CHAPTER III

METHODOLOGY

3.1. Overview

Boxing activity recognition has gained significant attention in sports analytics, particularly with the advent of wearable sensors and deep learning techniques. Traditional methods relying on video-based motion tracking are often constrained by high costs, occlusions and real-time processing limitations. To address these challenges, the research in this study have explored Inertial Measurement Unit (IMU) sensors and smartphone-based motion tracking for efficient boxing punch recognition, activity classification and fatigue detection. This thesis focuses on leveraging deep learning models—including DCNN**,** Bi-LSTM and Attention Mechanisms—to enhance boxing performance analysis. The methodology is divided into three key components. Section 3.2 presents a multi-sensor fusion DCNN model utilizing IMUs placed on both wrists and the upper back to improve the recognition of boxing punches. The sliding window technique is employed to process sensor data and different window sizes are evaluated for optimal efficiency. Section 3.3 advances this work by integrating smartphone-based sensors and an AT-DCNN-BL model, which captures spatial and temporal dependencies to improve real-life boxing activity classification. The attention mechanism refines feature extraction, ensuring accurate recognition of complex punch variations. Finally, Section 3.4 introduces a real-time fatigue monitoring system using smartphone sensors. A CNN-LSTM hybrid model is deployed to detect fatigue based on variations **i**n punch force, speed and consistency. Attention-based LSTM further enhances the model's ability to track fatigue-related movement patterns. These methodologies collectively contribute to a more accurate,

efficient and accessible boxing performance monitoring system, optimizing athlete training, injury prevention and real-time feedback mechanisms.

3.2. Methodology for Multi-Sensor Fusion-Based Boxing Punch Activity Recognition

Boxing, known as the "noble art," has deep roots as a combat sport focused on stand-up fist fighting. Swaddling (2008) and Poliakoff (1987) note its inclusion in the 1904 St. Louis Olympics. Attwood (2006) and Dinu et al. (2020) emphasize that mastering proper techniques and posture significantly impacts an athlete's performance, highlighting the need for emerging technologies to enhance boxing punch analysis. Innovative technologies are transforming boxing training, offering valuable tools for coaches and athletes. Moeslund and Granum (2001), Tejero-de-Pablos et al. (2018) and Khan et al. (2020) classify these into image/video-based and sensor-based systems. Image/video-based technologies, used for decades, analyze motion through methods like gesture recognition and computer vision, providing insights into speed, angle and impact. However, Moeslund and Granum (2001) identify limitations such as high computational costs, real-time tracking difficulties and blind spots due to camera placement. To address these challenges, Dinu et al. (2020), Benages et al. (2019), Millot et al. (2023), Keskinoglu et al. (2023) and Nithya and Nallavan (2022) highlight the growing use of IMU sensors in sports like volleyball, cricket, boxing, tennis and running. Additionally, Hsu et al. (2018), Sha et al. (2020), Zhao et al. (2019) and Wang et al. (2019a) explore their applications in everyday tasks, demonstrating their accuracy and adaptability for motion analysis.

Bragança et al. (2020) and Baloch et al. (2018) outline that activity recognition using sensors typically involves four key phases: data acquisition, pre-processing, feature extraction and activity recognition. The data acquisition is done through IMU

sensors placed over the athletes body. While traditional segmentation methods have long been used in machine learning for pre-processing, their necessity in deep learning has been questioned. However, Marittha et al. (2021) suggest integrating segmentation with deep learning for activity recognition. For instance, Ebner et al. (2020) evaluated variable window sizes (2, 2.5 and 3 seconds) at a frequency of 50 Hz and observed a decline in accuracy with larger window sizes. Similarly, Banos et al. (2014) proposed a 1.2-second window as an optimal balance for speed and accuracy in HAR using conventional machine learning techniques. Research highlights the significant impact of window size on recognition speed and accuracy, motivating the integration of deep learning's automatic feature extraction capabilities with a sliding window approach. This research examines the impact of varying sliding window dimensions in a deep learning model, employing performance indicators like accuracy and the F1 score - a critical measure for evaluating models on unbalanced data distributions. Overlapping sliding window sizes analyzed include 200, 100, 50, 25, 20, 15, 10 and 5 frames, corresponding to sampling intervals of 2, 1, 0.05, 0.20, 0.15, 0.10 and 0.05 seconds at a frequency of 100 Hz. The primary goal is to determine the window size that minimizes latency and computational costs while maintaining high recognition performance.

Considering the feature extraction technique, Nweke et al. (2018) highlight that feature extraction is a critical process in HAR, as it identifies meaningful attributes from IMU sensor data, reduces classification errors and lowers computational complexity. Feature extraction methods are categorized into manual and automatic approaches. Ranao and Chao (2016) explain that manual extraction relies on domain expertise and is labor-intensive, whereas automatic extraction uses deep learning algorithms to autonomously identify patterns and features. Pajak et al.

(2022), Mim et al. (2023), Khatun et al. (2022) and Khatun et al. (2023) emphasize that deep learning methods, especially DCNN, are highly effective in capturing local dependencies within data across multiple domains. DCNNs, in particular, excel in recognizing complex activities and are highly effective in time series classification tasks, making them ideal for robust HAR systems, as noted by Nweke et al. (2018), Almaslukh et al. (2018), Ignatov (2018) and Wang et al. (2019b).

Considering the activity recognition, Liu et al. (2018), Yan et al. (2014) and Connaghan et al. (2011) have made progress using single-sensor data, but Keskinog˘lu et al. (2023) point out that using a single sensor for activity recognition in sports poses significant challenges due to the complexity of movements. For instance, distinguishing between similar boxing punches, such as jabs and hooks, is difficult due to subtle differences, especially during the latter stages of a flexed arm movement. Moreover, individual variability in executing the same punch type complicates the recognition process. The inherent limitations of single sensors, such as restricted spatial coverage, occlusion, imprecision and uncertainty, limit their reliability.

Khan et al. (2020) propose the use of multiple sensors positioned at various body locations to overcome these challenges and improve performance parameters such as precision, accuracy, recall, sensitivity, specificity and F1 score. The multi-sensor approach leverages complementary data from all the sensors to address the shortcomings of individual sensors, thereby enhancing overall recognition performance.

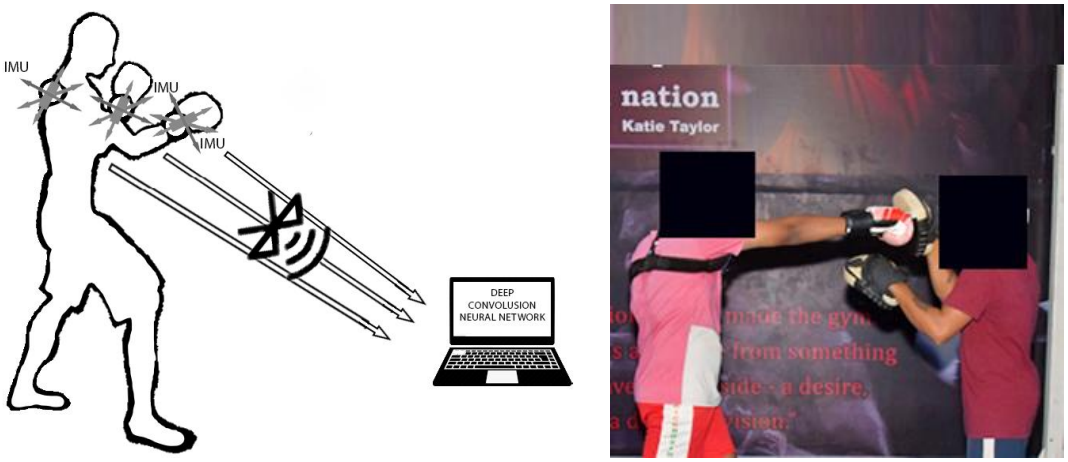
This section explains how to tackle the challenges in activity recognition and classification by emphasizing the importance of multi-sensor systems in achieving high accuracy and F1 scores. The contributions of this study are as follows:

1. **Introduction of a Novel DCNN Classification Model**: A deep learning-based DCNN classification model is proposed, specifically designed for real-time applications to accurately classify various boxing punches.
2. **Data Fusion Using DCNN Architecture**: The study employs a data fusion approach using DCNN, combining data from sensors located at three different body positions. This method improves the model's ability to differentiate between activities with similar patterns.
3. **Exploration of Optimal Window Size**: The research investigates the relationship between sliding window size and recognition accuracy. By analyzing various window sizes, the study provides insights into the smallest effective window size for detecting boxing punches.

