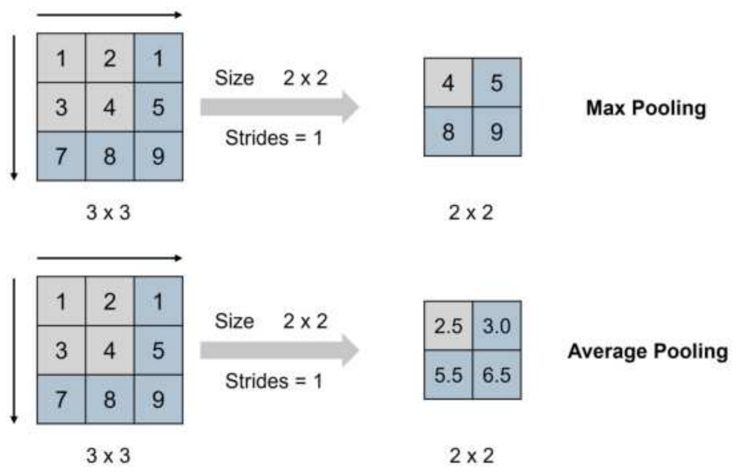


Figure 3.8: Max Pooling and Average Pooling: The grey sliding window moves over the input, performs the respective operation and combines the result into a single value. Adapted from [Che21].

Fully Connected Layers

Usually, the Fully Connected (FC) layer is found at the end of a CNN architecture. Here, each neuron is connected to all neurons of the previous layer. The input to the FC layer is a vector created by flattening the feature maps from the last pooling or convolutional layer. As the network's classifier, the FC layer plays a crucial role in prediction making [Shi23]. The rough structure can be seen in Figure 3.9.

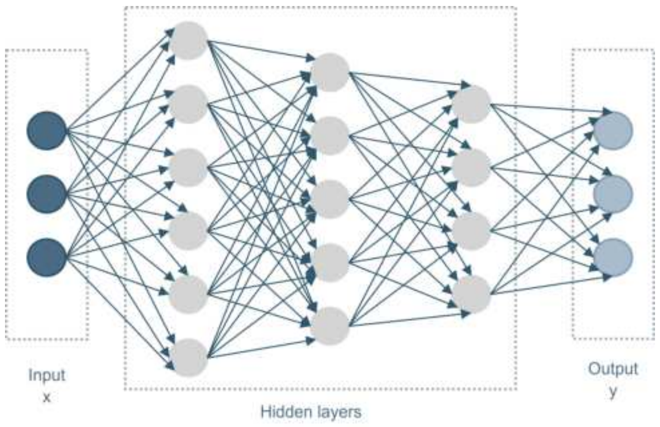


Figure 3.9: The architecture of a fully connected layer. The input vector x is the result of the flattened output of the pooling layer. Each neuron is connected to all neurons in the previous layer. The output vector y represents the different classes for classification. Adapted from [Zhu20].

3.3.3 LSTMs

LSTMs are an advanced type of RNNs [Shi23]. RNNs are equipped with internal memory and feedback connections, allowing them to process sequential data by treating each input as an independent entity while also considering the temporal order [Shi23; Liv20]. This enables RNNs to capture time-dependent patterns in the data [Liv20]. However, RNNs face challenges, particularly the vanishing gradient problem, which restricts their ability to learn long-term dependencies. As a result, they struggle to retain and utilize information over long sequences [Shi23].
To overcome these limitations, various models have been developed, with LSTMs being one of the most prominent. LSTMs address the vanishing gradient problem by storing valuable information in memory cells and discarding unnecessary data, leading to generally better performance than classical RNNs [Liv20; Shi23].

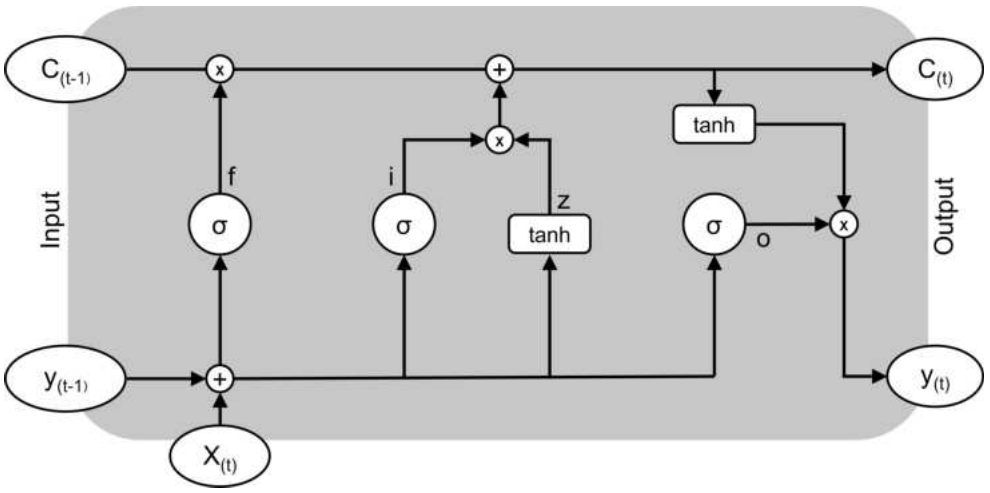


Figure 3.10: LSTM layer architecture: In the Figure, the input is represented by $X_{(t)}$ at time t, and the output is represented by $y_{(t)}$. $C_{(t)}$ is the cell state/memory. σ is the sigmoid function and addition and multiplication are represented using $+$ and x symbols. Adapted from [Yad24].

LSTM Layers

The basic LSTM layer architecture can be seen in Figure 3.10. In each computation step of an LSTM, three inputs are processed:

- the new input signal X^t,
- the previous hidden state Y^{t-1}, and
- the previous cell state C^{t-1}.

The hidden state Y^{t-1} represents the short-term memory, playing a crucial role in processing the current input. On the other hand, C^{t-1} serves as the cell's long-term memory, carrying information across time steps [Isl20].
The key components of the LSTM cell and their functions are as follows:

- **Forget gate f^t:** Determines how much of the information from the previous cell state C^{t-1} should be retained or discarded. This is done using a sigmoid function as in Figure 3.3, producing values between 0 and 1 to indicate the degree of forgetting [Cup22].
- **Candidate cell state z^t:** Represents the new information that might be added to the cell state [Isl20]. This candidate state is generated by applying a tanh function, defined as $tanh(x) = \frac{2}{1+e^{-2x}} - 1$. This scales the output values in the range $[-1, 1]$ [Che23].

- **Input gate i^t:** Decides the proportion of the candidate cell state z^t that should be incorporated into the new cell state. Similar to the forget gate, i^t is calculated using a sigmoid function [Cup22].
- **Output gate o^t:** Regulates which part of the updated cell state C^t will be passed on as the new hidden state Y^t to the next time step [Cup22].

The updated cell state C^t is computed by combining the modified previous cell state C^{t-1}, as dictated by the forget gate, with the candidate cell state z^t, modulated by the input gate.
Finally, the new hidden state Y^t is derived from the current cell state C^t, adjusted by the output gate o^t. This new hidden state Y^t is then passed on to the next time step [Isl20].
The different states of the LSTM unit can be expressed mathematically as follows [Yad24]:

$$\begin{aligned}
z^t &= \tanh\left(W_z X^t + R_z Y^{t-1} + b_z\right)\\
i^t &= \sigma\left(W_i X^t + R_i Y^{t-1} + b_i\right)\\
f^t &= \sigma\left(W_f X^t + R_f Y^{t-1} + b_f\right)\\
c^t &= c^{t-1} \odot f^t + z^t \odot i^t\\
o^t &= \sigma\left(W_o X^t + R_o Y^{t-1} + b_o\right)\\
y^t &= \tanh\left(c^t\right) \odot o^t
\end{aligned}$$

The weights for the LSTM gates are defined as follows [Yad24]:

- Weights for Bias: $b_z,\ b_i,\ b_f,\ b_o \in R^N$
- Weights for Input: $W_z,\ W_i,\ W_f,\ W_o \in R^{N \times M}$
- Weights for Recurrent: $R_z,\ R_i,\ R_f,\ R_o \in R^{N \times M}$

Here, N represents the total number of LSTM blocks in the network, while M denotes the total number of observations in the dataset. $\odot$ represents the point-wise multiplication of two vectors [Yad24].

3.3.4 CNN-LSTM

CNN and LSTM models are increasingly being combined into hybrid models to enhance activity classification accuracy. The main architecture of the CNN-LSTM model involves the input layer, the convolutional layer, the pooling layer, the sequential LSTM layer and the fully connected layer, wherein the first three layers belong to the CNN [Ahm23]. This approach leverages the unique strengths of both models: LSTMs are effective at capturing temporal patterns in data, while CNNs excel at extracting frequency-based features [Koş23]. Although CNNs are not particularly suited for capturing temporal patterns from either short- or long-term time series, LSTMs are specifically designed to handle complex temporal features sequentially by retaining information over time. However, LSTMs are less proficient in extracting spatial features [Che19]. By integrating these capabilities, the hybrid model produces a richer and more comprehensive signal representation, leading to better performance in activity classification tasks [Koş23]. Consequently, this architecture has been widely adopted for HAR tasks, where it is essential to consider both the sequential and spatial characteristics of the data [Cup22]. However, it is important to acknowledge that the increased complexity of the hybrid model results in higher computational costs. This trade-off must be carefully managed, particularly in contexts where computational resources are limited, as is often the case with wearable devices in HAR [Koş23].

3.4 Performance Evaluation

To effectively evaluate the performance of the deep learning models, several evaluation metrics are employed. The following section briefly outlines the computation of these metrics.

The confusion matrix is a widely used tool for performance evaluation. It presents a matrix where the rows represent the true classes, and the columns the predicted classes. This matrix illustrates the distribution of instances, with each entry corresponding to the number of classified instances [Sch19]. An example is provided in Table 3.1.

Table 3.1: Confusion matrix. Each entry $C_{i,j}$, where $i, j \in \{1, \dots, K\}$, represents the number of classified instances for K classes. Rows indicate the true class, while columns indicate the predicted class. Adapted from [Sch19].

	Predicted class			
	$C_{1,1}$	$C_{1,2}$	$\dots$	$C_{1,K}$
	$C_{2,1}$	$C_{2,2}$	$\dots$	$C_{2,K}$
True class	$C_{3,1}$	$C_{3,2}$	$\dots$	$C_{3,K}$
	$\vdots$	$\vdots$	$\ddots$	$\vdots$
	$C_{K,1}$	$C_{K,2}$	$\dots$	$C_{K,K}$

The confusion matrix is particularly useful for deriving various performance metrics in a multiclass classification problem, which is applicable in our case since we aim to distinguish between multiple activities simultaneously. The performance metrics for a class k include sensitivity $Sens_k$, precision $Prec_k$, F1-score $F1_k$, and balanced accuracy $BalAcc$ [Sch19].

The calculations for these metrics are provided as follows [Sch19; Sto21]:

$$Sens_k = \frac{C_{k,k}}{\sum_{i=1}^{K} C_{i,k}}$$

where $Sens_k$ represents the ratio of true positive instances to the total actual instances of class k.

$$Prec_k = \frac{C_{k,k}}{\sum_{j=1}^{K} C_{k,j}}$$

where $Prec_k$ represents the ratio of true positives to the total predicted instances of class k.

$$F1_k = 2 \cdot \frac{Prec_k \cdot Sens_k}{Prec_k + Sens_k}$$

where $F1_k$ represents the harmonic mean of $Prec_k$ and $Sens_k$.

$$BalAcc = \frac{1}{K} \sum_{k=1}^{K} Sens_k$$

where $BalAcc$ denotes the average sensitivity across all K classes.

To account for class imbalance, a weighted average of sensitivity, precision, and F1-score is calculated as follows [Sto21]:

$$Metric_{weighted,i} = \frac{\sum_{k=1}^{K} (metric_i \cdot w_k)}{\sum_{k=1}^{K} w_k}$$

where w_k represents the weight (or support) of class k, and $metric_i$ corresponds to the sensitivity, precision, or F1-score.

Chapter 4

Methodology

4.1 Data Recordings

The dataset used for this work was recorded by Stoeve and colleagues. The study acquisition is described in detail here [Sto21]. The relevant information for this work is described in the following sections.

4.1.1 Data Acquisition

The dataset used in this study was captured by an IMU equipped with a tri-axis accelerometer (± 16 g) and a tri-axis gyroscope (± 2000°/s). Each player was fitted with two sensors, one on each foot, with the sensor embedded in the insole of the player's standard soccer shoe. The insole's rigid material and cavity design ensured the sensor's position remained stable without hindering the player's movements. The sensor and insole setup is illustrated in Figure 4.1. Data collection was conducted at a sampling rate of 200 Hz. All sessions were recorded using at least one video camera to facilitate accurate labelling.
The dataset collection process was guided by two primary objectives. First, obtaining a sufficiently large volume of data was crucial to train the deep learning model effectively. Second, real-world data from soccer matches or training sessions was essential for evaluation purposes. Due to the complexity and time involved in gathering real-world data, both controlled laboratory data and real-world data from training sessions and games were collected. This dual approach enabled the accumulation of a comprehensive dataset.
The laboratory sessions include controlled exercises, for example dribbling through a cone course

followed by a medium-distance pass, as well as semi-controlled activities such as passing the ball to a teammate, playing the ball back and then shooting at the goal. These recordings are referred to as lab sessions. In contrast, the real-world data, hereafter referred to as field sessions, were recorded without a fixed protocol, capturing soccer teams during their regular training sessions or matches. An example of a typical real-world data recording session is shown in Figure 4.1 [Sto21].

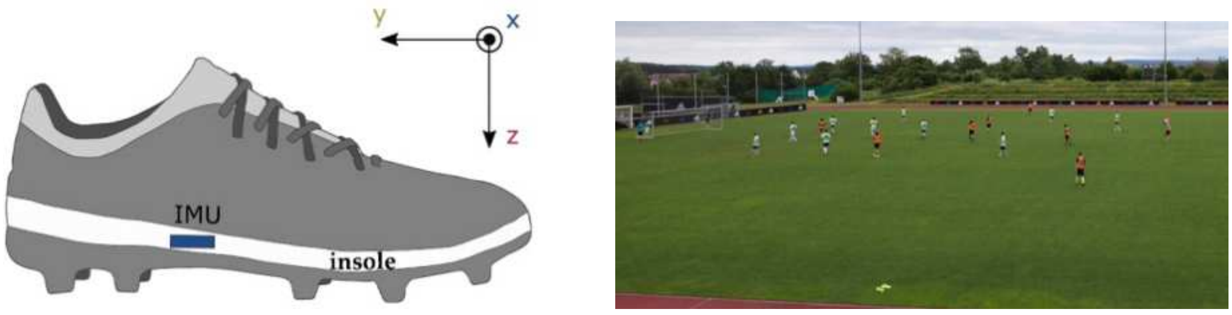


Figure 4.1: Hardware setup and example of in-field data acquisition. Sketch showing the IMU sensor placement in the sole of a soccer shoe with its coordinate system **(left)** and a snapshot from an 11-a-side game for capturing real-game data **(right)**. The video recording was used for labelling the data [Sto21].

4.1.2 Dataset

In total, data from 181 sessions were collected, comprising 38 field sessions and 143 laboratory sessions, involving 836 players (97 % male, 3 % female). The players included youth participants from the German U12 age group up to adult players. Due to hardware failures, data were recorded for only one foot in 292 instances. The dataset comprises 666.880 labelled ball contacts, including 581.659 null contacts, 49.956 dribbling contacts, 18.394 short passes medial, 4.900 short passes other, 2962 long passes, 8.904 shots, and 105 ambiguous shots. The average duration of lab sessions was 35 minutes (SD = 17), while field sessions averaged 73 minutes (SD = 38). Lab sessions typically involved an average of four players (SD = 3), whereas field sessions had an average of eight players (SD = 4). The largest recorded session involved 17 players [Sto21].
The same data split as Stoeve and colleagues was used [Sto21]. However, the data used to optimize peak detection, which is explained in more detail in Chapter 4.3, was excluded from the training and test set. Consequently, our training dataset consists of 153 sessions with 687 players, while our test dataset contains 19 sessions with 129 players. The class distribution in the test set was similar to the training set. The test set includes 10 laboratory sessions and nine field sessions to provide a comprehensive performance evaluation of the model under both controlled and real-world conditions. The overall data split can be seen in Figure 4.2.

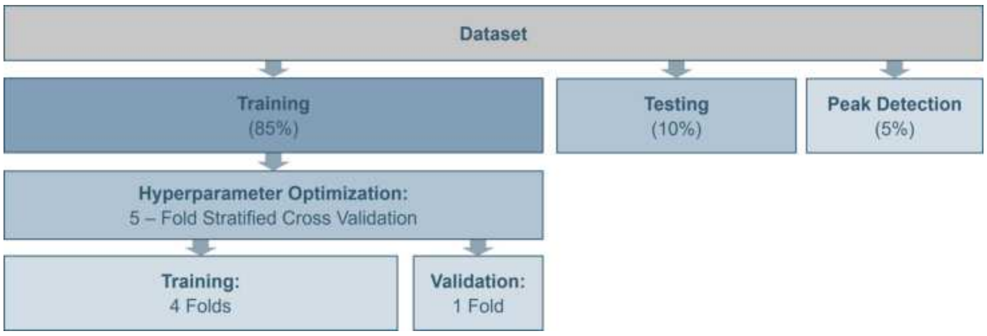


Figure 4.2: The overall data split. 85 % of the sessions are used for training, 10 % for testing and 5 % for peak detection optimization.

4.1.3 Labelling

The ball contacts were annotated by trained experts [Sto21]. The labelling distinguished between seven different classes, with further differentiation based on the use of the left or right foot. These seven classes were organized into four main categories: the null class, dribbling, passing, and shooting. Within the passing and shooting categories, additional subcategories were identified to further specify the type of action. An overview if the ball contact types with their corresponding classes and descriptions is given in Table 4.1. In the following, the classes light contact, short pass medial, short pass other, long pass, shot and ambiguous shot are defined as events.

Table 4.1: Overview of ball contact types with corresponding classes and descriptions. Adapted from [Sto21].

Ball Contact Type	**Description**	**Class**
none	no contact, e.g. player is running or sprinting	null
strong contact	strong contact, e.g. when the ball bounces off the foot	null
ambiguous contact	strong contact, e.g. during a duel	null
ambiguous short pass	unconventional short pass, e.g. using the heel	null
ambiguous range pass	unconventional range pass, e.g. applying spin to the ball	null
juggling kick	player is juggling, e.g. holds the ball in the air	null
light contact	small contact while player is not moving or during dribbling	dribbling
short pass medial	short pass with the medial part of the foot	short pass medial
short pass other	pass with another part of the foot than medial	short pass other
long pass	pass over longer distance	long pass
shot	kick directed towards the goal	shot
ambiguous shot	non-standard kick directed towards the goal, e.g. bicycle kick	ambiguous shot
left/right	foot used to make contact with the ball	foot

4.2 Preprocessing

Stoeve and colleagues performed a synchronization routine at the beginning of each session to synchronize the IMU data of the two sensors. The sensors were attached to a rod and smashed against the floor three times. This action produced three distinct peaks across all axes in the acceleration data. A CNN was trained to detect these signature peak patterns. If the CNN could not detect the peaks, the moment when the rod first touched the ground was manually determined from video recordings. The timeline was then interpolated, estimating the sampling frequency based on the number of samples between the detected peaks and the timing observed in the video. Subsequently, synchronization was manually verified for all recordings by checking labelled shot events. Finally, all IMU signals were normalized between -1 and 1 [Sto21].

4.3 Segmentation

In order to obtain comprehensive but still compact input data for our DL network, the peak detection method developed by Schuldhaus was applied [Sch19], which was also used by Stoeve and colleagues [Sto21]. This method identifies ball contacts by detecting peaks in smoothed gyroscope data from the IMU, as these peaks are key indicators for relevant movement events. A second-order high-pass filter was used for smoothing, with a window length of 400 samples (equivalent to 2 seconds). Stoeve and colleagues previously evaluated different window lengths (1s, 1.5s, and 2s) and concluded that a 2-second window was optimal. The windows were centred around the detected peaks. If peaks were closely spaced, only the peak with the highest magnitude was considered, while the others were neglected [Sto21].

The peak detection method was applied as follows: when a peak was detected, indicating a potential ball contact, the data around that peak was further extracted and analyzed.
If the data was classified as one of our target classes, it was saved with the appropriate label and the window was moved to a new position according to:

$$pos_{new} = pos_{peakdetected} + length_{window} \times overlap$$

where $pos_{peakdetected}$ represents the position of the detected peak, $length_{window}$ is the length of the window in samples, and $overlap$ is the fraction of the window that overlaps with the previous one. If the data did not belong to one of the target classes, it was categorized as a null class and the window was moved forward by:

$$pos_{new} = pos_{current} + length_{window} \times overlap$$

where $pos_{current}$ is the current position of the window.

Like Stoeve and colleagues, an overlap of 75 % was chosen [Sto21]. With a window size of 400 samples, this corresponds to an overlap of each window by 100 samples, as each window is shifted by 75 %. This overlap ensures that any additional peaks located near the current window boundary are not missed. This is particularly relevant in scenarios such as dribbling, where the optimal minimum distance for successful peak detection has been determined to be 100 samples. Since the peaks are always centred within the window, additional peaks may still emerge within the subsequent 200 samples of the same window. However, the 100-sample overlap ensures that if another peak occurs within the current window, at least 100 samples away from the previous one, it will be captured due to the overlap, preventing any peaks from being overlooked. This ensures that target events are not missed, thus increasing the overall sensitivity of event detection.

Unlike Stoeve and colleagues, who focused on detecting shots and passes characterized by relatively large amplitude peaks, this thesis aims to identify more frequent spaced ball contacts, such as those occurring during dribbling [Sto21]. To avoid overfitting the method to the entire dataset and to promote better generalization, the peak detection process was optimized using data from only 33 players from nine different sessions. Of these sessions, eight were conducted under laboratory conditions, while one was a field session. This data was excluded from further analysis. Specifically, three hyperparameters were adjusted: f_{cut}, *tresh_abs* and *min_dist*. f_{cut} denotes the filter's cutoff frequency, *tresh_abs* the signal amplitude within the peak detection, and *min_dist* defines the minimum distance between two consecutive events from the target class in samples. The second-order high-pass filter used in peak detection was retained.

To optimize the cutoff frequency f_{cut}, the implementation by Schuldhaus was followed. Given that peaks are likely to occur in higher frequency bands, the magnitude of the Discrete Fourier Transformation (DFT) coefficients for the accelerometer axes was calculated, considering only events from the target class. The energy was computed by summing the squared magnitudes of the coefficients of the DFT. The determined f_{cut} was set to the frequency at which 90 % of the energy level was reached. The final cutoff frequency was determined as the median of the individual cutoff frequencies from the considered instances [Sch19].

To optimize *tresh_abs* and *min_dist*, a grid search was conducted. The grid search parameters were predefined, as listed below. The goal was to maximize sensitivity, ensuring that all ball contacts from the target classes were detected without including too many null-class contacts. The search space for the grid search of parameters $tresh\_abs$ and min_dist was:

$$\textit{tresh\_abs} \in \{0.1 \cdot k\}_{k\in\{1,2,3,4\}}$$
$$\textit{min\_dist} \in \{50 \cdot k\}_{k\in\{1,2,3,4,5,6\}}$$

To address the imbalance in the dataset, rebalancing was performed at the end of the preprocessing pipeline using a combination of random undersampling and Adaptive Synthetic Sampling Approach (ADASYN). ADASYN is designed to address challenges associated with learning from imbalanced datasets by focusing on the minority class. It creates synthetic data for minority class samples, but instead of treating all samples equally, it generates more synthetic instances for those that are harder to classify. This approach enhances the learning process by reducing the bias caused by the class imbalance and adjusting the classification decision boundary to better handle the more difficult examples in the dataset. As a result, ADASYN helps improve the model's ability to learn from imbalanced data [He08].
Our rebalancing works as follows: first the mean frequency of all instances within the target classes was calculated, then generate synthetic instances for the underrepresented target classes to this mean value and subsequently undersample the overrepresented classes, including the null class. This approach helps mitigate class imbalance caused by peak detection as well as the naturally higher frequency of certain soccer events, such as the more frequent contacts during dribbling compared to shooting, for instance.
Additionally, the sensor placement on the left or right foot was determined based on the sensor ID tagged during data collection. Figures 4.3 and 4.4 display the acceleration signals on the left and the gyroscope signals on the right. Each row corresponds to a different action, all executed with the left foot. Figure 4.3 includes examples from the null class, dribbling, short pass medial, and short pass other, while Figure 4.4 presents examples from the long pass, shot class, and ambiguous shot class. The only exception is the long pass in Figure 4.4, which includes actions performed with both the left and right foot. Notably, the X component of the gyroscope data and the Y component of the accelerometer data are mirrored.

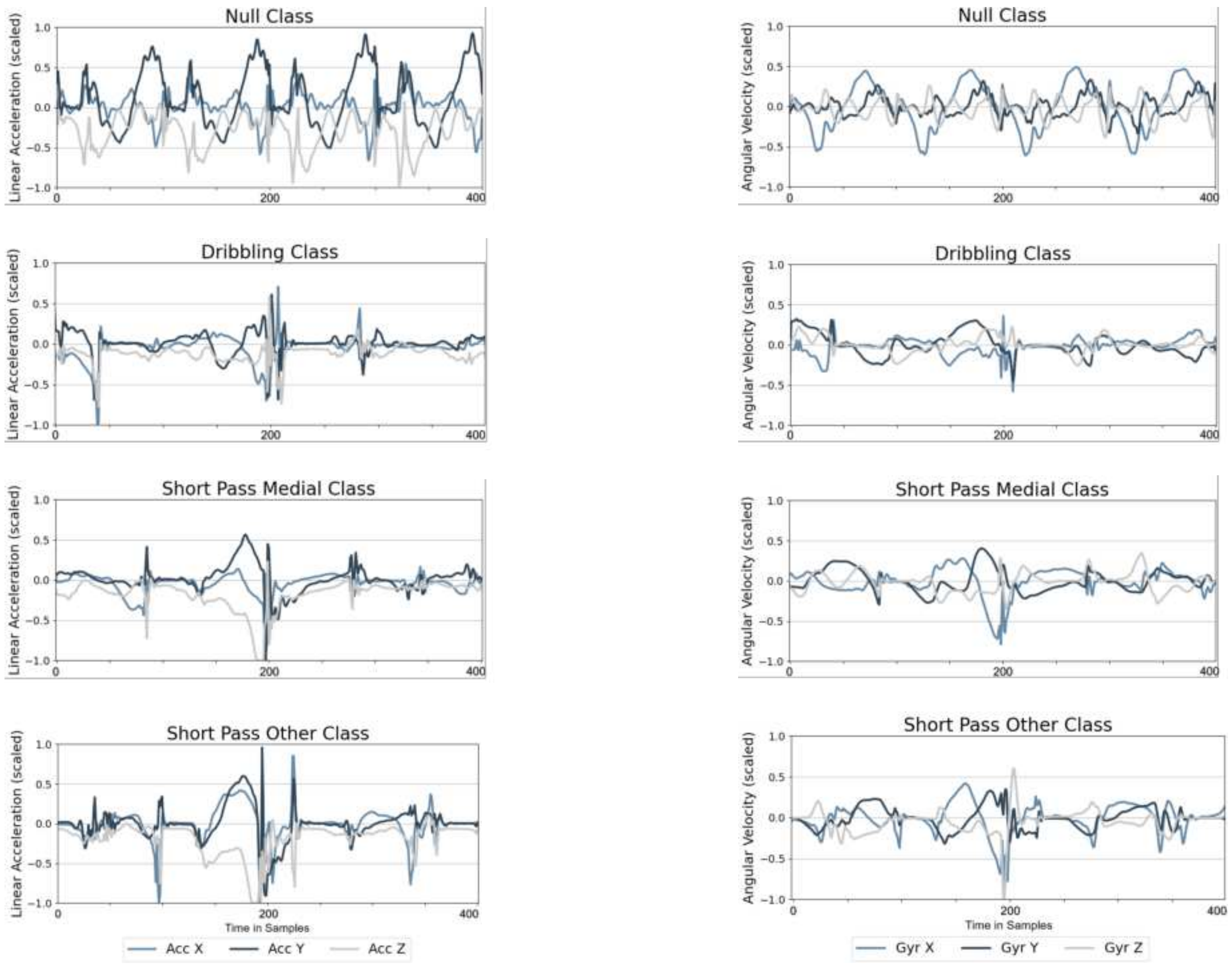


Figure 4.3: Examples of the null class and target classes: Dribbling, Short Pass Medial, and Short Pass Other. The accelerometer signals are shown on the left, while the gyroscope signals are on the right. Each window contains 400 samples and is normalized to a range between -1 and 1.

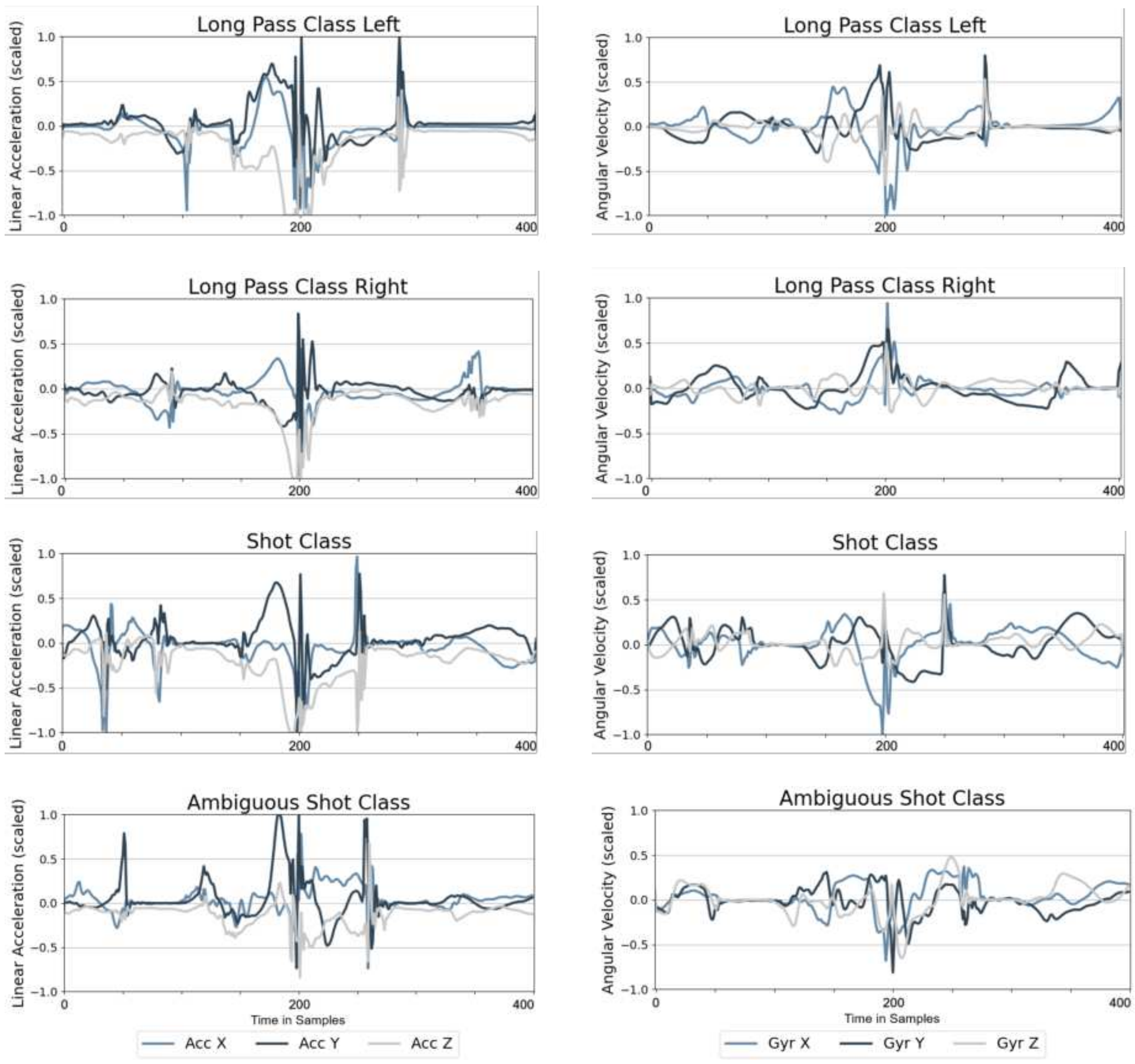


Figure 4.4: Examples of the target classes: Long Pass, Shot, and Ambiguous Shot. The accelerometer signals are shown on the left, while the gyroscope signals are on the right. Each window contains 400 samples and is normalized to a range between -1 and 1.