In summary, the proposed contributions advance boxing activity recognition by introducing an innovative deep learning model, leveraging data fusion techniques and optimizing window sizes for improved real-time punch recognition accuracy.

3.2.1. The Proposed System of Measurement

Using multiple sensors for activity detection provides several advantages over single-sensor systems. Uddin et al. (2020) explain that these advantages include improved noise reduction, reduced ambiguity and the ability to integrate data from multiple input signals. This study introduces an innovative data fusion methodology designed to leverage these benefits. The proposed approach extracts distinct features from three individual IMU sensors, strategically placed at different positions on the player’s body that is on both wrists and the upper back (Figure 3.1 (a) & (b)).



(a) (b)

Figure 3.1 IMU sensors placed on both wrists and the upper back

Each sensor's data is independently processed through convolutional layers to extract features specific to its location and characteristics. The extracted features from all three sensors are then combined at the classification layer, as illustrated in Figure 3.2. This data fusion technique significantly enhances the overall performance of the activity detection system, demonstrating the effectiveness of leveraging multiple sensors for accurate and robust activity recognition.

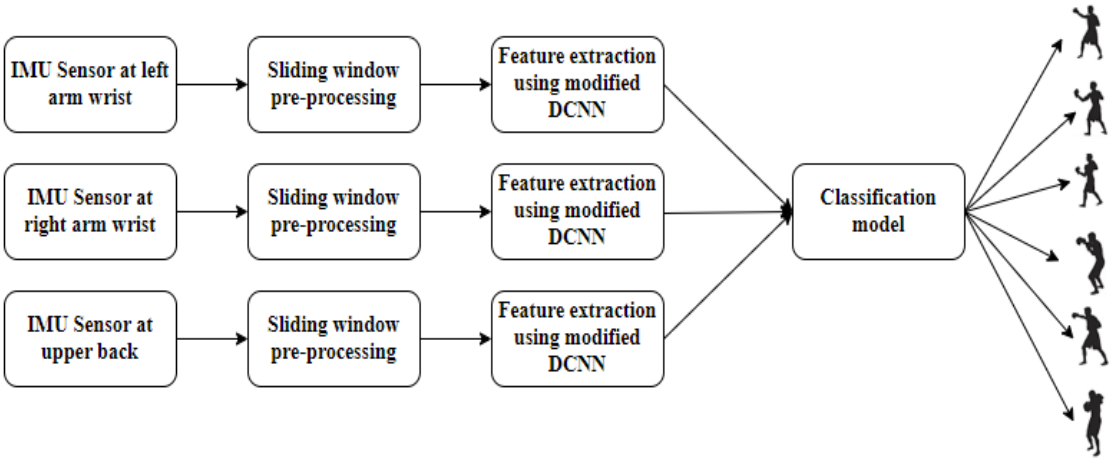


Figure 3.2 The system architecture

For the measurements, athletes were equipped with sensors placed on both wrists beneath their boxing gloves, along with a third sensor positioned on the upper

back using a clavicle belt. This arrangement ensured accurate data collection while minimizing interference with the athletes' movements.

The study commenced with a 10-minute warmup session, after which each participant performed six predefined punch types. These punches were recorded over a total duration of 10 hours. The study involved ten players with varying skill levels to ensure diverse data representation. Initially, the players executed all six punch types in a random sequence, followed by successive sets of punches with a 30-second rest period between each set.

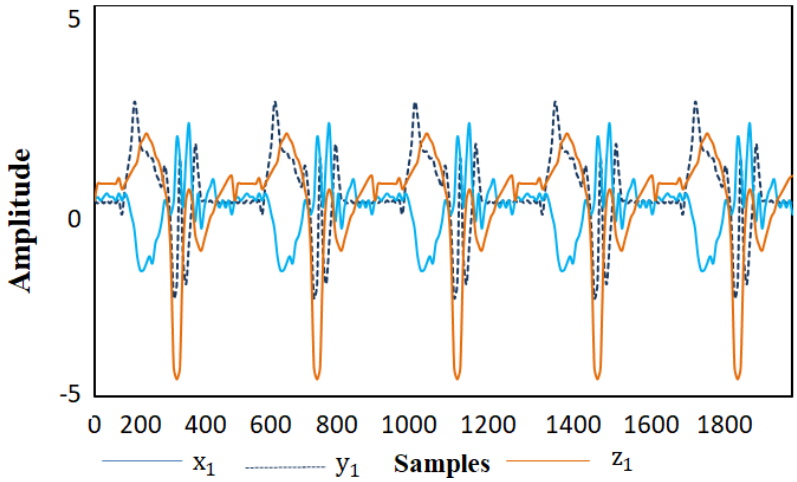


Figure 3.3 IMU signals representation obtained from a participant

The design of the wearable devices, including the armbands and the clavicle belt, was carefully optimized to provide a secure fit without restricting the athletes' natural range of motion. Figure 3.3 illustrates the signals recorded during the execution of a jab punch data collected in the study.

3.2.2. Hardware Platform

The SparkFun Razor 9 Degrees of Freedom (DOF) sensor, shown in Figure 3.4, is a preconfigured device designed for seamless IMU data logging. Each SparkFun Razor 9 DOF unit features an IMU that captures data from a 9DOF accelerometer, gyroscope and magnetometer. The recorded data can be stored on a

microSD card or wirelessly transmitted to a PC, enabling efficient computational analysis and processing.

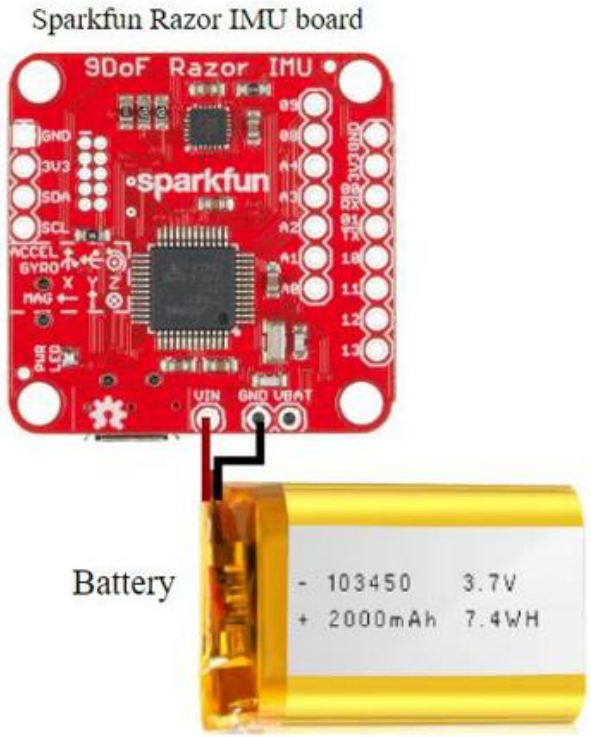


Figure 3.4 The wearable sensor including 9DOF IMU

The sensor provides a wide range of options for both acceleration and angular rate measurements. It offers selectable acceleration ranges of ±2/±4/±8/±16 g and angular rate ranges of ±125/±250/±500/±1000/±2000/±4000 degrees per second (dps). The integrated magnetometer is sensitive enough to detect magnetic fields as small as 0.4 mg, with an accuracy of ±0.5°. The key to utilizing this data effectively lies in interpreting it to extract meaningful insights. Consequently, the signal transmission section requires a significant amount of power to remain active during the entire data transmission process, which extends from the sensor to the deep learning layers for processing and inference.

To assess the feasibility of real-time sensor use, factors such as energy consumption, size limitations and weight constraints must be considered. Therefore, a low-capacity Li-ion battery (3.7V, 2000 mAh) has been integrated into the system.