4.4 Models and Hyperparameter Optimization

Given the similarity between our task and that of Cuperman and colleagues, who aimed to classify soccer-specific activities, their basic framework was adapted to meet the specific requirements of this project. Cuperman and colleagues achieved a classification accuracy of up to 97.53 % using a combination of CNN and CNN-LSTM models, which showed strong results in activity classification [Cup22]. Following their approach, both a CNN and a CNN-LSTM model were implemented for the classification, adapting their model structure and optimizing them for our data and objectives. Therefore, in the following sections, the CNN and CNN-LSTM models will be optimized and analyzed separately, allowing for a comprehensive comparison of both models at the end.
Six input streams were used: three from the accelerometer and three from the gyroscope. In this way, both linear and rotational movements can be captured, helping in distinguishing between different events, as each movement has its own characteristics.

The basic structure of our models for the CNN and CNN-LSTM is shown in Figure 4.5.
Both models were optimized with the same approach using the Optuna framework with Bayesian optimization [Aki19]. The optimization process involved 100 trials. Before each trial, the events were shuffled to ensure that the model was exposed to a different distribution of the data, avoiding possible biases due to the order of the data. The maximum number of epochs was set to 100, with a fixed batch size of 64. The ADAM optimizer was used for both models due to his adaptability and efficiency in training complex models [Kin14].
A five-fold stratified cross-validation was used, ensuring each fold maintained a balanced distribution of instances to address the class imbalance in the dataset. In each of the five iterations, only the training folds were balanced using our class balancing method, while the validation fold retained the original class distribution for a realistic evaluation, demonstrating how the model would respond to real-world conditions. During training, the weighted F1-score was continuously monitored after each batch to evaluate the model performance. An early stopping was implemented, which terminated the training process if the weighted F1-score did not improve in 20 consecutive epochs. The learning rate was dynamically adjusted using a learning rate scheduler that reduced the learning rate by a factor of 0.5 if no improvement in performance was observed over 10 epochs. The minimum learning rate was set to 0.00001 to prevent the learning rate from becoming too small and stalling the training progress. After cross-validation, the mean of the weighted F1-scores from all folds was determined to calculate the final F1-score for each trial.

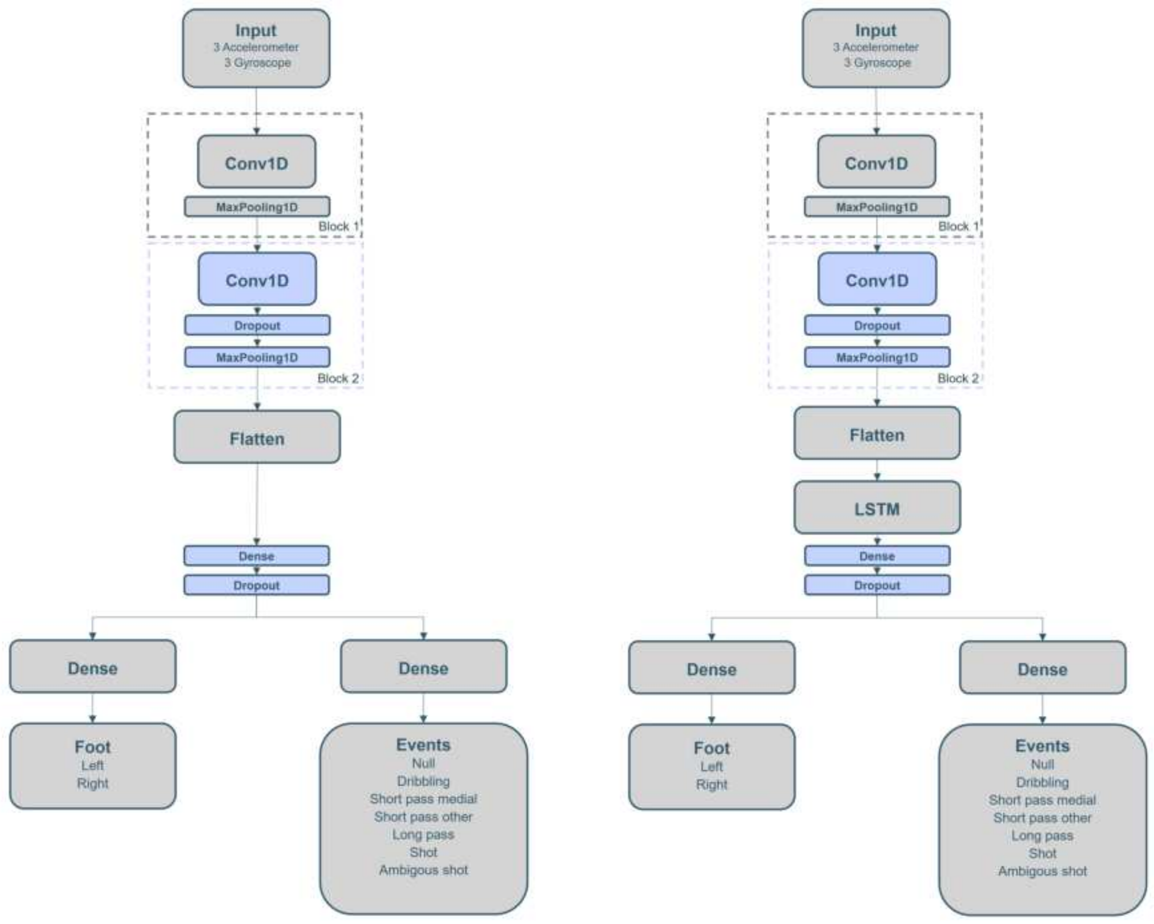


Figure 4.5: The basic structure of our CNN model **(left)** and CNN-LSTM model **(right)** is shown. The grey blocks represent the fixed core structure of the models, while the blue blocks indicate optional components that were included or adjusted during hyperparameter optimization.

The main hyperparameters that were optimized are described below:

- filters: The number of filters in the Conv1D layers.
- filter_LSTM: The number of filters in the LSTM layer.
- kernel_size: The size of the convolutional kernel in the Conv1D layers.
- learning_rate: The learning rate for the optimizer.
- num_layers: The number of times the combination of Conv1D, Dropout, and MaxPooling1D layers is repeated.
- use_dense_dropout: Adds an optional Dense layer with 64 units and a Dropout layer (rate 0.5) after the Flatten layer.

Chapter 5

Results

5.1 Peak Detection Optimization

When optimizing the cut-off frequency f_{cut} for the segmentation, an optimal value of 22.7 Hz was determined. Since a high-pass filter for our gyroscope data was used, all frequencies below 22.7 Hz are filtered out. In this way, low-frequency noise is removed while the higher-frequency signals remain relevant for the segmentation. Additionally, the hyperparameters *tresh_abs* and *min_dist* were optimized using a grid search. The grid search results indicated that the optimal values were *tresh_abs = 0.1* and *min_dist = 100* samples, achieving a sensitivity of 91 %. This means that with these parameter settings, the peak detection algorithm correctly detected 91 % of the originally labelled target events.

5.2 Hyperparameter Optimization

Table 5.1 gives an overview of the search spaces and the final results for the hyperparameters.

Table 5.1: Overview of the CNN and CNN-LSTM models and the respective results. Each hyperparameter was optimized individually within the corresponding search space.

Hyperparameter	**Sampling**	**Search Space**	**Result CNN**	**Result CNN-LSTM**
filters	integer	16 - 128	112	114
filter_LSTM	categorical	[32,64,128]	-	64
kernel_size	categorical	[3, 5]	5	5
learning_rate	log-uniform	1×10^{-5} - 1×10^{-2}	1.52×10^{-3}	0.31×10^{-3}
num_layers	integer	0 - 3	3	3
dense_dropout	categorical	[True, False]	False	False

5.3 Models

5.3.1 CNN

The CNN achieved a weighted F1-score of 88.26 % in the best trial during hyperparameter tuning with stratified five-fold cross-validation. The hyperparameters from this trial were saved and the final model was configured accordingly. The model comprises four convolutional blocks, each followed by a pooling layer. All convolutional layers are activated using the ReLU activation function. While the first block only includes a convolutional and pooling layer, the subsequent three blocks also contain a dropout layer to reduce overfitting. The pooling layers progressively reduce the input dimensionality. After flattening the features, the model outputs predictions through two dense layers: one for event prediction using softmax activation, and another for binary foot classification with sigmoid activation. In total, there were 210,792 trainable parameters.
The final architecture of the CNN model is shown in Table 5.2 and is further illustrated in Figure 5.1 for better comprehension.

Table 5.2: Summary of the CNN model architecture, showing the layer types, parameters, output shapes, and the total number of trainable parameters in each layer.

Layer Type	Parameter	Output Shape	# of Parameters
InputLayer		(400, 6)	0
Conv1D	filter: 112, kernel size: 5	(396, 112)	3,472
MaxPooling1D	size: 2, strides: 2	(198, 112)	0
Conv1D	filter: 112, kernel size: 5	(194, 112)	62,832
Dropout	rate: 0.5	(194,112)	0
MaxPooling1D	size: 2, strides: 2	(97, 112)	0
Conv1D	filter: 112, kernel size: 5	(93, 112)	62,832
Dropout	rate: 0.5	(93, 112)	0
MaxPooling1D	size: 2, strides: 2	(46, 112)	0
Conv1D	filter: 112, kernel size: 5	(42, 112)	62,832
Dropout	rate: 0.5	(42, 112)	0
MaxPooling1D	size: 2, strides: 2	(21, 112)	0
Flatten		(2352)	0
Dense - Events Output	activation: softmax	(7)	16,471
Dense - Foot Output	activation: sigmoid	(1)	2,353

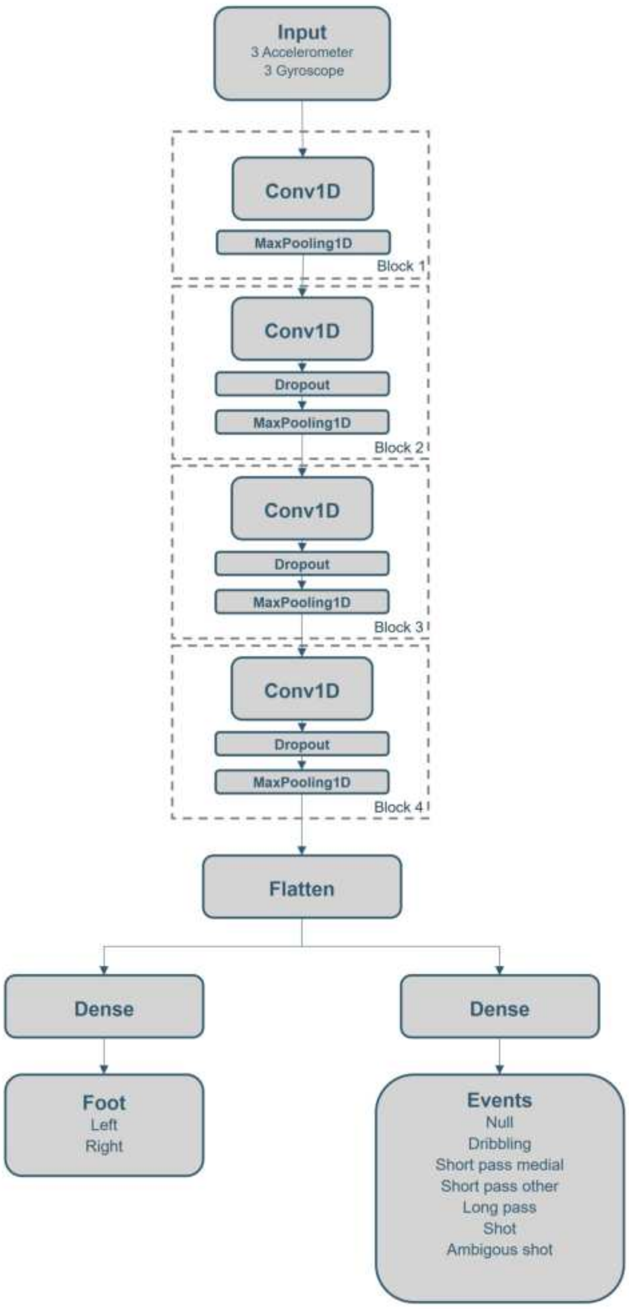


Figure 5.1: Summary of the CNN model architecture. The model consists of four blocks of Conv1D, MaxPooling1D, and Dropout layers, followed by a Flatten layer. The output branches into two separate Dense layers, classifying the foot used (left/right) and the type of event (seven possible classes).

Overall Scores

The model was then trained again with the fully rebalanced training data and evaluated using the unknown test data. The same training settings set during the hyperparameter optimization were used for this training. Figure 5.2 shows the trend in accuracy for the different events and the foot over the epochs. In addition, the total loss over the epochs for the entire model is shown on the right-hand side of the Figure.

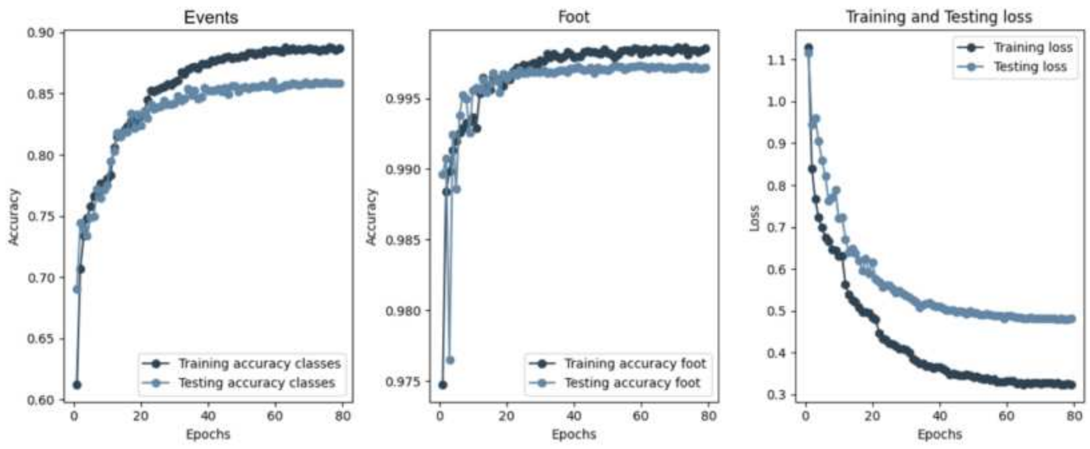


Figure 5.2: In the first and second plots, the training and testing accuracy for the events and the foot are shown over the epochs. The third plot illustrates the training and testing loss for the entire model throughout the epochs.

As presented in Table 5.3, the model demonstrates good overall performance when evaluated on both laboratory and real game data, achieving a weighted F1 score of 92.54 % for event differentiation and 97.83 % for foot classification. The high weighted sensitivity values, 94.47 % for the events and 97.84 % for foot classification, indicate that the model correctly identifies most positive cases. However, notable results are found in the balanced accuracy scores. In contrast to the foot classification, where the balanced accuracy is high at 97.87 %, the balanced accuracy for the general events is only 40.41 %. Overall, the model demonstrates better performance in foot classification compared to its ability to distinguish between the seven different classes.
It is also worth noting that event differentiation performs better with the real game data, achieving a weighted F1 score of 91.43 %, compared to 86.15 % with the laboratory data. Conversely, foot classification is more accurate with the laboratory data, reaching a weighted F1 score of 99.20 %, while only 97.22 % was achieved with the real game data.

Table 5.3: Performance metrics for event and foot predictions with the corresponding scores.

	Metric	Laboratory Test Set - Score	Real Game Set - Score	Overall - Score
Events	Accuracy	83.61 %	88.82 %	91.21 %
	Weighted Sensitivity	90.45 %	95.11 %	94.47 %
	Weighted Precision	83.61 %	88.82 %	91.21 %
	Weighted F1	86.15 %	91.43 %	92.54 %
	Balanced Accuracy	53.71 %	32.31 %	40.41 %
Foot	Accuracy	99.20 %	97.22 %	97.84 %
	Weighted Sensitivity	99.20 %	97.22 %	97.84 %
	Weighted Precision	99.20 %	97.22 %	97.84 %
	Weighted F1	99.20 %	97.22 %	97.83 %
	Balanced Accuracy	99.20 %	97.20 %	97.87 %
Events & Foot	Total Loss	0.5798	0.5375	0.4485

Detailed Score Composition

For a more detailed analysis, the confusion matrices can be analyzed, offering insights into how well each class is performing. The confusion matrices for the CNN, which show the different events and the distinction between the feet, have been normalized between 0 and 1 to allow for clearer comparisons between classes and are presented in Figure 5.3 and 5.4.
Overall, the results indicate that the null class (unknown), light contact during dribbling, short pass medial, and shots exhibit the highest accuracy. However, in 19 % of the cases, light contacts are misclassified as the null class (unknown). Short pass medial is most frequently confused with short pass other (11 % of the cases), but it still maintains a solid accuracy of 68 %. In contrast, short pass other is correctly classified in only 45 % of cases. This class is predominantly confused with short pass medial (27 %), but is also often misclassified as the null class (unknown) (12 %) and light contact (10 %). Among the passing classes, long pass performs the worst, being correctly classified in only 21 % of cases. The most common misclassification for long pass is with shots (36 %), followed by short pass medial (24 %) and short pass other (12 %). In contrast, shots are correctly identified 70 % of the time, though they are occasionally confused with short pass medial (15 %). The ambiguous shot class, however, performs the poorest, with no instances being correctly classified. Instances from this class are most commonly misclassified across three categories: shots, short pass other, and the null class (unknown), each with an accuracy of 27 %.

The model distinguishes between the left and right foot very well. The accuracy is 97 % for the left foot and 99 % for the right foot, demonstrating very good performance in foot differentiation.

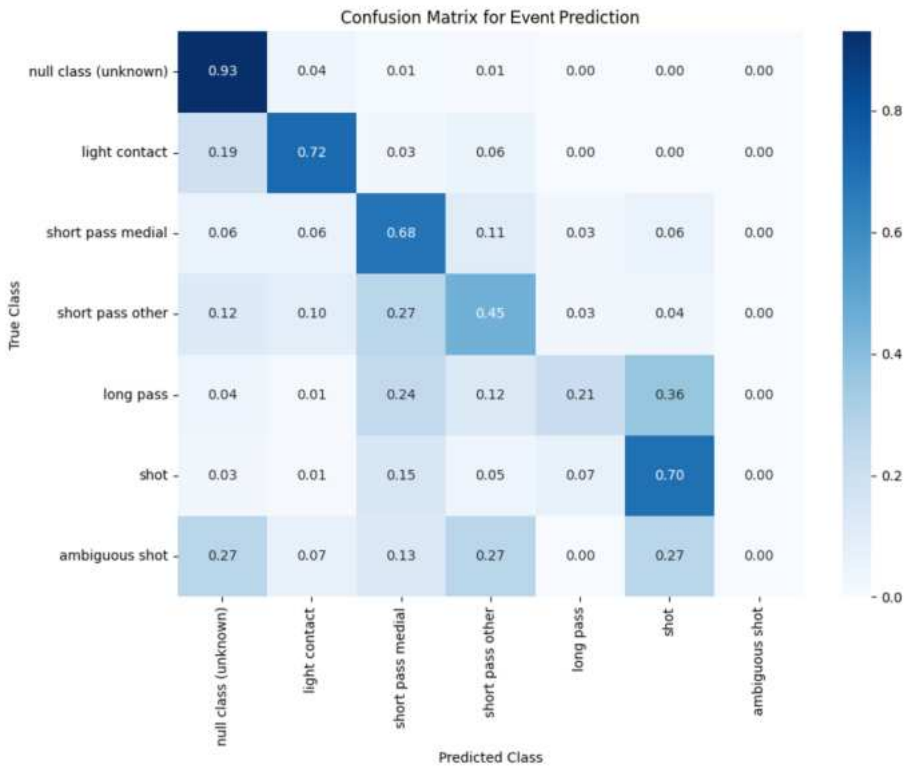


Figure 5.3: Confusion matrix for event predictions. The darker the colour, the more often the predicted class matches the true class. The matrix shows the classification performance, with the values along the diagonal representing correct predictions and the elements outside the diagonal representing wrong classifications. The values were normalized between 0 and 1, where 0 indicates no occurrences and 1 represents perfect classification for that class.

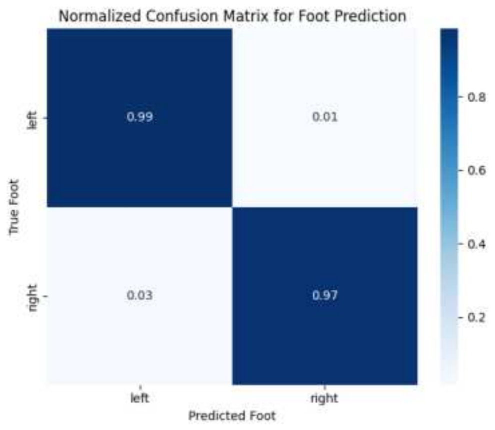


Figure 5.4: Confusion matrix for foot predictions. The values were normalized between 0 and 1, where 0 indicates no occurrences and 1 represents perfect classification for that class.

As previously discussed in Chapter 4.1.2, the test dataset comprises a combination of laboratory settings data and real game data. The following sections provide a detailed overview of the model's performance when evaluated separately on each test set for the CNN.

Laboratory Test Set

The confusion matrix is presented below in Figure 5.5 and 5.6 for the differentiation between the events and the two feet, based exclusively on the laboratory test data set.

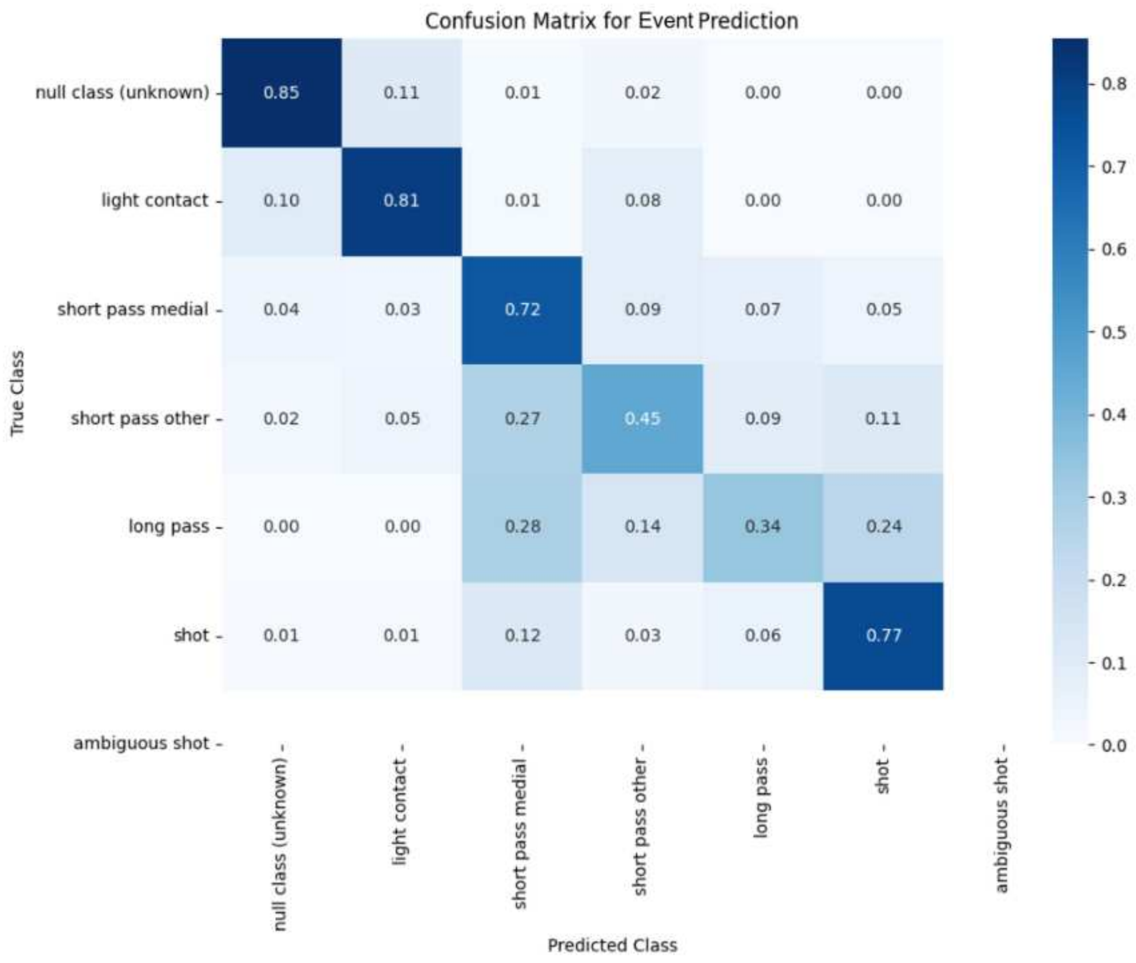


Figure 5.5: Confusion matrix for event predictions based on the laboratory test set. The values are normalized between 0 and 1.

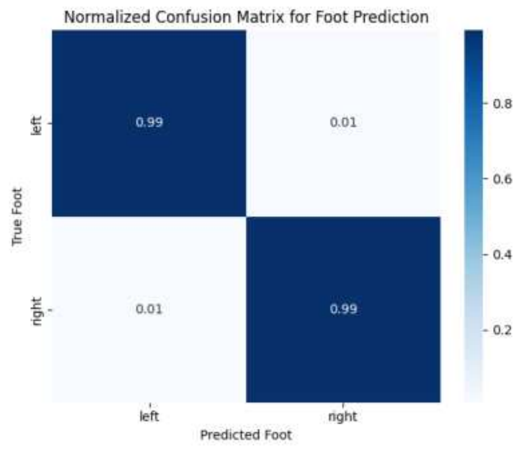


Figure 5.6: Confusion matrix for foot predictions based on the laboratory test set. The values are normalized between 0 and 1.

As with the overall performance shown in Figure 5.3, the null class (unknown) achieves the highest accuracy at 85 %. However, similar to the combined test set, it is most often confused with light contacts (11 %). Notably, light contacts are more accurately distinguishable from other classes, achieving an accuracy of 81 %, although they are frequently misclassified as the null class (unknown) (10 %). The short pass medial class also demonstrates relatively strong performance, with an accuracy of 72 %. Meanwhile, the short pass other class maintains the same accuracy as observed in the overall test set at 45 %, with the most frequent misclassification occurring with short pass other (27 %). Among the events, long pass performs the least effectively, with a correct classification rate of only 34 %. It is most often misidentified as either a short pass medial (28 %) or a shot (24 %). The shots class achieves a robust accuracy of 77 %. Notably, the ambiguous shot class is absent from the laboratory dataset.
The model distinguishes between the feet very accurately on the laboratory dataset, achieving correct classification in 99 % of cases for both left and right.

Real Game Test Set

In the real game test dataset, the null class (unknown) is well distinguished from the other classes with high accuracy of 91 %. Light contacts, however, are recognized with lower accuracy compared to the laboratory sessions, achieving only 70 %. The short pass medial class shows similar behaviour to the lab sessions, being most frequently confused with short pass other (16 %), yet correctly classified in 65 % of cases. Conversely, short pass other performs better under real-world scenarios, with an accuracy of 52 %, though it is still often misclassified as short pass medial (22 %). Long pass continues to perform among the worst, with only 18 % correct classification and nearly half of the cases (45 %) misclassified as a shot. Shots perform worse under real conditions, achieving only 59 % accuracy compared to 77 % in the laboratory test set.
As previously mentioned, ambiguous shots appear only in real scenarios; however, they are never correctly classified and are most frequently misidentified as short pass other (53 %).
The model continues to distinguish well between the two feet, though it performs slightly worse than under laboratory settings, with 98 % correct classifications for the left foot and 97 % for the right foot.

The results can be seen in Figures 5.7 and 5.8 for the events and the foot differentiation, based exclusively on the real game test data set.

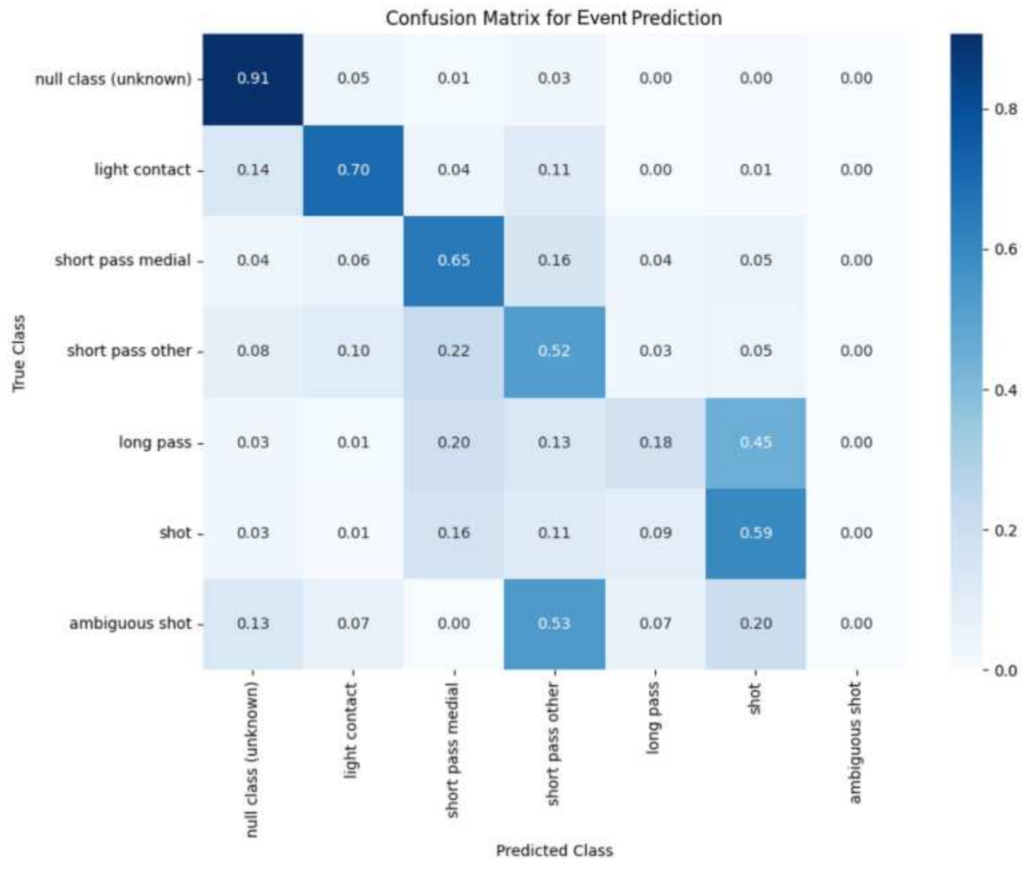


Figure 5.7: Confusion matrix for event predictions based on the real game test set. The values are normalized between 0 and 1.

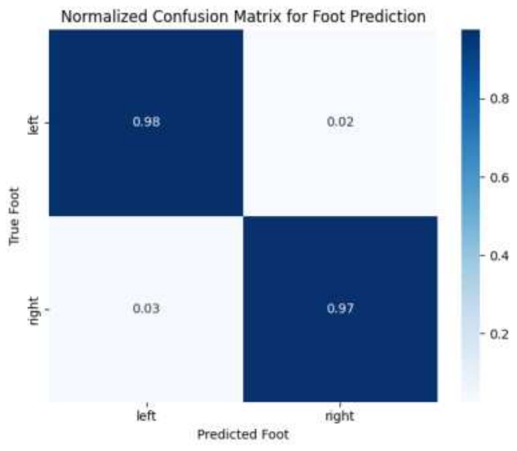


Figure 5.8: Confusion matrix for foot predictions based on the real game test set. The values are normalized between 0 and 1.

Null-Pass-Shot Classification

To gain a broader view of the classification results based on the entire test dataset, the events were additionally grouped into three main categories: null, passes, and shots. The null class includes "null class (unknown)" as well as "light contacts." Passes comprise "short pass medial," "short pass other," and "long pass," while the shots category consists of "shot" and "ambiguous shot." The results are displayed in figure 5.9.