Kautz et al. (2017) and Khan A et al. (2016) state that for accurate analysis of sports motion activity features, a high sampling rate of 100 samples per second is required. However, Hooshmand et al. (2017) explain that when analyzing the expected battery life of a 9 DOF wearable device with a 2000mAh battery and sampling data at 100 Hz, the battery would last no longer than 10 hours. This limitation poses a challenge for real-time applications, even with the use of compression algorithms, as noted by Bassoli et al. (2017) though these algorithms can significantly extend battery life. But Bisio et al. (2016) highlight that leveraging the on-board processing capabilities of the device allows data to be stored in burst sequences, utilizing the internal memory's buffering capacity, which can potentially extend the battery life to around two days.

3.2.3. Selection of Neural Network Model

The emergence of deep learning has brought about a transformative shift in the signal processing environment and feature extraction techniques. Prior to the advent of deep learning, the conventional approach involved manual feature extraction, which entailed generating domain-specific attributes, as discussed by Ma et al. (2020). These features were then used to train machine learning models on the processed data. However, this method has certain limitations. It demands the processing of acquired data signals and necessitates domain experts labeling the data for both data collection and analysis of raw data. Moreover, model-fitting features are required for each new dataset, further complicating the process.

The classical machine learning approach has many additional drawbacks. These include the challenge of developing a generalized model capable of accommodating diverse movements performed by distinct individuals. With the objective of justifying the computational complexity inherent in deep learning and recognizing the potential of deploying deep learning models on affordable, portable, wearable devices for automatic feature extraction, the proposed research endeavour centres on the development and training of a neural network. The neural network is devised to conduct predictions while conserving resources to the fullest extent, thus ensuring its portability on low-cost embedded devices.

In this context, the study introduces a streamlined deep learning network personalized to recognize boxing punches. Notably, the research establishes the efficacy of deep learning in SAR compared to conventional neural network techniques, an assertion verified by Khan et al. (2020). The basic aim of this study is to offer an efficient and automated means of recognizing boxing punches using deep learning technology, facilitating real-time recognition while keeping computational demands in check.

3.2.4. Dataset and Sliding Window Creation

The dataset used in the experiments was collected from three 9 DOF IMU sensors, with two sensors placed on each athlete's wrists and one on the upper back. The deep learning model was evaluated by dividing the data into training and testing sets. The training set consisted of raw data from eight athletes, while data from two athletes was reserved for testing. The IMU dataset contained 261,790 samples for testing and 79,361 samples for validation.

There are two main approaches for creating sliding windows: one method involves dividing sequences into frames, while the other segments the time following

the data sequence into fixed intervals. These windows can be either overlapping or non-overlapping and fixed or adaptive. Fixed windows maintain the same size throughout the sequence while adaptive windows adjust based on specific criteria related to movement. Overlapping windows occur when the subsequent window includes part of the previous sequence, or when there is overlap between adjacent windows, as outlined by Ma et al. (2020) and Dehghani et al. (2019a).

In this study, 9 DOF IMU sensor signals were recorded, with peaks in acceleration corresponding to punch impacts. To identify the acceleration peaks associated with punch impacts and mitigate noise and motion artifacts caused by individual and environmental factors, preprocessing steps involved applying low-pass and high-pass filters. To address potential data dropouts and insufficient information in frames shorter than two seconds, cubic spline interpolation was used to resample the data. The resampled values were subsequently normalized to fall within a spectrum ranging from -1 to 1.

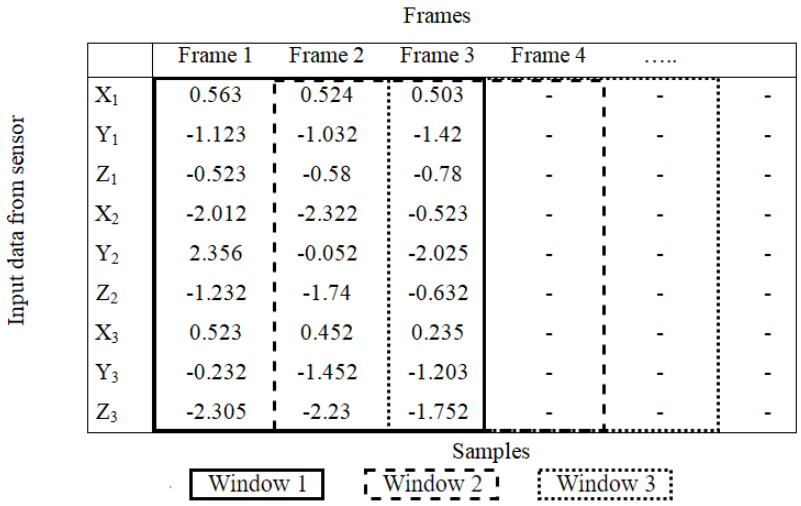


Figure 3.5 Sliding window schematic

A sampling frequency of 100 Hz was used, which resulted in 100 samples of X, Y and Z acceleration values per second. The signal was divided into two-second sliding windows with a 50% overlap between consecutive windows. This window size

and overlap ratio, inspired by work of Ronald et al. (2021), provide a balance between capturing relevant activities and minimizing time delays, making the approach suitable for real-time applications.

Understanding the components of a sliding window is essential. A sequence represents a larger sample that may contain one or more individual samples. Each observation within the sequence corresponds to a time step, forming the "window size." A feature in this context refers to the input of the sequence, such as X1, Y1, Z1, X2, Y2, Z2, X3, Y3 and Z3, which represent a single time interval (Figure 3.5). As the window moves through the sample, the next phase begins.

For the smallest sliding window size of five, the resulting input matrix is quite large, specifically (261785, 5, 9), as shown in Table 3.1. This matrix is created by dividing the total number of samples by the number of time steps, incorporating all the observed data columns. As the window size increases, the total number of samples decreases, which corresponds to the reduction in the overall count due to the subtraction of the window size.

Table 3.1: IMU window sizes distribution for training the model with 261790 samples

Window size			
5	**10**	**15**	**20**
(261785, 5, 9)	(261780, 10, 9)	(261775, 15, 9)	(261770, 20, 9)
Window size			
25	**50**	**100**	**200**
(261765, 25, 9)	(261740, 50, 9)	(261680, 100, 9)	(261680, 200, 9)

This study employs fixed overlapping sliding windows, where all available samples are divided into windows of size "n." The dataset was collected at a constant sampling rate of 100 Hz, with an initial window size of 100 frames, which corresponds to a duration of one second. To identify the optimal window configuration, the study further investigates smaller window sizes, experimenting with lengths of 20, 15, 10 and 5 frames. These correspond to time intervals of 0.20, 0.15, 0.10 and 0.05 seconds, respectively.

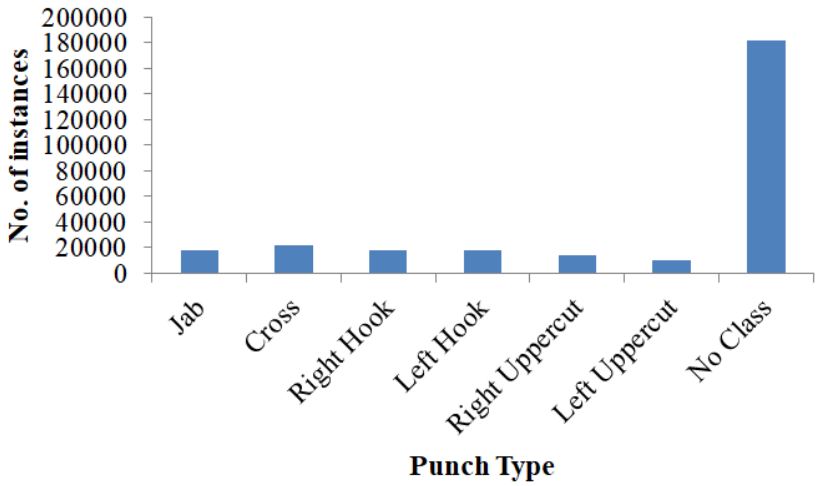


Figure 3.6 Composition of data (261,790 Instances)

The decision to use a shorter sliding window is intentional, as it enables real-time data processing, which aligns with the goal of embedding the inference process within the sensor node. This strategy enhances the model's ability to process data in real-time by incorporating the inference step directly into the sensor node. As a result, the study generates a dataset containing 261,790 instances, with a comparison of these instances to their respective activities presented in Figure 3.6.

3.2.5. Spatial Feature Extraction

Deep learning models leverage their intrinsic ability to automatically extract features directly from raw data, which is a key advantage for the proposed research methodology in recognizing boxing punches. This automatic feature extraction

capability within deep learning frameworks is central to the approach used in this study.

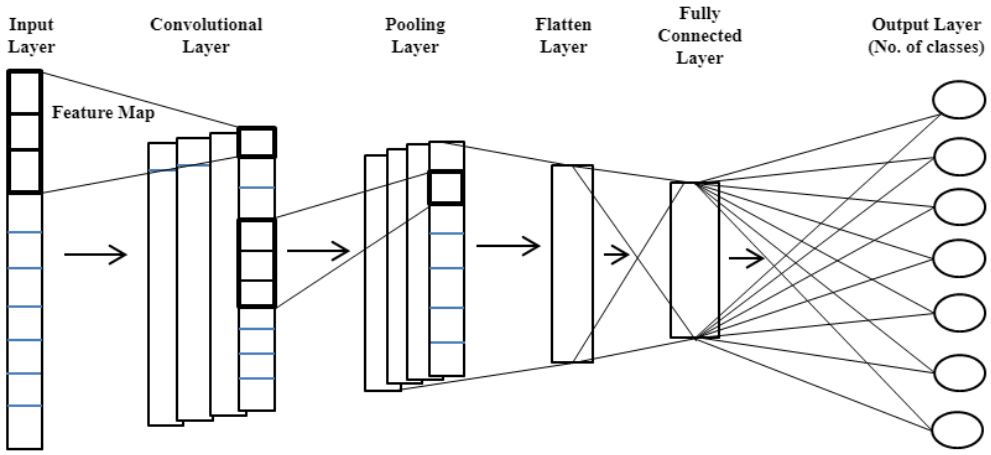


Figure 3.7: 1D convolutional architecture for spatial feature extraction

Among deep learning techniques, DCNN are particularly noteworthy for their ability to both extract features and perform classification tasks on input activity data, as highlighted by Khan et al. (2020). The convolution operation involves sliding a one-dimensional filter across the sensor data signal, a process shown in Figure 3.7. Ihianle et al. (2020) and Namatēvs (2017) explain that this DCNN architecture efficiently extracts unique features from time series data, particularly excelling at processing shorter segments within a larger dataset.

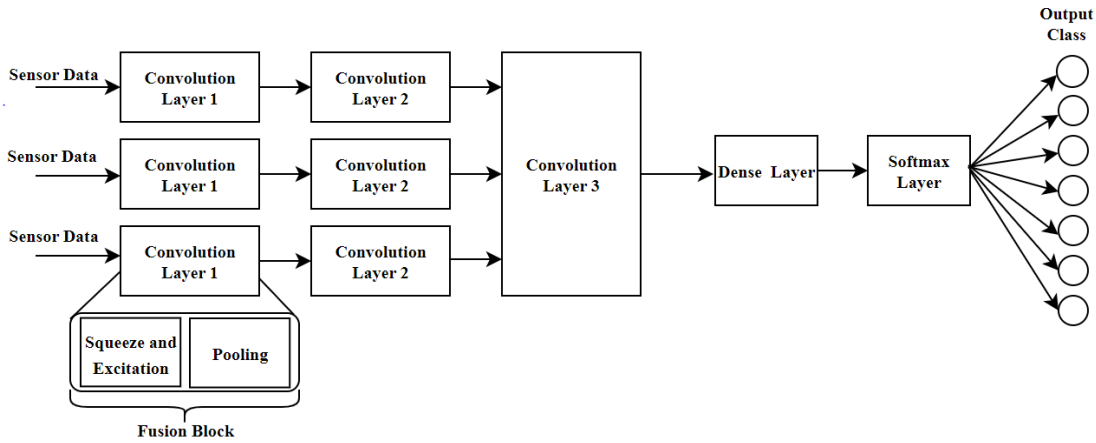


Figure 3.8 Architecture of the optimized DCNN model for boxing punch recognition

In time series data, DCNNs are essential for capturing local dependencies by correlating adjacent signals, a key feature utilized in this study for extracting local characteristics. The Rectifier Linear Unit (ReLU) activation function is used to address the non-linearity introduced during the convolutional operation. For dimensionality reduction and to decrease computational load, the pooling layer uses the max-pooling method, which has demonstrated better performance than average pooling. As a multilayer architecture, as shown in Figure 3.8, the DCNN processes the input data through several stages, including fully connected layers and a softmax layer, after the convolution and pooling operations. Bergstra et al. (2011) explain that during training, hyperparameters are adjusted to create the link between the input IMU sensor data and the output activity classes.

3.2.6. Optimisation of DCNN Model and Data Fusion Approach

Due to their impressive performance, DCNN are frequently used for feature extraction. The optimization of the model follows the methodologies recommended by Castanedo (2013). Zulkifli (2018) explains that data fusion involves integrating information from multiple sources, which becomes crucial in situations involving complex behaviors. A single sensor often proves inadequate, particularly for capturing diverse actions, thus requiring sensors to be placed at multiple body locations. To overcome this limitation, the present study proposes a sensor fusion approach, where three sensors; placed on each wrist and one on the upper back, contribute their respective x, y and z coordinates to separate convolutional layers for feature extraction.

Since features from each sensor are extracted independently, the sensor fusion technique results in a higher number of trainable parameters compared to a standard DCNN architecture without fusion. As detailed in the results section, this approach

leads to improved accuracy in recognizing related activities, such as distinguishing between a hook and a jab. In the context of boxing punch recognition, this study employs a deep classification architecture, namely DCNN, enhanced with sensor fusion and sliding window techniques. A reference DCNN model, without optimization, is used for comparison to evaluate the outcomes. The hyperparameters for each configuration were selected through an iterative process, testing different value ranges using a trial-and-error approach. To examine their effect on recognition accuracy, various numbers of convolutional layers were tested.

Table 3.2: Comparison of DCNN accuracy for different fully connected layers

Convolution Layer Count	Fully Connected Layer Count	Accuracy
2	1	92.80%
3	1	94.20%
3	2	93.80%
4	1	93.60%
4	2	93.20%

The configuration that achieved the highest accuracy included three convolutional layers and one fully connected layer, as detailed in Table 3.2. Initially, increasing the number of convolutional layers improved recognition performance; however, this improvement got halted after three layers. Beyond this point, performance began to decline, likely due to the model's tendency to memorize the training data, a phenomenon known as overfitting. Pigou et al. (2018) explain that while such models may perform well on training data, their accuracy tends to drop when tested on new, unseen data. To address this issue of overfitting, a dropout

technique is implemented, where neurons are randomly deactivated during training. This process prevents these neurons from participating in the forward pass and temporarily isolates them, ensuring that their weights are not updated during the backward pass. In this study, Pigou et al. (2018) applied a dropout rate of 0.2 (equivalent to 20% dropout) to reduce overfitting effectively.

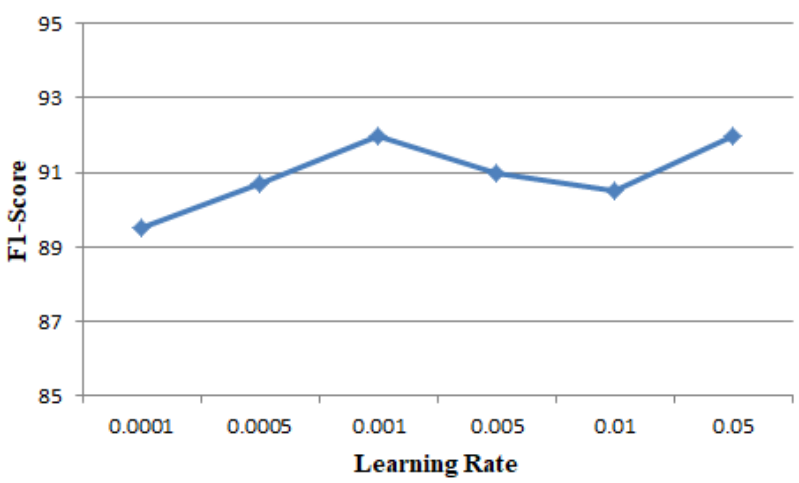


Figure 3.9: Figure showing learning rate affecting F1 score

Hamad et al. (2019) emphasize the critical role that the learning rate plays in optimizing network weights during gradient descent. They explain that smaller learning rates lead to slower convergence, while larger learning rates can result in issues such as non-convergence or instability in training. Getting the reference from the study in the proposed method, the learning rate is selected from a range between 0.0001 and 0.01, keeping all other hyperparameters constant. The impact of the learning rate on the window size is illustrated in Figure 3.9 showing that a learning rate of 0.001, combined with a batch size of 64 and trained for 50 epochs, maintained a high F1 score.