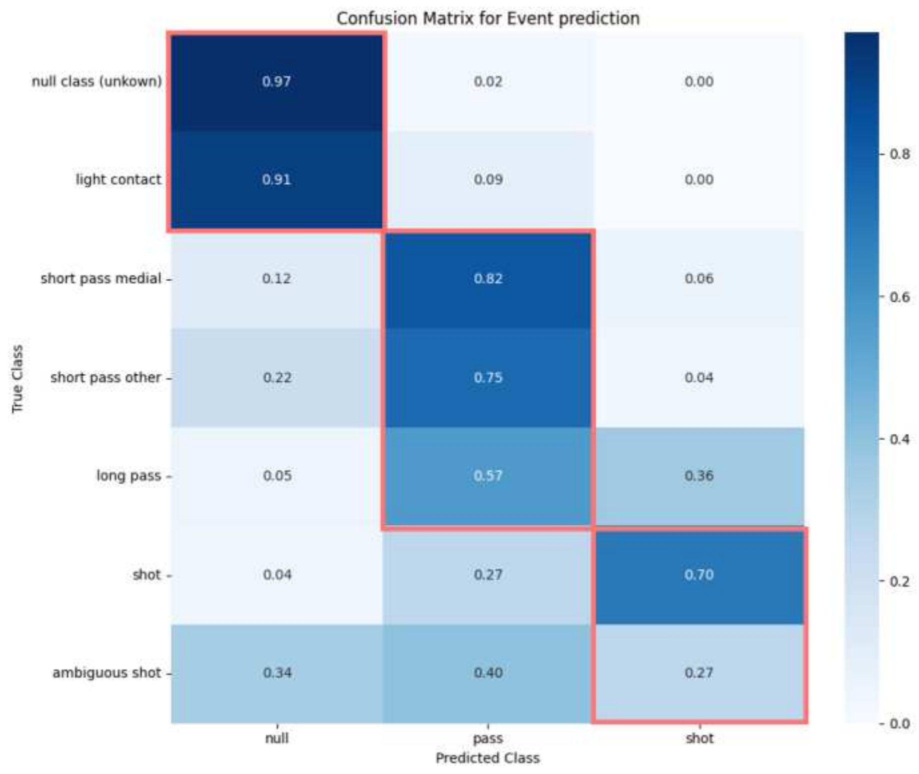


Figure 5.9: Confusion matrix for the grouped classes: null, pass and shot

The null class is recognized fairly accurately, with 97 % accuracy for null class (unknown) and 91 % for light contacts. However, 34 % of ambiguous shots are also misclassified as part of the null class. Passes are relatively well separated from the null class. However, passes are misclassified as shots in 27 % of cases and as ambiguous shots in 40 % of cases. Shots are correctly classified with 70 % accuracy, while ambiguous shots are correctly identified only 27 % of the time. Shots are most often misclassified as long passes (36 %).

5.3.2 CNN-LSTM

The CNN-LSTM achieved a weighted F1-score of 87.35 % in the best trial during hyperparameter tuning with stratified five-fold cross-validation. The hyperparameters from this trial were saved, and the final model was configured accordingly.
Like the standard CNN model 5.3.1, this architecture features four convolutional blocks, each followed by a pooling layer. However, the key difference is that the final block is followed by an LSTM layer before the final classification. All convolutional layers use the ReLU activation function, whereas the LSTM layer uses the tanh activation function. While the first block contains only a convolutional and pooling layer, the subsequent three blocks also include dropout layers to reduce overfitting. The pooling layers progressively downsample the input dimensions. After the features are flattened, they are passed through an LSTM layer that captures the temporal dependencies in the data. As with the previous CNN model, the output consists of two dense layers: one with softmax activation for event prediction and another with sigmoid activation for binary foot classification. The model contains a total of 828,840 trainable parameters.
The final architecture of the CNN-LSTM model is shown in Table 5.4 and is further illustrated in Figure 5.10 for better comprehension.

Table 5.4: Summary of the CNN-LSTM model architecture, showing the layer types, parameters, output shapes, and the total number of trainable parameters in each layer.

Layer Type	**Parameter**	**Output Shape**	**# of Parameters**
InputLayer		(400, 6)	0
Conv1D	filter: 114, kernel size: 5	(396, 114)	3,534
MaxPooling1D	size: 2, strides: 2	(198, 114)	0
Conv1D	filter: 114, kernel size: 5	(194, 114)	65,094
Dropout	rate: 0.5	(194,114)	0
MaxPooling1D	size: 2, strides: 2	(97, 114)	0
Conv1D	filter: 114, kernel size: 5	(93, 114)	65,094
Dropout	rate: 0.5	(93, 114)	0
MaxPooling1D	size: 2, strides: 2	(46, 114)	0
Conv1D	filter: 114, kernel size: 5	(42, 114)	65,094
Dropout	rate: 0.5	(42, 114)	0
MaxPooling1D	size: 2, strides: 2	(21, 114)	0
Flatten		(2394)	0
LSTM	filter: 64	(64)	629,504
Dense - Events Output	activation: softmax	(7)	455
Dense - Foot Output	activation: sigmoid	(1)	65

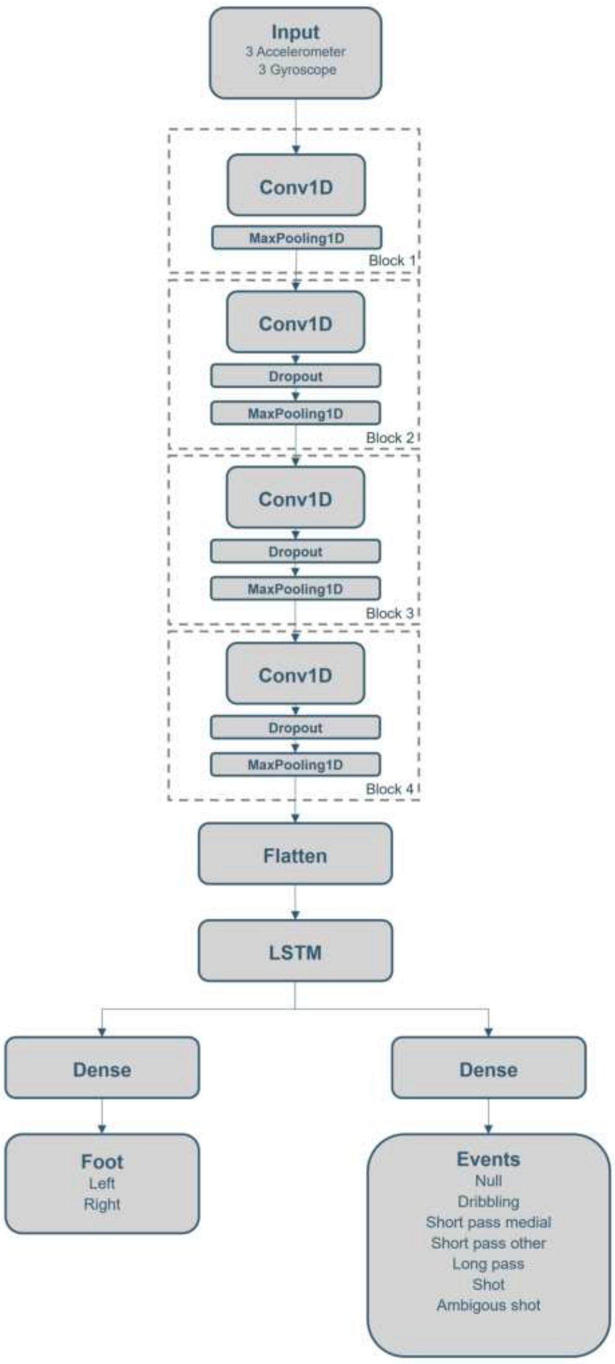


Figure 5.10: Summary of the CNN-LSTM model architecture. The model consists of four blocks of Conv1D, MaxPooling1D, and Dropout layers, followed by a Flatten and LSTM layer. The output branches into two separate Dense layers, classifying the foot used (left/right) and the type of event (seven possible classes).

Overall Scores

The same approach as with the CNN model was used. The model was trained again with the fully rebalanced training data and evaluated on the unseen test data. The training settings from the hyperparameter optimization for the CNN-LSTM model were also applied here. Figure 5.11 shows the trend in accuracy for the different events and the foot classification over the epochs. Additionally, the total loss over the epochs is depicted on the right-hand side of the Figure.

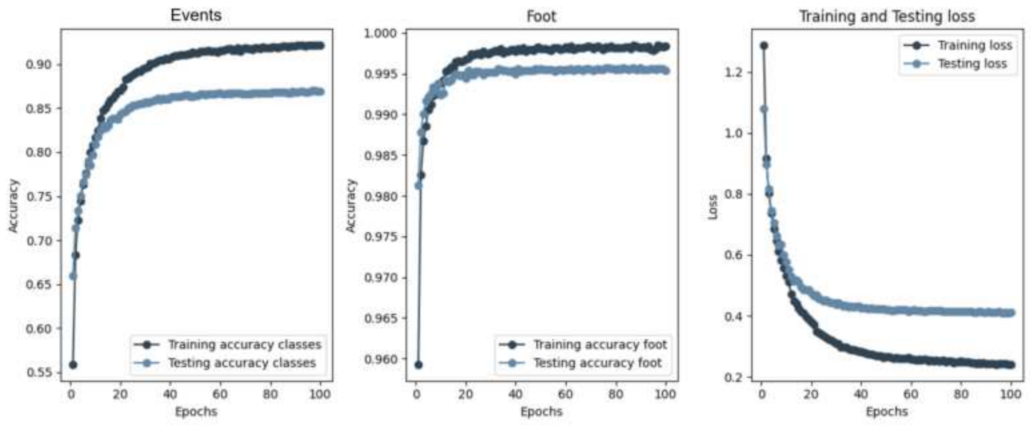


Figure 5.11: In the first and second plots, the training and testing accuracy for the events and the foot are shown over the epochs. The third plot illustrates the training and testing loss for the entire model throughout the epochs.

As illustrated in Table 5.5, the model performs generally well, achieving a weighted F1 score of 90.08 % for distinguishing between different events and a weighted F1 score of 96.27 % for foot classification. The weighted sensitivity values are comparable to those of the CNN, with 94.01 % for the events and 96.28 % for the feet. However, the weighted precision for the events is lower, with the model reaching 87.66 %, whereas the CNN achieved a higher score of 91.21 %. In contrast, the model achieves a similar weighted precision of 96.28 % for foot classification, comparable to the CNN. As already noted with the CNN, the balanced accuracy for the events is for the CNN-LSTM also particularly poor, with a score of 36.67 %. The CNN also only reached a score of 40.41 %. In contrast, the balanced accuracy for the foot classification remains strong at 96.18 %.

Additionally, the event prediction performs better on the real game dataset, achieving a weighted F1 score of 90.87 % compared to 85.17 % on the laboratory data. In contrast, foot classification works more accurately under controlled settings, with a weighted F1 score of 99.17 % on the laboratory data, while reaching only 95.17 % under real-world conditions.

Table 5.5: Performance metrics for event and foot predictions with the corresponding scores.

	Metric	**Laboratory Test Set - Score**	**Real Game Set - Score**	**Overall - Score**
Events	Accuracy	82.90 %	88.42 %	87.67 %
	Weighted Sensitivity	89.76 %	94.70 %	94.01 %
	Weighted Precision	82.90 %	88.42 %	87.66 %
	Weighted F1	85.17 %	90.87 %	90.08 %
	Balanced Accuracy	52.71 %	31.42 %	36.67 %
Foot	Accuracy	99.17 %	95.20 %	96.27 %
	Weighted Sensitivity	99.17 %	95.17 %	96.28 %
	Weighted Precision	99.17 %	95.17 %	96.27 %
	Weighted F1	99.17 %	95.17 %	96.27 %
	Balanced Accuracy	99.19 %	95.04 %	96.18 %
Events & Foot	Total Loss	0.5470	0.5063	0.5222

Detailed Score Composition

Overall, it can be observed that the CNN-LSTM model does not perform quite as well as the CNN. However, the CNN-LSTM achieves the best accuracy results for the null class (unknown) (89 %), light contact during dribbling (76 %), short pass medial (74 %), and shots (68 %). Similar to the CNN, light contacts are most frequently confused with the null class (unknown) (14 % of the cases). Short pass medial performs better with the CNN-LSTM, achieving 74% accuracy, compared to 68 % with the CNN. Short pass other is recognized almost equally well by the CNN-LSTM as by the CNN, with an accuracy of 43 %. However, it is misclassified as short pass medial in 30 % of cases and as light contact in 11 % of cases. As with the other model, the long pass is the worst-performing pass type, with an accuracy of only 20 %. It is most often misclassified as a shot (42 %) or short pass medial (26 %). Shots, on the other hand, are correctly identified 68 % of the time, with the most common confusion being with short pass medial (15 %). The ambiguous shot, as with the CNN, is also not correctly identified a single time by the CNN-LSTM.
The model also distinguishes between the left and right foot well, predicting correctly in 96 % of cases for both feet. However, this is slightly less accurate compared to the CNN, which achieved even better results.

In the following, a more detailed analysis of the scores is displayed, presenting the event differentiation in Figure 5.12 and foot distinction in Figure 5.13. The values have been normalized to a scale between 0 and 1.

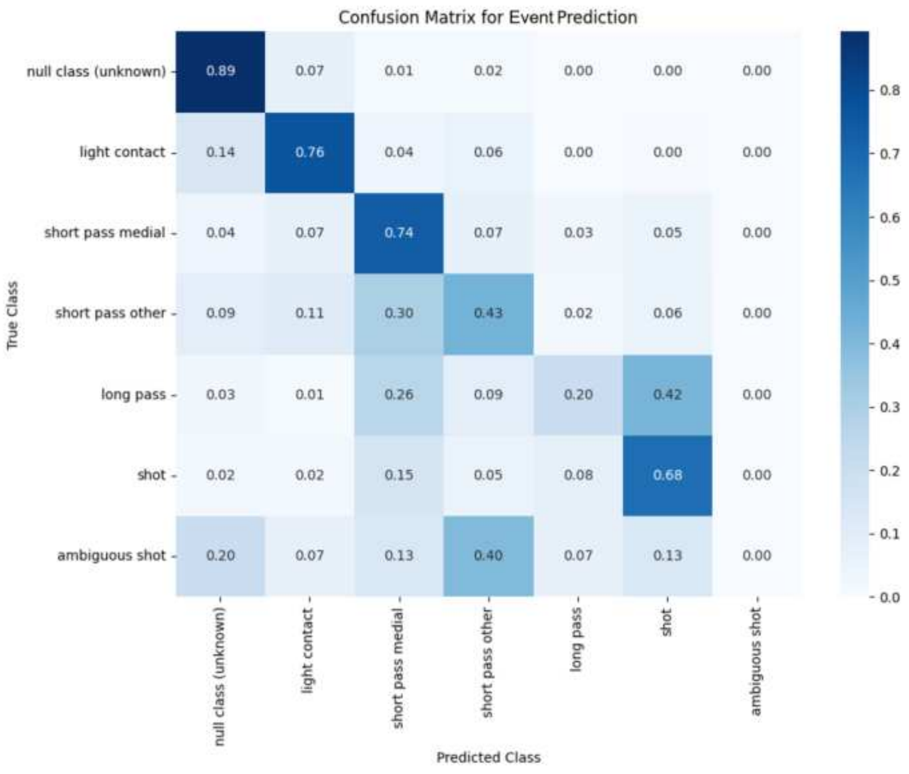


Figure 5.12: Confusion matrix for event predictions. The darker the colour, the more often the predicted class matches the true class. The matrix shows the classification performance, with the values along the diagonal representing correct predictions and the elements outside the diagonal representing wrong classifications. The values were normalized between 0 and 1, where 0 indicates no occurrences and 1 represents perfect classification for that class.

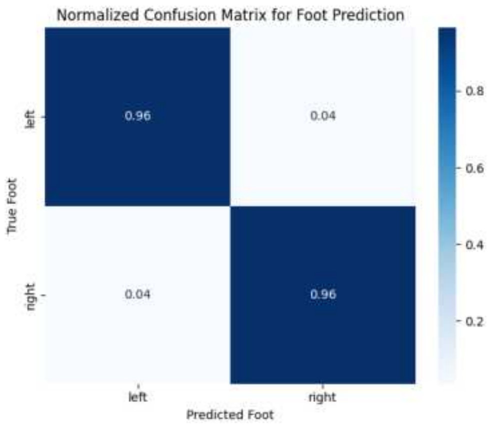


Figure 5.13: Confusion matrix for foot predictions. The values were normalized between 0 and 1, where 0 indicates no occurrences and 1 represents perfect classification for that class.