However, the smaller batch size aids in quicker model convergence. Among the tested kernel sizes (3, 4, 5 and 7), it was observed in the study that a kernel size of 3 provided the best performance and was selected for the final experiments. The

results of the hyperparameter configuration that had and their positive impact on performance are summarized in Table 3.3.

Table 3.3 Hyperparameter settings for the model

Hyperparameter	Experimental values	Selected values
CNN layers	1-6	3
Kernel size	3, 5, 7	3
Feature maps	256, 128, 64, 32	128, 64, 64
Pooling size	2, 3, 4	2
Dropout	0.2, 0.3, 0.4, 0.5	0.2
Optimizer	RMS Prop, Adam	Adam
Learning rate	0.0001 to 0.01	0.001
Batch size	32, 64, 128, 256	64

The objective of the network is to identify unique features within data sequences and classify them into predefined boxing punch categories. In our proposed approach, a sequential architecture is employed, consisting of several layers, including convolutional, max pooling, dense and dropout layers. These layers work together to extract distinctive features from the input data.

The proposed DCNN fusion model includes three convolutional layers with kernel sizes of three and 128, 64 and 64 filters, respectively. Following the notation used by Srivastava et al. (2014), the network can be represented as C(128) – C(64) – C(64) – D(256) – Sm. In this representation, C(Fl) refers to the number of feature maps in convolutional layer F1, D(n) indicates the number of units in dense layer n and Sm represents the softmax output layer.

For the output layer, a softmax activation function is applied, which is defined by the following equation:

$$q_{i=\frac{e^{k_i}}{\sum_{a=1}^{n} e k_a}} \quad (3.1)$$

Equation 3.1 defines the softmax function, which computes the probability q_i by normalizing the sum of all exponentials from the elements k_i of the input vector. This function transforms the network's raw outputs into a probability distribution.

ReLU activation functions are used throughout all layers, including the output layer. The choice of these activation functions is integral to the overall model structure. The specific hyperparameters for the proposed model are outlined in Table 3.3. The model was implemented using Keras with a TensorFlow backend. Testing and training were conducted on hardware equipped with an Intel Xeon E-2224G processor (3.5 GHz) and 32 GB of RAM. This research provides the outcomes of a deep learning model (DCNN) that utilizes an optimal sliding window configuration for identifying boxing punches involving similar movements. The DCNN mode was evaluated using sensor data fusion and variations in the sliding window approach.

3.2.7. Performance Metrics and Evaluation

Sparse categorical cross-entropy was selected as the loss function to assess the performance of the proposed model. Given the multi-class nature of the network's output, the study included seven distinct activity classes: cross, jab, left hook, right hook, left uppercut, right uppercut and no hit. The ReLU activation function was used throughout and the Adam optimization algorithm, a stochastic gradient-based method, was employed for training.

Performance Metrics: The evaluation of the enhanced DCNN model, which incorporated sensor data fusion, was conducted using two primary metrics: accuracy and the F1 score. Accuracy measures the proportion of correct predictions to the total number of samples. Yan (2109) explain that while accuracy is commonly used when there is no class imbalance, this study acknowledged that the frequency of 'jab' and

'cross' motions was higher compared to 'hook' or 'uppercut' motions, creating an imbalance among the classes. As a result, the F1 score was also included, as it is sensitive to class distribution and is less affected by class imbalance. The F1 score is a combination of precision and recall, providing a more balanced evaluation in such cases. Higher accuracy or F1 scores indicate better model performance (He et al. 2009), with the F1 score also accounting for the proportion of samples in each class.

Additionally, the evaluation process considered the model's inference time, which refers to the time taken to generate a response. Inference time is crucial for real-time applications like SAR where rapid decision-making and immediate responses are essential. The performance comparison parameters included accuracy, precision, recall and the F1 score. While accuracy evaluated the overall performance of the DCNN across all class outcomes, precision, recall and F1 score were used to assess how well the model recognized each individual class. The formulas for calculating these evaluation metrics are provided below.

$$\text{ClassificationAccuracy} = \left(\frac{\text{true positive} + \text{true negative}}{\text{true positive} + \text{true negative} + \text{false positive} + \text{false negative}} \right) \tag{3.2}$$

$$\text{Precision} = \frac{\text{true positive}}{\text{true positive} + \text{false positive}} \tag{3.3}$$

$$\text{F1-score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{3.4}$$

$$\text{Recall} = \frac{\text{true positive}}{\text{true positive} + \text{false negative}} \tag{3.5}$$

Evaluation: Evaluation is a crucial step in the design of any system and in the context of HAR, it is traditionally performed using k-fold Cross-Validation (CV). Hastie et al. (2009) explain that in k-fold CV, the entire dataset is randomly divided into k equal

subsets. The model is trained on k-1 of these subsets, while the remaining subset is used for testing. It is important to note that the test set in this method may include data from the same subjects as those in the training set, which is commonly referred to as subject-dependent CV in the literature.

Arlot & Celisse (2010) describe the standard assumption in CV as the data being independent and identically distributed, meaning that each data point is assumed to come from the same distribution independently. However, this assumption may not hold when dealing with data from human subjects for two key reasons. Bulling et al. (2014) highlight the first reason: significant variability between subjects in how they perform activities. This means that data samples from the same subject are likely to be more similar to each other than those from different subjects. Factors such as age, gender and experience can influence this variability. The second reason is temporal dependence in activities performed by the same individual. For example, data samples collected within the same training session are likely to exhibit more similarity due to factors like fatigue and training patterns compared to samples collected at different times. These issues suggest that k-fold CV may overestimate the performance of HAR systems.

Dehgani et al. (2019b) discuss that this overestimation is even more pronounced when k-fold CV is used with overlapping sliding windows, as the overlap between adjacent windows introduces additional dependencies among the data points. To address these concerns, Janidarmian et al. (2017) and Al Machot et al. (2019) propose an alternative approach called subject-independent cross-validation (subject-independent CV). In this method, the data are divided by subject. In each iteration, the model is trained on data from all but two subjects, with the remaining two serving as the test set. This approach effectively eliminates the intra-subject dependencies seen

in subject-dependent CV, providing a more accurate evaluation of model performance.

3.3. Methodology for Smartphone Based Boxing Activity Recognition

Mobile computing technologies have played a critical role in bridging the gap between athletes and sports monitoring systems, enabling the development of intelligent and automated solutions for monitoring individual sports activities. The proliferation of smartphones equipped with multifunctional sensors has made it possible to acquire and analyze data for human activity recognition. While accelerometers in smartphones are widely used for recognizing sports activities, traditional methods struggle with the complexity and real-time demands of intricate activities due to the high-dimensional nature of sensor data.

This study proposes a smartphone-based architecture for HAR, utilizing inertial accelerometers to record sequences of an athlete's game actions, extract relevant features and derive activity data through multiple three-axis accelerometers. Specifically, an Attention-Assisted Deep Convolution Neural Network with Bi-directional Long short term memory model is introduced. In this approach, raw data undergoes pre-processing using a sliding window technique. DCNN layers are employed for automatic feature extraction, while the Bi-LSTM structure processes the boxing activity data. An attention mechanism is incorporated to focus on key features within the boxing activity classification data, redistributing feature weights to improve classification accuracy.

Other classification models, including DCNN, Bi-LSTM, Multi-Layer Perceptron Neural Networks (MLPNN) and Random Forest (RF), were also applied to the dataset collected via smartphone sensors. The study explores the training of these deep learning methods and demonstrates the superiority of the AT-DCNN-BL model,

which achieved the highest accuracy rate of 92.71%, outperforming all other models tested on the mobile sensor dataset.