As noted in Chapter 4.1.2, the test dataset combines lab and real game data. The following sections detail the CNN-LSTM model's performance.

Laboratory Test Set

The following displays the confusion matrix for distinguishing between the various events in Figure 5.14 and the two feet in Figure 5.15, based only on the laboratory test dataset.

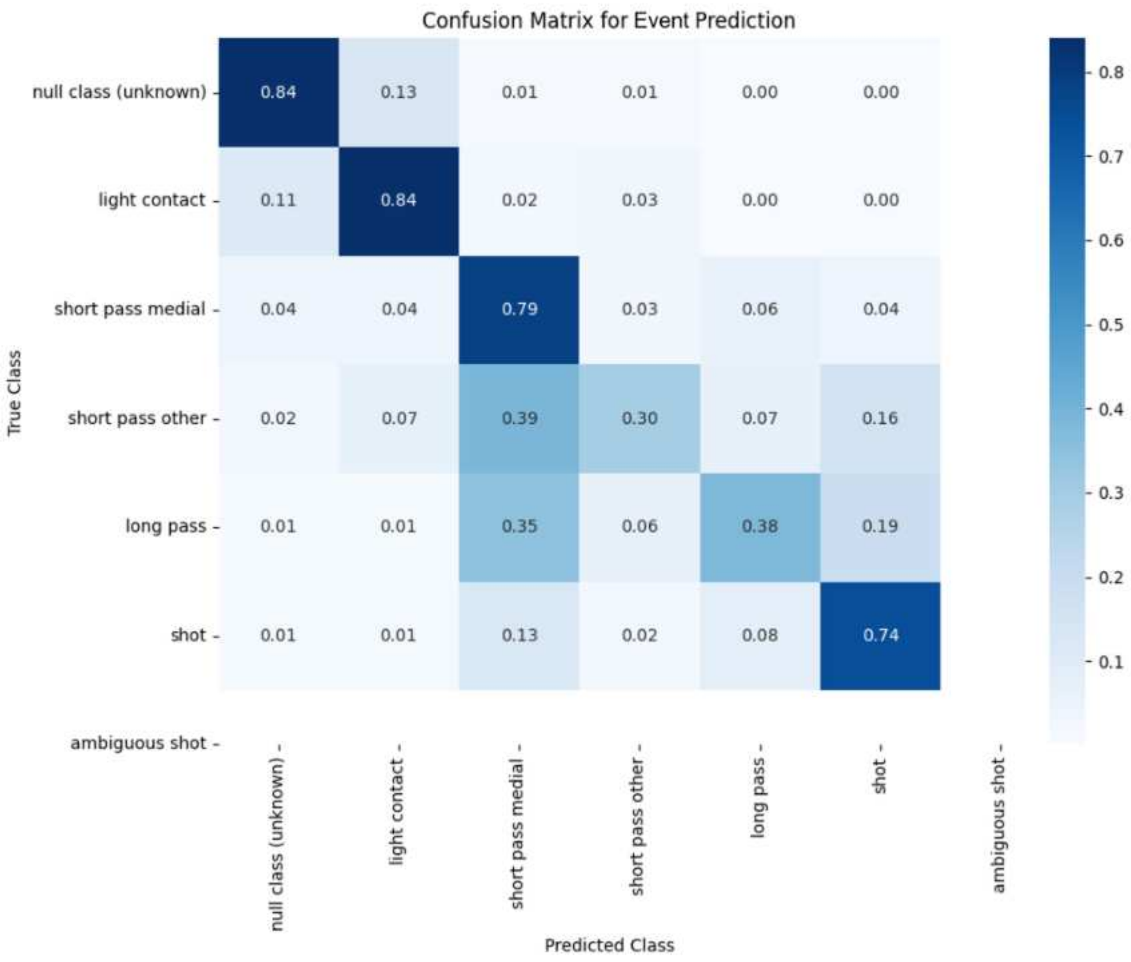


Figure 5.14: Confusion matrix for event predictions based on the laboratory test set. The values are normalized between 0 and 1.

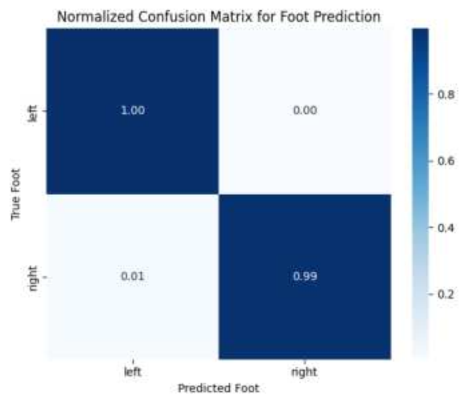
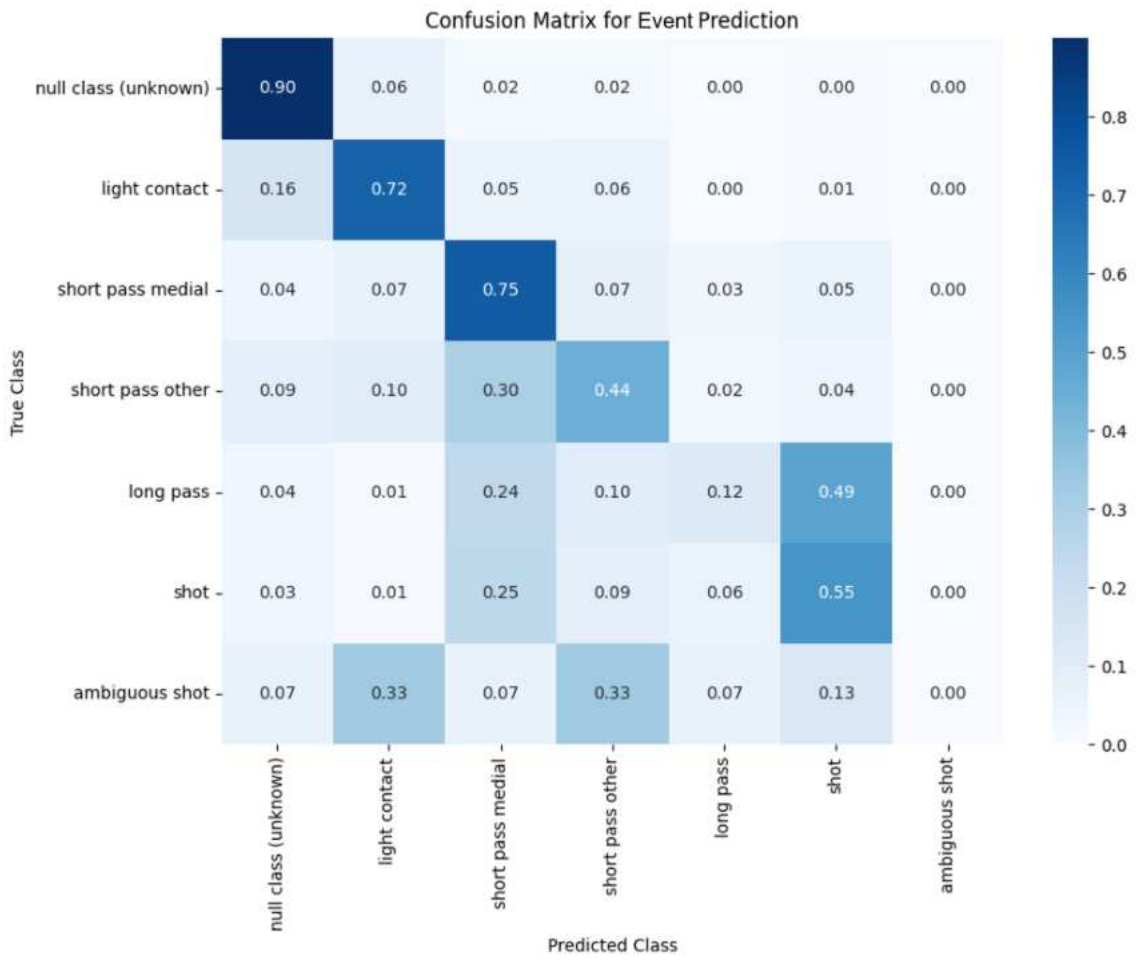


Figure 5.15: Confusion matrix for foot predictions based on the laboratory test set. The values are normalized between 0 and 1.

In the laboratory data, the null class (unknown) is classified just as well as the light contact class, both of which achieve an accuracy of 84 %. These two classes are most frequently confused with each other. The short pass medial class achieves an accuracy of 79 % and thus performs better than the overall performance shown in Figure 5.12 when training with the whole test set. In contrast, the short pass other is not well recognized with an accuracy of only 30 %. It is most frequently misinterpreted as a short pass medial (39 %) and sometimes also as a shot (16 %). The long pass performs slightly better with 38 % correct classifications, but is similarly frequently misclassified as a short pass medial (35 %) and occasionally as a shot (19 %). Shots perform relatively reliably with 74 % accuracy, although they are most frequently misclassified as a short pass medial (13 %). As mentioned above, ambiguous shots are not included in the test set. The model distinguishes the feet with very high accuracy in the laboratory dataset, correctly identifying 100 % of the left foot cases and 99 % of the right foot cases.

Real Game Test Set

The confusion matrix in Figure 5.16 and 5.17 illustrates the model's ability to differentiate between the various events and between the two feet, based only on the real game test dataset.



	null class (unknown)	light contact	short pass medial	short pass other	long pass	shot	ambiguous shot
null class (unknown)	0.90	0.06	0.02	0.02	0.00	0.00	0.00
light contact	0.16	0.72	0.05	0.06	0.00	0.01	0.00
short pass medial	0.04	0.07	0.75	0.07	0.03	0.05	0.00
short pass other	0.09	0.10	0.30	0.44	0.02	0.04	0.00
long pass	0.04	0.01	0.24	0.10	0.12	0.49	0.00
shot	0.03	0.01	0.25	0.09	0.06	0.55	0.00
ambiguous shot	0.07	0.33	0.07	0.33	0.07	0.13	0.00

Figure 5.16: Confusion matrix for event predictions based on the real game test set. The values are normalized between 0 and 1.

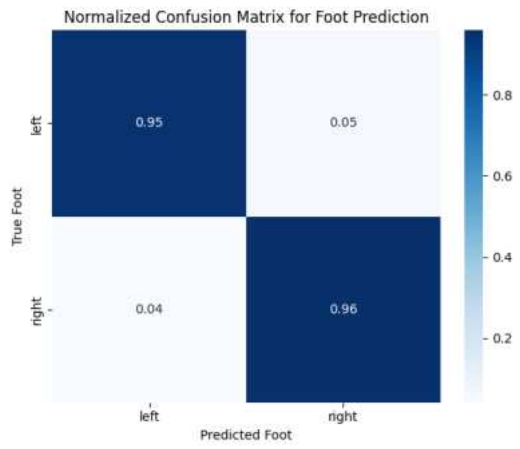


Figure 5.17: Confusion matrix for foot predictions based on the real game test set. The values are normalized between 0 and 1.

The CNN-LSTM model classifies the null class (unknown) in the real game data more accurately than in the laboratory data, achieving an accuracy of 90 %. However, light contacts are again classified less accurately than with the laboratory data, reaching only 72 %. Short pass medial is relatively well distinguishable from other classes with an accuracy of 75 %, whereas short pass other performs worse, with only 44 % correct classifications. In 30 % of cases, short pass other is misclassified as short pass medial. Notably, nearly half (49 %) of the long passes are misclassified as shots, with only 12 % correctly identified. A large portion (24 %) of long passes is also classified as short pass medial. Shots are correctly identified 55 % of the time, while ambiguous shots are never correctly classified; they are most frequently misclassified as light contacts and short pass other (both 33 %). The model correctly identifies the left foot in 95 % of cases and the right foot in 96 %.

Null-Pass-Shot Classification

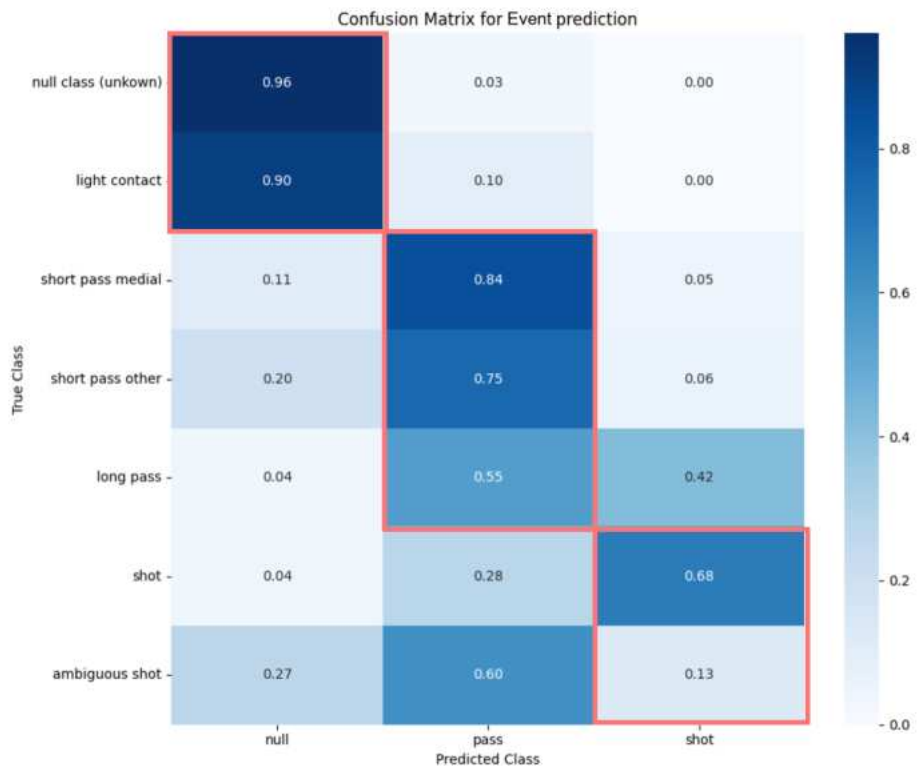


Figure 5.18: Confusion matrix for the grouped classes: null, pass and shot

When examining the overall distinction between the null class (including light contacts), passes, and shots based on the whole test dataset, we see that the null class achieves high accuracy, with 96 % for the unknown and 90 % for light contacts. Passes are also well differentiated from the null class, with only 3 % and 10 % misclassifications, respectively. However, distinguishing passes from the individual shot class is more challenging, with a misclassification rate of 28 %. The ambiguous shot category is classified as a pass in 60 % of cases. Shots are better separated from the null class than passes, with 68 % correctly classified, though they are most often confused with long passes (42 %).

Chapter 6

Discussion

To explore whether a DL model can effectively recognize specific soccer activities, such as dribbling, different types of passes, and automatic left and right foot recognition using two IMUs embedded in the sole of a soccer shoe, this work implemented both a CNN and a CNN-LSTM model. Compared to traditional ML approaches, these DL models bypass manual, subjective, and time-consuming feature extraction processes, as they can autonomously learn relevant features [Cup22]. For this reason, DL approaches are increasingly utilized in HAR tasks, and their application is expanding into sports contexts like soccer [Cup22]. However, despite showing promising results in classifying soccer activities, many DL approaches still focus on general actions and make limited use of real game data, which restricts their practical applicability [Sto21]. Therefore, this work aimed to develop a DL approach capable of recognizing these more specific soccer activities, as well as automatic left and right foot recognition. The main objectives were:

i) Can deep learning algorithms classify activity metrics in soccer?
ii) Can the analyzed algorithms continue to recognize the metrics even in game-like situations?

The following discussion reviews this thesis's findings, focusing on the preprocessing and segmentation steps, the choice of model architecture and its comparison with existing literature, hyperparameter optimization, and model performance results. Additionally, the applicability of the models in real-game scenarios is addressed.