This study introduces an AT-DCNN-BL model for boxing activity recognition, specifically addressing challenges like noisy data, temporal dependencies and computational efficiency. The model integrates cross-domain knowledge to distinguish variations within similar activities and applies hybrid methods for enhanced feature extraction and classification. The contribution of this proposed work are**:**

1. Proposing an AT-DCNN-BL model to extract sports activity features and distinguish variations in similar activities across different contexts.

2. Experimentally analyzing the performance of various classification models, including DCNN, Bi-LSTM, MLPNN and RF, on boxing datasets acquired via smartphone sensors. The results demonstrate the superior performance of the AT-DCNN-BL model.

Various investigations have examined activity identification utilizing sensor data in conjunction with machine learning methodologies. Wang et al. (2019a) highlighted the potency of DCNN frameworks in extracting features pertinent to activity recognition. Ronao & Chao (2016) put forth a DCNN-based model aimed at improving human activity classification via smartphone sensor readings, leveraging automated extraction of features from time-series one-dimensional signals. Ignatov (2018) introduced an architectural design utilizing DCNNs that incorporated both statistical and dynamic attributes, achieving superior accuracy compared to foundational models.

Alternative methodology proposed by D'Angelo & Palmieri (2021) involve transforming accelerometer recordings into HAR-Images for processing with DCNN

frameworks. Another approach by Gholamrezaii & Almodarresi (2021) consists of convolutional layers exclusively to minimize computation time, while Khan & Ahmad (2021) proposed that attention-driven multi-head models have also been employed for HAR. Though these models excel at identifying spatial-location-related characteristics, they struggle with effectively capturing temporal aspects.

To tackle the issue of temporal feature extraction, some researchers devised a two-phase methodology integrating spectral transformation for converting data into a time-frequency format before employing DCNN-based classifiers as proposed by Amer & Ji (2021). Additionally, Ai et al. (2020), Vaswani et al. (2017) and Hu et al. (2024) describes that LSTM networks have been instrumental in recognizing sequences with long-term dependencies, benefiting the inference of extended-duration human behaviors. However, Wang et al. (2018) describes that LSTM-based architectures require substantial processing time due to their sequential handling of continuous sensor inputs, rendering them slower compared to parallelized techniques. Moreover, when exposed to noisy datasets, LSTM frameworks often exhibit diminished classification accuracy.

Table 3.4 Sensors for activity recognition found in popular smart mobile phones

Sensor	Samsung Galaxy S23	iPhone 13	One Plus 11
Accelerometer	✓	✓	✓
Gyroscope	✓	✓	✓
Light	✓	✓	✓
Proximity	✓	✓	✓
Barometer	✓	✓	✓
GPS	✓	✓	✓

This study introduces an innovative AT-DCNN-BL architecture tailored for boxing activity classification, designed specifically to mitigate issues related to time efficiency and performance deterioration in noisy environments. Table 3.4 presents a summary of frequently integrated sensors in widely used smartphone models. The discussion narrows its scope to motion-sensing technologies, notably accelerometers and gyroscopes. When utilizing smartphones for human activity detection, particularly within boxing contexts, several unique difficulties emerge, differing from wearable sensor-based implementations. The primary obstacle involves varying orientations and placements of smartphones during operation. Additionally, the intricate movements inherent to human activities contribute to sensor data often being intermixed with noise.

To counteract these challenges in boxing activity detection, research efforts have concentrated on optimizing methodological feasibility by prioritizing noise mitigation, feature selection and classification advancements.Wang et al. (2019) and Ronao & Chao (2016) emphasize the role of DCNN-based models for feature extraction and classification using smartphone sensor data. Ignatov (2018) and others, such as D'Angelo & Palmieri (2021), Gholamrezaii & Almodarresi (2021) and Khan & Ahmad (2021), have investigated models utilizing HAR-images, convolution-only architectures and attention-based multi-head models for SAR. However, these approaches faced challenges in extracting temporal features effectively.

To address these limitations, Amer & Ji (2021) incorporated spectral analysis and DCNNs for classification, while Ai et al. (2020), Vaswani et al. (2017) and Hu et al. (2024) employed LSTMs for long-term sequence recognition. However, LSTM-based models are computationally expensive for sequential data processing and

struggle with noisy data. The AT-DCNN-BL model proposed in this study addresses these issues, offering robust performance in boxing activity recognition.

Table 3.4 presents a list of sensors commonly found in popular smartphones, with a focus on motion sensors such as accelerometers and gyroscopes. Recognizing human activities using smartphones, particularly in the context of boxing, poses unique challenges compared to wearable sensors. One major challenge lies in the varied placement and orientation of smartphones during use. Additionally, the complex nature of human activities often results in sensor data being contaminated with noise. To address these challenges in boxing activity recognition, research has emphasized enhancing methods for noise reduction, feature extraction and classification.

3.3.1. Experimental Methods

This study explores a range of classification techniques, spanning from conventional machine learning models to advanced deep learning architectures. Among these, the proposed AT-DCNN-BL model demonstrates superior performance when applied to data collected via smartphone sensors. The evaluation metrics employed to evaluate the proposed models are discussed in detail in the results section. The findings emphasize the effectiveness of attention weight adjustment techniques, particularly in handling noisy data. The following sections provide detailed descriptions of the models employed in this study.

Random Forest (RF) Model

The Random Forest (RF) model builds upon the decision tree algorithm to enhance classification performance. A decision tree consists of nodes, including a root node at the top, internal nodes and leaf nodes. Root and internal nodes use conditional statements based on classification features, with each feature evaluated independently

for its effectiveness. Common metrics, such as Gini Impurity and Entropy, are used to select features. For Gini, the feature with the lowest Gini score, indicating the best class separation, is chosen as the root node. The procedure progresses through every internal node, systematically choosing attributes with minimal Gini indices until either all attributes are exhausted or a Gini index of zero is reached, at which point the terminal nodes become leaf nodes and classify the feature vector.

The Random Forest model improves upon this by generating *N* bootstrapped feature subsets from the original dataset. Each subset is used to train a separate decision tree, resulting in *N* distinct models. The final prediction is determined by majority voting across these *N* trees, enhancing robustness and accuracy.

Multi-Layer Perceptron Neural Network Model

The MLPNN is a feedforward neural network designed for classification and prediction tasks. It consists of an input layer, one or more hidden layers and an output layer. Each neuron within the network applies an activation function and uses adjustable weights to influence its output.

In this study, the MLPNN architecture comprises an input layer, two hidden layers with 256 neurons each and an output layer. Training is performed using the backpropagation technique, which optimizes the network by adjusting weights to minimize the cost function, typically via gradient descent. The hidden layers act as independent classifiers, enabling the network to handle increasingly complex classification problems.

DCNN Model

The DCNN is a deep learning architecture composed of layers such as convolutional layers, pooling layers and fully connected layers. As depicted in Figure

3.10, its structure includes an input layer, multiple convolutional and pooling layers, followed by a fully connected layer and an output layer.

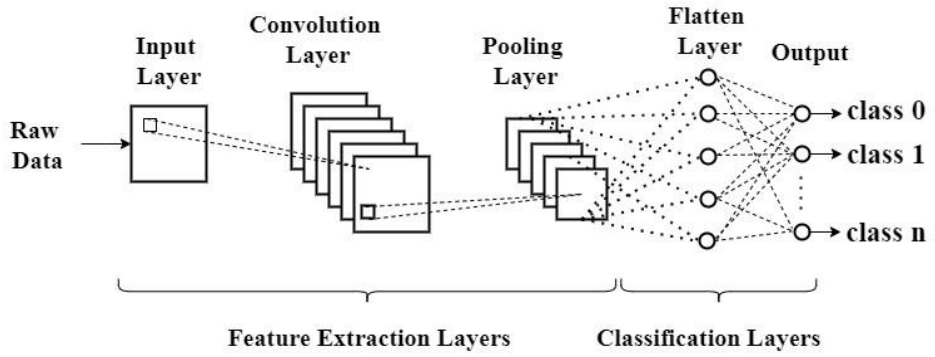


Figure 3.10 Architecture of DCNN

The input layer processes time-series data and convolutional layers extract features, with each successive layer capturing increasingly complex patterns. Pooling layers, typically using max-pooling, reduce the dimensionality of features by extracting the maximum value from smaller blocks of the feature map. After passing through multiple convolutional and pooling layers, the data is transformed into a 1-D vector, which is then processed by fully connected layers to capture nonlinear relationships. The final classification is performed in the output layer using the softmax function, which generates probability distributions for the possible classes.

LSTM Model

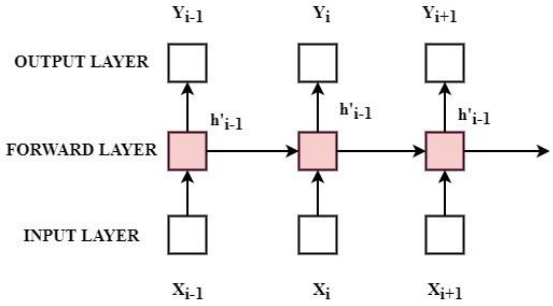


Figure 3.11 LSTM model layers

The LSTM model is a specialized type of Recurrent Neural Network (RNN) designed for analyzing time-series data. The key innovation of LSTM is its cell state,

which helps retain and process information across sequences. LSTM employs gates that is composed of sigmoid layers and pointwise multiplication, to regulate the flow of information dynamically, addressing challenges like vanishing or exploding gradients.

LSTM models are widely applied in fields such as language translation, speech recognition and image analysis. Figure 3.11 illustrates the layers within an LSTM model, highlighting their functionality in retaining long-term dependencies.

Bi-LSTM Model

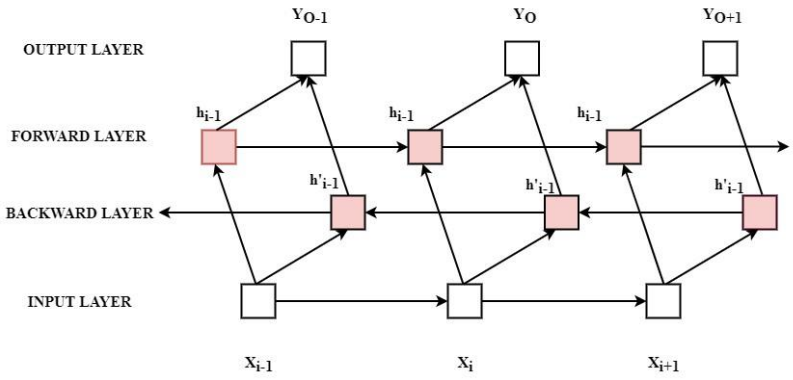


Figure 3.12 Bi-LSTM model layers

The Bi-LSTM model builds upon the traditional LSTM by incorporating bidirectional processing. While standard LSTM processes sequences in a single direction (based on past data), Bi-LSTM incorporates both forward and backward computations. This dual approach uses two RNN structures: a forward RNN, which relies on past data and a reverse RNN, which incorporates future data. By simultaneously considering information from both past and future time steps, Bi-LSTM is particularly effective for data with strong bidirectional dependencies. This capability makes it superior to unidirectional LSTM models in predictive accuracy, especially for tasks requiring detailed contextual understanding. Figure 3.12 illustrates the structure of the Bi-LSTM model.

3.3.2. The Proposed System

This section introduces the Attention-Assisted Deep Convolutional Neural Network with bi-LSTM (AT-DCNN-BL) model. Figure 3.13 illustrates the data flow in the proposed system. The raw sensor data is first segmented using a sliding window technique, serving as a pre-processing step. This segmented data is then passed through the DCNN layer to extract features. The activity data, obtained from a single channel, is processed through a DCNN-based Bi-LSTM architecture. The Bi-LSTM network extracts meaningful features from the noisy data using forward and backward propagation across its hidden states. Following this, an attention mechanism refines the extracted features, selecting the most critical ones and redistributing their weights. A fully connected layer then classifies the activity classes and a dropout layer is employed to mitigate overfitting.

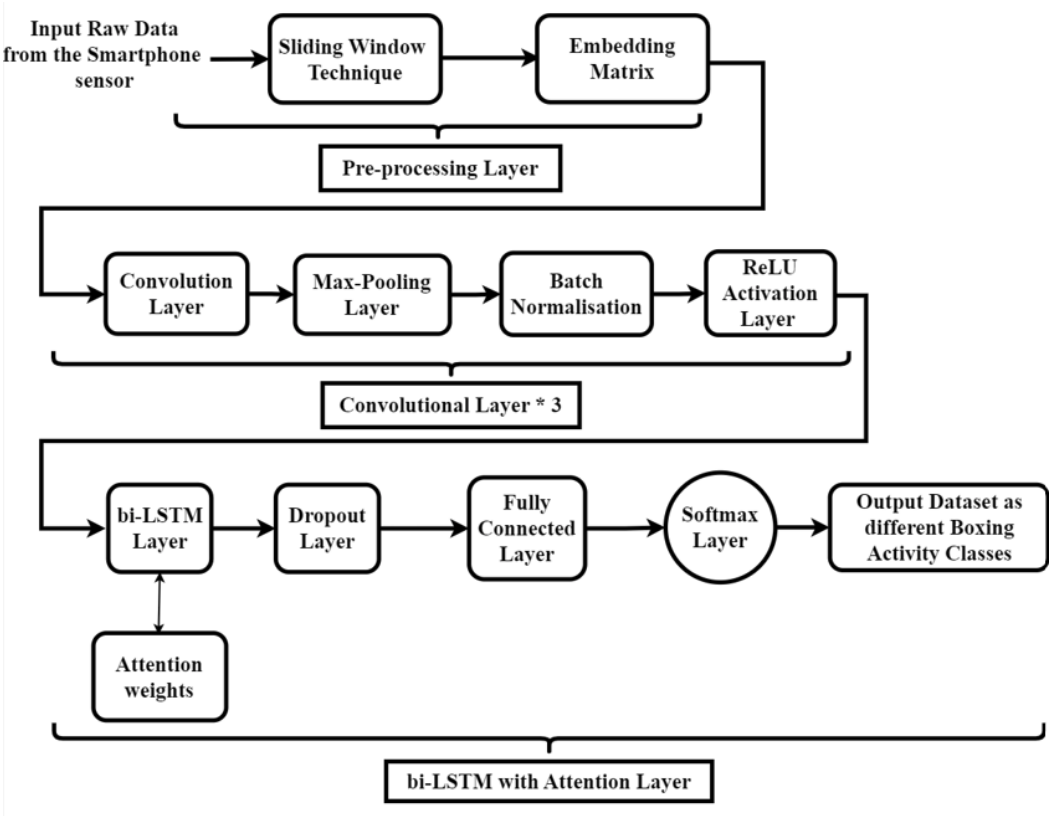


Figure 3.13 The data flow diagram of the proposed system

Contribution of Bi-Directional LSTM (Bi-LSTM) for the proposed system

Sensor data consists of sequential signals that often include noise, adversely affecting prediction accuracy. To address this, the Bi-LSTM network is utilized. Unlike traditional LSTM networks, which process data in a unidirectional manner, the Bi-LSTM architecture (depicted in Figure 3.13) uses two stacked LSTM networks. The forward LSTM processes information from past values, while the backward LSTM learns from future values in reverse. These hidden states' information is then combined in the subsequent layer.

In Figure 3.13, x_{i-1}, x_i and x_{i+1} represent inputs at times t−1, t and t+1, while the weights of the LSTM gates are denoted by w. The outputs hi and hi′ correspond to the forward and backward directions, respectively. The output gate retains information about bi-directional steps.

In this study, the feature vector generated by the DCNN is fed simultaneously into the Bi-LSTM. To enhance performance and reduce overfitting, L2 regularization is applied, along with a tanh activation function for normalization.

Attention Mechanism

The attention layer serves two primary functions. First, it consolidates the outputs from the preceding Bi-LSTM layer, ensuring the outputs are assembled in channel order to preserve channel dependency in the input data. Second, it identifies and prioritizes essential representations for activity recognition.

The attention mechanism computes the final hidden state and attention score vectors by considering inputs from various Bi-LSTM channels. The representation scores are calculated using a dot-product function, which is adopted for its superior time complexity. After computing these scores, a softmax function is applied to

normalize them. The resulting normalized scores are aligned and summed to create context vectors, which are used for final classification.