6.1 Preprocessing and Segmentation

The preprocessing pipeline closely follows the methodologies introduced by Stoeve and colleagues [Sto21] as well as Cuperman and colleagues [Cup22], though several enhancements were incorporated to suit the specific requirements of our model better.
The definition of the null class was extended to improve the robustness of the model and its applicability in real-life game scenarios. In contrast to Stoeve and colleagues, who restricted their null class to "none", "light contacts" and "strong contacts", the null class of this model includes additional categories such as "ambiguous contact", "ambiguous short pass", "ambiguous range pass", and "juggling kicks". Therefore, in this study, the null class is defined to include activities that are considered irrelevant to the specific classification objectives. By assigning these ambiguous actions to the null class, the model avoids misinterpreting ambiguous activities as specific well-defined movements, such as ambiguous passes. This approach ensures a clear focus on the relevant, unambiguous action classes. Nevertheless, ambiguous shots were intentionally included in the relevant classes to gain insight into how the model handles such uncertain activities. By encompassing a wider variety of movement types, the model gains robustness and accuracy in the dynamic context of real games, improving its generalization capability.
To differentiate relevant from non-relevant events, Cuperman and colleagues used a threshold-based approach to filter out non-relevant events before passing them to the model, thereby reducing computational costs [Cup22]. However, this method has limitations, especially for subtle actions like dribbling. These actions can be challenging to distinguish from non-relevant ones, such as running, as they often involve light ball interactions that result in only small peaks in the sensor data. As a result, basic thresholding may struggle to consistently capture these finer ball touches that occur during dribbling.
In future work, an initial, simplified ML or DL model could be added to distinguish high-intensity from low-intensity activities. This extra layer could effectively filter out low-impact events while retaining high-impact ones, resulting in improved precision compared to a simple threshold approach. Further event detection methods should be explored in the future.

As described in Chapter 4.3, a peak detection method similar to that of Stoeve and colleagues to identify potential events was used [Sto21]. However, this method was adjusted for our specific application by optimizing the parameters min_dist, tresh_abs, and f_{cut}. The data used for optimization was excluded from further analysis for optimization. By omitting these data points, the model remains robust and can generalize to new data due to an objective assessment.
The peak detection algorithm was further optimized for sensitivity. Stoeve and colleagues set

a sensitivity threshold of $100\ \%$ for the train set, aiming to ensure that no single event goes undetected [Sto21]. This is appropriate for the isolated detection of shots and passes, as these are distinguishable from other events due to prominent peaks in the signal. In our case, the average sensitivity for peak detection optimization was $91\ \%$ for the selected data as described in Chapter 4.1.2. Additionally, we set limits on the parameter ranges for min_dist and tresh_abs to prevent overly small parameter values. The optimized peak detection was further tested on the entire test set, achieving an accuracy of $89.54\ \%$, indicating an appropriate event detection. The undetected events are light ball contacts with minimal amplitude, which are not critical for detecting actual dribbling actions, as a light touch of the ball alone, by our definition, does not constitute dribbling. Stoeve and colleagues used parameter values of min_dist $= 300$ samples, tresh_abs $= 0.3$, and $f_{\text{cut}} = 20$ Hz [Sto21]. In contrast, our parameters are set to min_dist $= 100$ samples, tresh_abs $= 0.1$, and $f_{\text{cut}} = 22.7$ Hz. These differences are reasonable considering our analysis includes dribbling, which has a smaller minimum distance between events according to Stoeve and colleagues [Sto21]. This also aligns with the lower tresh_abs, as dribbling contacts typically produce smaller amplitude peaks in the signal than shots or passes.
The slightly higher cutoff frequency f_{cut} is due to our aim of capturing faster, higher-frequency peaks, as seen in dribbling. A high-pass filter with a higher cutoff frequency supports this objective by allowing high-frequency peaks to pass through more effectively, making relevant dribbling events more clearly visible in the signal.

To balance the dataset, Stoeve and colleagues, as well as Cuperman and colleagues, both used undersampling methods [Sto21; Cup22]. Given the greater class imbalance in our dataset due to finer class distinctions, a combined approach of random undersampling and ADASYN oversampling was applied to improve balance and maintain sufficient sample size for underrepresented classes. By applying random undersampling, many samples from the null class are removed, while ADASYN synthesizes new instances for underrepresented classes without merely duplicating them. This method has shown promising results in similar applications , as it generates more varied synthetic samples, which can help the model capture finer-grained patterns in complex actions such as ambiguous shots, short passes, and long passes [Lar23].
While ADASYN is effective in creating instances for less frequent classes, a potential limitation is verifying the realism of the generated data, particularly for fine-grained distinctions like specific types of passes or shots. To mitigate this risk, the synthetic data are used strictly in model training, leaving the test set unaffected.
In conclusion, the combined undersampling and ADASYN approach is well-suited to the demands

of this study, as it enables the model to learn essential features from a more balanced dataset while maintaining class variability. Future work could benefit from collecting additional data for underrepresented classes, as real data would reduce the need for synthetic sampling and further enhance model robustness.

6.2 Choice of Model Architecture

Looking at the architecture results of our two models, the CNN, with a weighted F1 score of 92.5 % for event and 97.8 % for foot classification, outperformed the CNN-LSTM, which achieved 90.1 % for event and 96.3 % for foot detection. Both models have the same classification behaviour, for instance, they both recognize the short pass medial better than the short pass other. This suggests that the CNN-LSTM does not offer a distinct advantage over the standalone CNN. However, it should be noted that, since the event windows were treated as independent and the cell state was neither retained nor initialized for the next sequence, only short-term dependencies were analyzed. For future research, it would be interesting also to consider long-term dependencies.
An interesting finding is that, during hyperparameter optimization, the CNN components in both models delivered very similar results. Both models achieved optimal performance with four convolutional blocks, no dense dropout at the end, a kernel size of five, and nearly identical filter counts (112 for the CNN and 114 for the CNN-LSTM). This suggests that the data processing in the CNN part of each model might be nearly identical, with the LSTM component in the CNN-LSTM likely responsible for the slight performance drop. Furthermore, the CNN-LSTM requires a much higher number of trainable parameters, reaching 828,840 compared to CNN's 210,792. This substantial difference, almost four times more, is associated with longer runtime.
Considering both the higher runtime and lower performance, the CNN is the more suitable choice for practical applications or future product integration, as it allows for faster and safer activity recognition.
While the current approach has yielded good results, there is still room for improvement. Exploring different ways to capture time-dependent patterns could provide valuable benefits for accurately classifying activities in dynamic sports like soccer. Therefore, other models like Time Convolutional Networks (TCNs) with attention mechanisms should be explored in the future, as they are more effective at understanding time-dependent patterns and adapting their receptive field flexibly. Additionally, the use of attention allows these models to assign higher weights to important features, which can enhance overall model performance [Wei24]. This approach has already demonstrated improved results in the field of HAR in soccer image processing tasks

[Wan23; Mou24] but also with IMU data [Wei24].
Additionally, Bijalwan and colleagues recently introduced a multi-modal fusion model approach for HAR using IMUs, which synergistically integrates TCNs, CNNs, and LSTMs. Each network leverages its unique strengths: TCNs capture temporal dependencies, CNNs extract local features, and LSTMs manage sequential information. This combined approach outperforms the individual models in both accuracy and F1 score, highlighting the advantages of integrating these architectures [Bij24]. In the next step, this type of architecture could be applied to our dataset to achieve even better classification results.

6.3 Model Comparison to Existing Literature

Our CNN achieved a weighted F1 score of 92.54 % whereas the CNN-LSTM reached a weighted F1 score of 87.54 %. As already mentioned in Chapter 4.4, our model selection was influenced by the results of Stoeve and colleagues and Cuperman and colleagues [Sto21; Cup22].
Cuperman and colleagues achieved an average accuracy of 97.53 % with their CNN-LSTM model and 98.08 % with their CNN model. This shows that the CNN performed slightly better than the CNN-LSTM in their study. Nevertheless, it should be noted that for both approaches, the CNN model was based on a combination of different sub-networks, where the sensors were processed differently through various filter configurations (either using different filters for each sensor, the same filters for all sensors, or across all sensors at once), and the extracted feature maps from all sub-networks were concatenated at the end [Cup22]. Although a direct comparison of the results is not possible due to the different dataset, this similarity in architecture, particularly the number of convolutional layers, indicates that our model has a robust structure that can achieve strong performance. Nevertheless, this fusion approach resulted in a more complex model architecture compared to ours. This also shows that multi-modal fusion architectures can be promising for the future [Bij24].
Stoeve and colleagues achieved a strong performance with a weighted F1 score of 92.8 % for their CNN model [Sto21]. Their model consists of three convolutional layers, each followed by a max pooling layer, with the last two layers also including dropout. This architecture is broadly similar to mine, and since the dataset is the same, this makes the comparison more relevant. Notably, Stoeve's model uses only three convolutional layers, while we found four to be optimal. Additionally, in our model, we use max pooling only after the first convolutional layer and apply both dropout and max pooling for all subsequent layers, making the results even more comparable. This suggests that finer granularity requires a more complex model structure to capture subtle

differences effectively. Consequently, Stoeve's model had only 111,763 trainable parameters, while ours had nearly doubled at 210,792, indicating a higher computational load.

6.4 Hyperparameter Optimization

To find the best hyperparameters, Stoeve and colleagues and Cuperman and colleagues also conducted hyperparameter optimization, both using 5-fold cross-validation [Sto21; Cup22]. In contrast, in this work, a 5-fold stratified cross-validation was applied.
Stratification plays a crucial role here, as it not only enhances the comparability of validation results across folds but also provides a more accurate assessment of model performance by evaluating on a distribution that reflects the "real-world" class balance. Additionally, for highly imbalanced datasets, stratification helps ensure that each fold contains instances of all classes, avoiding cases where a fold might lack representation for one or more classes entirely.
Unlike Stoeve and colleagues, we optimized with only 100 trials instead of 500 [Sto21]. In the future, further optimization could be explored for our fixed parameters, such as increasing the number of trials and extending the maximum number of epochs, as early stopping was not always triggered, suggesting that additional training could be beneficial.

6.5 Outcomes of Event Classification

The fine differentiation between events works well for both models, as shown in Chapter 5. The CNN achieved thereby a weighted F1 score of 92.54 %. Since both models exhibit similar behaviours, we will now address both models simultaneously. Although the classification generally performs well, some classes yield better results than others. Relevant actions are distinguishable from those categorized as null class, which represent non-relevant actions. The null class is most often confused with dribbling contacts, likely because these ball contacts are usually very light, resulting in only slight signal amplitude variations.
Within the passing classes, it is notable that short pass medial and short pass other are frequently misclassified as each other. In general, short pass medial is more often correctly recognized, while short pass other is often mislabeled as short pass medial. Although this may seem unexpected at first due to the different contact points on the foot, it is important to note that the class short pass other is broadly defined and includes any part of the foot that comes into contact with the ball except the medial side. This broad definition makes it difficult to identify unique features accurately representing this class. Nonetheless, both actions have the same goal: to pass the ball

to a teammate over a short distance. Consequently, while foot positioning may vary slightly, the intensity, or signal amplitude, is very similar due to the short distance, complicating differentiation. Additionally, the angle at which the foot strikes the ball can vary in short passes, regardless of which part of the foot makes contact with the ball [Lee10]. It is also challenging for the labeller to determine exactly which part of the foot made contact, especially during real game sessions where multiple players are on the field, making it difficult to observe each detail precisely. Another observation is that long passes are rarely classified correctly and are most often confused with either shots or short pass medial. This is because a long pass may resemble a shot in force if it's harder, or a short pass if it's softer. Additionally, the similar motion in long passes, short passes, and shots contributes to these confusions. Explainable AI methods could be explored in the future to gain deeper insights into the decision-making process of the models, thereby providing a clearer understanding of the factors that contribute to misclassification [Pro23].
As discussed in Chapter 6.2, incorporating an attention mechanism could be beneficial, as it is often used to emphasize certain parts of a sequence and assign them more weight [Wei24]. This might place more emphasis on the moments leading up to the critical motion phases, such as just before ball contact, thereby allowing for better distinctions. Ambiguous shots are particularly challenging to differentiate due to their subjective assignments. In many cases, a shooting motion is present in this class, but the ball may not be fully touched, or unusual types of shooting motions are performed. It was expected that the shooting motion could be detected by the DL model through the rapid acceleration of the leg, even if it did not result in a pronounced peak in the IMU signal due to no ball contact. This signal pattern could indicate a shot, but the amplitude does not, making it ambiguous. Ultimately, the classification of such events strongly depends on the labeller's interpretation of whether it is a clear shot or whether the action fits into the ambiguous class.

6.6 Outcomes of Event Leg Classification

The distinction between the left and right foot works very well, with an achieved weighted F1 score of 97.83 % by the CNN. As shown in Figure 4.4, where the execution of a long pass is demonstrated for both left and right feet, the acceleration and gyroscope data appear horizontally mirrored. This makes it quite easy for the model to differentiate between left and right foot actions, as the movements are fundamentally similar. This reliable distinction can be used to gain valuable insights into player performance, such as how often a player uses one foot compared to the other. Based on this information, training can be adjusted to focus more on the less frequently used foot.

Furthermore, future activity recognition could incorporate the analysis of the non-dominant leg. Although this leg does not directly perform actions, such as during a shot, it might still provide valuable additional insights. Research should be conducted to explore the potential benefits of including this data from the supporting leg.

6.7 High-Level Class Comparison

Although the primary aim of this work was to gain deeper insights into more specific soccer activities, the general identification of null, pass, and shot events remains of interest. Summarizing the events in higher-level classes allows us to compare the results with those of Stoeve and colleagues. Overall, the findings are closely aligned with those of Stoeve and colleagues, who reported an average performance of 90 % for the null class, 77 % for the pass class, and 75 % for the shot class [Sto21]. In comparison, our results showed similar average performances: 94 % for the null class, 71 % for the pass class, and 70 % for the shot class (ambiguous shots excluded). Even if the performance scores are slightly different, the more detailed class breakdown does not change the general challenges in distinguishing between null, pass, and shot actions. In summary, the null class remains highly recognizable and distinguishable from the others, likely because it includes easily recognizable actions like jogging or running. The pass class is also clearly separable from the null class, but the models struggle to distinguish it from the shot class, leading to frequent misclassifications. The shot class, in turn, is easily distinguished from both the null and pass classes, except for long passes. Long passes are misclassified as shots in 42 % of cases for our CNN-LSTM model and 36 % for our CNN model, likely due to the similar technique and force involved in long passes and shots [Lev98].

6.8 Real Game Applicability

The goal was to ensure that the activities could also be reliably detected in real-game scenarios. A key consideration here is the real-game applicability of these setups. Our setup is designed to have minimal impact on the player, using only two IMUs to enable more practical, product-oriented applications in real sports settings. While many studies have employed configurations with multiple IMUs attached to various parts of the body, such setups may be less practical for in-game tracking, as players are unlikely to wear multiple sensors during training or competition without impacting their movement [Cup22; Hos17; Lar23].

Additionally, as shown in the confusion matrices of both models in Chapter 5.3.1 and Chapter

5.3.2, performance on the lab data is better than on the real game data. This was expected, as actions in the lab setting are less complex, movements are more regular, and events are thus easier to separate. Additionally, there is no opponent interaction as in real games, which further reduces complexity in the lab environment [Sto21]. The impact of opponents could be further explored in the future.

Although the weighted F1 score for the lab data set is slightly lower (86.54 %) compared to the real game data set (91.43 %) as achieved by the CNN, this difference can be attributed to the varying instance counts across classes. For example, the real game test dataset contains about ten times more instances of the null class than the laboratory test dataset, artificially boosting the weighted F1 score. However, if we examine the recognition rates of the individual classes through the confusion matrices, performance is indeed better in the laboratory setting compared to the real game setting. This is also highlighted by the balanced accuracy metric, which gives equal weight to each class regardless of its frequency. Classes with fewer instances, such as pass and shot, are equally considered in the evaluation. As seen in Tables 5.3 and 5.5, the balanced accuracy is higher for lab data, indicating a better classification performance of the model under lab conditions.

In summary, although the model's performance is lower on real game data, the weighted F1 score indicates that it can still reliably classify actions under real-world conditions. Both the lab and real game data play a crucial role: lab data enables the model to learn the movements more precisely, while real game data reflects the complexity of actual gameplay.

To further improve classification performance, additional data should be collected in both lab and real game environments. Especially for classes with fewer instances, the model's performance decreased, as seen with ambiguous shots or long passes. Consequently, with additional data, the model could potentially learn the class-specific features more effectively. To achieve even higher accuracy in classifying specific events, especially since individual events like long passes do not always achieve high accuracy, another promising approach would be to use transfer learning, as it can accelerate model training and improve prediction accuracy [Mou24]. Transfer Learning has already shown improvements in HAR based on wearable sensors and could therefore lead to potential improvements for our task [Jia24; Gan24]. Another step could involve providing the model with additional context data, such as the player's position, age, and similar information.

Chapter 7

Conclusion

In this thesis, we investigated whether DL models can accurately identify soccer-specific, fine-grained events, such as dribbling, types of passes, types of shots, and left and right foot recognition, using only two IMUs embedded in the soles of each soccer shoe. Additionally, we explored whether these DL models could recognize these events in game-like situations.

The dataset included data from over 800 players. Compared to Stoeve and colleagues, this study broadened the class definitions, particularly for the null class, to encompass a wider range of real-life actions [Sto21]. A peak detection algorithm, based on Schuldhaus [Sch19], was implemented with optimized hyperparameters to detect and extract finer events effectively. To address the highly imbalanced dataset, a class-balancing method was applied, which either randomly undersamples a class or generates synthetic instances using ADASYN.
Two models, CNN and CNN-LSTM, were implemented and optimized through hyperparameter tuning. The classification outcomes of these models were then analyzed in detail, focusing on event classification and foot recognition performance. To provide a more comprehensive overview of general performance, events were grouped into broader categories, such as null, pass, and shot, making the results comparable with those of Stoeve and colleagues [Sto21]. To assess real-game applicability, the test dataset was further divided into lab and real-game sessions for detailed analysis.

The results of this thesis demonstrate that the proposed DL models can reliably identify most soccer-specific metrics. Challenges remain with the class long pass and class short passes other. Both models showed strong performance, with the CNN achieving a weighted F1 score of 92.54 % for event detection and 97.83 % for foot recognition, while the CNN-LSTM achieved a weighted F1

score of 90.08 % for event detection and 96.27 % for foot recognition. Future work could explore alternative model architectures, such as TCNs or multi-modal fusion architectures. Additionally, attention mechanisms could be integrated to enhance performance further.
In terms of real-game applicability, the model performs better on lab data but handles real-game data similarly well. The higher weighted F1 score on real-game data compared to the laboratory data suggests that the model effectively handles the strong class imbalance present in real-game scenarios.

In this work, a first step was taken towards classifying more specific events to gain deeper insights into player performance. This paves the way for more effective training and improved player analyses, even in real-game scenarios, as it enables training methods and game strategies to be more individually tailored to the player. By understanding deeper aspects of player performance, strengths and potential weaknesses can be better identified, allowing for more targeted improvement plans for each player.

List of Figures

List of Tables

Bibliography

[adi23] adidas. *"FUSSBALLLIEBE": EM-Spielball von adidas*. Accessed on 16.08.2024. 2023. URL: https://www.dfb.de/news/detail/fussballliebe-em-spielball-von-adidas-256779/.

[Ahm23] Shams Forruque Ahmed, Md Sakib Bin Alam, Maruf Hassan, Mahtabin Rodela Rozbu, Taoseef Ishtiak, Nazifa Rafa, M Mofijur, A B M Shawkat Ali, and Amir H Gandomi. "Deep learning modelling techniques: current progress, applications, advantages, and challenges". en. In: *Artif. Intell. Rev.* 56.11 (Nov. 2023), pp. 13521–13617.

[Aki19] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. "Optuna: A Next-generation Hyperparameter Optimization Framework". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. KDD '19. Anchorage, AK, USA: Association for Computing Machinery, 2019, pp. 2623–2631. ISBN: 9781450362016. DOI: 10.1145/3292500.3330701. URL: https://doi.org/10.1145/3292500.3330701.

[Alo18] Omar Alobaid, Lakshmish Ramaswamy, and Khaled Rasheed. "A machine learning approach for identifying soccer moves using an accelerometer sensor". In: *2018 International Conference on Computational Science and Computational Intelligence (CSCI)*. Las Vegas, NV, USA: IEEE, Dec. 2018.

[Bij24] Vishwanath Bijalwan, Abdul Manan Khan, Hangyeol Baek, Sangmin Jeon, and Youngshik Kim. "Interpretable human activity recognition with temporal convolutional networks and model-agnostic explanations". In: *IEEE Sens. J.* 24.17 (Sept. 2024), pp. 27607–27617.

[Cha18] Nitin Kumar Chauhan and Krishna Singh. "A review on conventional machine learning vs deep learning". In: *2018 International Conference on Computing, Power and Communication Technologies (GUCON)*. Greater Noida, Uttar Pradesh, India: IEEE, Sept. 2018.

[Che19] Yong Chen, Shuai Zhang, Wenyu Zhang, Juanjuan Peng, and Yishuai Cai. “Multifactor spatio-temporal correlation model based on a combination of convolutional neural network and long short-term memory neural network for wind speed forecasting”. en. In: *Energy Convers. Manag.* 185 (Apr. 2019), pp. 783–799.

[Che21] Leiyu Chen, Shaobo Li, Qiang Bai, Jing Yang, Sanlong Jiang, and Yanming Miao. “Review of image classification algorithms based on convolutional neural networks”. en. In: *Remote Sens. (Basel)* 13.22 (Nov. 2021), p. 4712.

[Che23] Longwen Chen and Dean Hu. “An effective swimming stroke recognition system utilizing deep learning based on inertial measurement units”. en. In: *Adv. Robot.* 37.7 (Apr. 2023), pp. 467–479.

[Cup22] Rafael Cuperman, Kaspar M B Jansen, and Michał G Ciszewski. “An end-to-end deep learning pipeline for football activity recognition based on wearable acceleration sensors”. en. In: *Sensors (Basel)* 22.4 (Feb. 2022), p. 1347.

[Gan24] H S Ganesha, Rinki Gupta, Sindhu Hak Gupta, and Sreeraman Rajan. “Few-shot transfer learning for wearable IMU-based human activity recognition”. en. In: *Neural Comput. Appl.* 36.18 (June 2024), pp. 10811–10823.

[Gra19] Margarita Granat. *How to Use Convolutional Neural Networks for Time Series Classification.* Accessed on 20.08.2024. 2019. URL: https://towardsdatascience.com/how-to-use-convolutional-neural-networks-for-time-series-classification-56b1b0a07a57.

[Hal23] Nils Haller, Stefan Kranzinger, Christina Kranzinger, Julia C Blumkaitis, Tilmann Strepp, Perikles Simon, Aleksandar Tomaskovic, James O’Brien, Manfred Düring, and Thomas Stöggl. “Predicting injury and illness with machine learning in elite youth soccer: A comprehensive monitoring approach over 3 months”. en. In: *J. Sports Sci. Med.* 22.3 (Sept. 2023), pp. 476–487.

[Han18] Su-Hyun Han, Ko Woon Kim, Sangyun Kim, and Young Chul Youn. “Artificial neural network: Understanding the basic concepts without mathematics”. en. In: *Dement. Neurocognitive Disord.* 17.3 (Sept. 2018), pp. 83–89.

[He08] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. “ADASYN: Adaptive synthetic sampling approach for imbalanced learning”. In: *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence).* Hong Kong, China: IEEE, June 2008.

[Hos17] H M Sajjad Hossain, Md Abdullah Al Hafiz Khan, and Nirmalya Roy. “SoccerMate: A personal soccer attribute profiler using wearables”. In: *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. Kona, HI: IEEE, Mar. 2017.

[Hsu18] Yu-Liang Hsu, Shih-Chin Yang, Hsing-Cheng Chang, and Hung-Che Lai. “Human daily and sport activity recognition using a wearable inertial sensor network”. In: *IEEE Access* 6 (2018), pp. 31715–31728.

[Isl20] Md Zabirul Islam, Md Milon Islam, and Amanullah Asraf. “A combined deep CNN-LSTM network for the detection of novel coronavirus (COVID-19) using X-ray images”. en. In: *Inform. Med. Unlocked* 20.100412 (Aug. 2020), p. 100412.

[Jan21] Christian Janiesch, Patrick Zschech, and Kai Heinrich. “Machine learning and deep learning”. en. In: *Electron. Mark.* 31.3 (Sept. 2021), pp. 685–695.

[Jay24] Brindha Jayakumar and Nallavan Govindarajan. “Multi-sensor fusion based optimized deep convolutional neural network for boxing punch activity recognition”. en. In: *Proc. Inst. Mech. Eng. P. J. Sport. Eng. Technol.* (Mar. 2024).

[Jha22] Debesh Jha, Ashish Rauniyar, Havard D Johansen, Dag Johansen, Michael A Riegler, Pal Halvorsen, and Ulas Bagci. “Video analytics in elite soccer: A distributed computing perspective”. In: *2022 IEEE 12th Sensor Array and Multichannel Signal Processing Workshop (SAM)*. Trondheim, Norway: IEEE, June 2022.

[Jia24] Qi Jia, Jing Guo, Po Yang, and Yun Yang. “A holistic multi-source transfer learning approach using wearable sensors for personalized daily activity recognition”. en. In: *Complex Intell. Syst.* 10.1 (Feb. 2024), pp. 1459–1471.

[Kin14] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: (2014). eprint: 1412.6980 (cs.LG).

[Kon22] Yuki Kondo, Shun Ishii, Hikari Aoyagi, Tahera Hossain, Anna Yokokubo, and Guillaume Lopez. “FootbSense: Soccer moves identification using a single IMU”. In: *Smart Innovation, Systems and Technologies*. Smart innovation, systems and technologies. Singapore: Springer Nature Singapore, 2022, pp. 115–131.

[Koş23] Enes Koşar and Billur Barshan. “A new CNN-LSTM architecture for activity recognition employing wearable motion sensor data: Enabling diverse feature extraction”. en. In: *Eng. Appl. Artif. Intell.* 124.106529 (Sept. 2023), p. 106529.

[Kru24] Karina Kruse, Wolfgang Sauerwein, Jörn Lübben, and Richard Dodel. “Smart technologies and textiles and their potential use and application in the care and support of elderly individuals: A systematic review”. en. In: *Rev. Adv. Mater. Sci.* 63.1 (July 2024).

[Lar23] Aske Gye Larsen and Giovanni Papi. “Prediction of football actions and identification of optimal sensor placements using a semi-supervised learning approach”. MA thesis. Aalborg, Denmark: Aalborg University, May 2023.

[Lee10] A Lees, T Asai, T B Andersen, H Nunome, and T Sterzing. “The biomechanics of kicking in soccer: a review”. en. In: *J. Sports Sci.* 28.8 (June 2010), pp. 805–817.

[Lee22] Arie-Willem de Leeuw, Rick van Baar, Arno Knobbe, and Stephan van der Zwaard. “Modeling match performance in elite volleyball players: Importance of jump load and strength training characteristics”. en. In: *Sensors (Basel)* 22.20 (Oct. 2022), p. 7996.

[Lev98] J Levanon and J Dapena. “Comparison of the kinematics of the full-instep and pass kicks in soccer”. en. In: *Med. Sci. Sports Exerc.* 30.6 (June 1998), pp. 917–927.

[Lil20] Timothy P Lillicrap, Adam Santoro, Luke Marris, Colin J Akerman, and Geoffrey Hinton. “Backpropagation and the brain”. en. In: *Nat. Rev. Neurosci.* 21.6 (June 2020), pp. 335–346.

[Liv20] Ioannis E Livieris, Emmanuel Pintelas, and Panagiotis Pintelas. “A CNN–LSTM model for gold price time-series forecasting”. en. In: *Neural Comput. Appl.* 32.23 (Dec. 2020), pp. 17351–17360.

[Man23] Udomporn Manupibul, Ratikanlaya Tanthuwapathom, Wimonrat Jarumethitanont, Panya Kaimuk, Weerawat Limroongreungrat, and Warakorn Charoensuk. “Integration of force and IMU sensors for developing low-cost portable gait measurement system in lower extremities”. en. In: *Sci. Rep.* 13.1 (June 2023), p. 10653.

[Mic23] Marie-Florine Michel, Olivier Girard, Vincent Guillard, and Cyril Brechbuhl. “Well-being as a performance pillar: a holistic approach for monitoring tennis players”. en. In: *Front. Sports Act. Living* 5 (Sept. 2023), p. 1259821.

[Mos21] Ahmed Hesham Mostafa, Hala Abdel-Galil, and Mohamed Belal. “Ensemble model-based weighted categorical cross-entropy loss for facial expression recognition”. In: *2021 Tenth International Conference on Intelligent Computing and Information Systems (ICICIS)*. Cairo, Egypt: IEEE, Dec. 2021.

[Mou24] Chuan Mou. “The attention mechanism performance analysis for football players using the internet of things and deep learning”. In: *IEEE Access* 12 (2024), pp. 4948–4957.

[Nic24] Mark Nicholls, Derik Coetzee, Robert Schall, and Wilbur Kraak. “Analysing match-related performance indicators in Super Rugby Competitions: A study of the 2017–2019 seasons”. en. In: *Int. J. Sports Sci. Coach.* 19.3 (June 2024), pp. 1066–1081.

[Pap20] David Paper. “Predictive modeling through regression”. In: *Hands-on Scikit-Learn for Machine Learning Applications*. Berkeley, CA: Apress, 2020, pp. 105–136.

[Per13] Jürgen Perl, Andreas Grunz, and Daniel Memmert. “Tactics Analysis in Soccer – An Advanced Approach”. In: *International Journal of Computer Sience in Sport* 12 (Jan. 2013).

[Pro23] Andria Procopiou and Andriani Piki. “The 12th player: Explainable artificial intelligence (XAI) in football: Conceptualisation, applications, challenges and future directions”. In: *Proceedings of the 11th International Conference on Sport Sciences Research and Technology Support*. Rome, Italy: SCITEPRESS - Science and Technology Publications, 2023, pp. 213–220.

[Rei21] Brian Reilly, Oliver Morgan, Gabriela Czanner, and Mark A Robinson. “Automated classification of changes of direction in soccer using inertial measurement units”. en. In: *Sensors (Basel)* 21.14 (July 2021), p. 4625.

[Sar21] Vangelis Sarlis, Vasilis Chatziilias, Christos Tjortjis, and Dimitris Mandalidis. “A Data Science approach analysing the Impact of Injuries on Basketball Player and Team Performance”. en. In: *Inf. Syst.* 99.101750 (July 2021), p. 101750.

[Sch19] Dominik Schuldhaus. “Human Activity Recognition in daily life and sports using inertial sensors”. PhD thesis. Nov. 2019.

[Shi23] Farhad Mortezapour Shiri, Thinagaran Perumal, Norwati Mustapha, and Raihani Mohamed. “A comprehensive overview and comparative analysis on deep learning models: CNN, RNN, LSTM, GRU”. In: (2023).

[Sto21] Maike Stoeve, Dominik Schuldhaus, Axel Gamp, Constantin Zwick, and Bjoern M Eskofier. “From the laboratory to the field: IMU-based shot and pass detection in football training and game scenarios using deep learning”. en. In: *Sensors (Basel)* 21.9 (Apr. 2021), p. 3071.

[Wan23] Xin Wang and Yingqing Guo. “The intelligent football players’ motion recognition system based on convolutional neural network and big data”. en. In: *Heliyon* 9.11 (Nov. 2023), e22316.

[Wei24] Xiong Wei and Zifan Wang. “TCN-attention-HAR: human activity recognition based on attention mechanism time convolutional network”. en. In: *Sci. Rep.* 14.1 (Mar. 2024), p. 7414.

[Yad24] Hemant Yadav and Amit Thakkar. “NOA-LSTM: An efficient LSTM cell architecture for time series forecasting”. en. In: *Expert Syst. Appl.* 238.122333 (Mar. 2024), p. 122333.

[Zay18] Amer Zayegh and Nizar Al Bassam. “Neural Network Principles and Applications”. In: *Digital Systems*. IntechOpen, Nov. 2018.

[Zhu20] Jinsong Zhu and Jinbo Song. “An intelligent classification model for surface defects on cement concrete bridges”. en. In: *Appl. Sci. (Basel)* 10.3 (Feb. 2020), p. 972.

Appendix A

Acronyms

ADASYN Adaptive Synthetic Sampling Approach

AI Artificial Intelligence

ANN Artificial Neural Network

CNN Convolutional Neural Network

COD Change of Direction

DFT Discrete Fourier Transformation

DL Deep Learning

DNN Deep Neural Network

DT Decision Tree

FC Fully Connected

GAN Generative Adversarial Network

GRU Gated Recurrent Units

HAR Human Activity Recognition

IMU Inertial Measurement Unit

k-NN k-Nearest Neighbour

LSTM Long Short-Term Memory

ML Machine Learning

NB Naive Bayes

NN Neural Network

RBM Restricted Boltzmann Machine

ReLU Rectified Linear Unit

RNN Recurrent Neural Network

SVM Support Vector Machine

TCN Time Convolutional Network

UEFA Union of European Football Associations

VAR Video Assistant Referee