Additional Layers of the Proposed System

The DCNN layer is instrumental in capturing subtle variations at each time step, which are critical for sequential learning in the Bi-LSTM layer during activity recognition. This layer is configured with 64 filters and uses the Rectified Linear Unit (ReLU) activation function.

A fully connected layer follows the attention layer, reducing the dimensionality of features and aiding in the final classification. Given the multi-class nature of the data, a softmax activation function is applied to predict the probability distribution over all classes.

To prevent overfitting, a dropout layer with a 0.5 dropout rate is placed after the dense layer. During training, this dropout randomly deactivates a portion of the neurons in the hidden layers. The categorical cross-entropy loss function is used during model compilation to calculate the cost on both training and validation sets. The Adam optimizer adjusts weights and biases through backpropagation, ensuring efficient convergence and optimal performance.

3.3.3. Dataset

The research examined six fundamental boxing techniques: left jab (LJB), right cross (RCR), right hook (RHK), left hook (LHK), right uppercut (RUC) and left uppercut (LUC). To maintain uniformity, data acquisition included two experienced lightweight-class competitors, each weighing around 62 kg and standing taller than 173 cm, with over five years of professional background. A total of 2,100 instances were gathered, yielding an approximate average of 350 entries per category, with no fewer than 300 per classification to ensure equilibrium.

To minimize the impact on athletes' movements, Cizmic et al. (2023) explain that the measuring device, a smartphone weighing 130 g with a 6.3-inch screen, was secured to the glove using a Velcro™ strap. Data was sampled at a frequency of 100 Hz, capturing three-axis accelerations (x, y, z) and human joint approach angles (pitch, roll, yaw) using accelerometer and gyroscope sensors. Each sample provided six-dimensional time-series data, ensuring consistency.

The data length for each sample varied depending on when recording began and ended. To standardize, 300 frames were selected around the punching action. Ma et al. (2018) and Rawashdeh et al. (1847) explain that linear acceleration was multiplied by the mass to calculate the force along each axis and the resultant forces were summed to determine the overall punch force. Using timestamps from the smartphone app, punch force and resultant acceleration were calculated. Samples were adjusted to 300 frames by truncating excess frames or padding with zeros. The dataset was split into 60% training and 40% testing using Scikit-learn's train-test split method.

Jayakumar et al. (2024) note that, given the size of the dataset, direct processing was impractical. Therefore, data segmentation with a 15-frame segment length was performed using a sliding window approach with 50% overlap. Data from different smartphone models was synchronized through down-sampling to ensure consistency.

The AT-DCNN-BL model was implemented using Keras with a TensorFlow backend. Its performance was compared with other RNN models using the sensor-acquired dataset. The proposed model achieved an accuracy of 92% and a cross-entropy loss below 0.10 after 200 training epochs. Hyperparameters were optimized using the GridSearch approach, with a dropout rate of 0.2 yielding the best results.

3.3.4. Performance Metrices

Several key evaluation metrics were employed to assess the models' performance. The evaluation matrices; accuracy, precision, recall, F1-score and the formulas to calculate those values are discussed in equation 3.2 to 3.5.

A confusion matrix was also used to visualize predictions. Columns in the matrix represent predicted categories, while rows indicate actual categories. This matrix is invaluable for identifying confusion among similar categories.

Accuracy and loss values were monitored during training and logged at the end of each epoch. The resulting graphs provided insights into model performance trends, allowing for timely adjustments to address underfitting or overfitting issues.

3.4. Methodology for Smartphone Based Fatigue Monitoring

Boxing, a high-risk sport, lacks robust fatigue assessment methodologies crucial for performance evaluation and injury prevention. Previous studies have explored low-cost inertial sensors like accelerometers and electromyography (EMG) to analyze fatigue-related performance metrics. However, these methods struggle to capture the dynamic nature of fatigue in high-intensity sports (Benages et al., 2019).

This study proposes a novel fatigue classification method using smartphone-integrated inertial sensors. It evaluates punch consistency through acceleration patterns, hand speed and time intervals between punches. The framework employs deep learning architectures, including CNNs, LSTMs, Bi-LSTMs, attention-enhanced LSTMs and hybrid CNN-LSTM models. Among these, the CNN-LSTM model achieved 99% accuracy in fatigue classification, demonstrating its ability to integrate spatial and temporal features effectively. The findings suggest significant potential for real-time fatigue monitoring, offering immediate feedback on key metrics. This cost-effective, non-invasive approach enhances training safety and reduces injury risks. By

integrating IMU sensors with deep learning, this study addresses critical gaps in boxing fatigue assessment and advances real-time sports performance monitoring (Hsu et al., 2018; Sha et al., 2020; Zhao et al., 2019; Wang et al., 2018). The experiments were conducted using a customized boxing glove embedded with sensors, as illustrated in Figure 3.14.

Figure 3.14 Smartphone sensor in the boxing glove

3.4.1. Proposed Methodology

3.4.1.1. Research Subject and Instrument Used

In today's fast-paced world, the prevalence of smart devices such as smartphones and smartwatches has grown significantly. These devices, often equipped with various embedded sensors, are widely utilized for a variety of applications. Among these sensors, the accelerometer, gyroscope and magnetometer stand out for their ubiquity and functionality. The accelerometer measures acceleration, the gyroscope captures angular velocity and the magnetometer determines the sensor's orientation relative to the Earth's magnetic field.

The proposed work leverages these integrated sensors within a smartphone to monitor muscle fatigue during boxing training. The conditions were categorized into two groups: fatigue and non-fatigue. For this study, an Android smartphone (Oppo, running Android 11) was utilized as the primary sensor device. However, smartphone-based sensors have certain limitations in boxing applications, including

inconsistencies due to placement variations and potential signal interference caused by sweat. To mitigate these challenges, the smartphone was securely mounted to the boxing glove worn by the participants (Figure 3.14). This position effectively minimizes placement inconsistencies and eliminates direct contact with sweat, ensuring consistent data capture. The activities were recorded using the "Sensor Logger," a paid Android application

3.4.1.2. Data Collection Process

The data collection process posed challenges due to the high volume of sensor-generated data. To conduct the study, five professional boxers were recruited as participants. The data collection utilized a smartphone running the "Sensor Logger" app at a sampling frequency of 250 Hz, generating 250 samples per second. For each participant, data was recorded over 1 minute and 10 seconds. The first two seconds of each session were excluded as transitional periods, leaving 68 seconds of usable data. Participants performed punches in 5-second bursts, with 2-second rest intervals between each burst. This setup generated 10 segments of 5-second recordings, mimicking patterns used in prior studies (Shepherd et al (2017); Biró et al. (2024).)

The experiment captured two primary variables: the acceleration of the dominant hand and the time between punches. Data from the accelerometer and gyroscope, both tri-axial sensors, was recorded across the X, Y and Z axes. This resulted in nine total columns: three for accelerometer data (x_acc, y_acc, z_acc), three for gyroscope data (x_gyr, y_gyr, z_gyr) and three for positional data (x_pos, y_pos, z_pos). The app saved the data as a .csv file, which was pre-processed by removing irrelevant strings and filling in missing values. The dataset comprised 85,000 rows, generated by five participants performing activities at a sampling rate of 250 Hz. Pre-processing procedures included cleaning and preparing this dataset for further analysis.

3.4.1.3. Data Pre-Processing

After collection, the raw data underwent pre-processing and feature engineering. Missing values were imputed using the median method and noise was reduced via filtering techniques. A label encoding approach converted categorical fatigue labels into numerical format. The dataset was split into two subsets: inputs and outputs. A K-fold cross-validation approach was applied for model training, with an 80-20 split between training and testing datasets. Feature scaling normalized the data using both Standard Scaler and Robust Scaler techniques. Scalar fitting and transformation were performed exclusively on the training set to prevent data leakage.

Given the time-series nature of the dataset, sliding windows were used to partition the data into 2.5-second segments, each containing 250 samples across nine columns. A stride or hop size was applied to minimize overlap, ensuring data diversity. These windows were transformed into 3D data formats suitable for neural network algorithms, particularly 2D CNNs. However, traditional machine learning models required data flattening, producing 2,250 columns per activity. To diminish overtraining risks, max-pooling techniques reduced data dimensionality. The data was reshaped to align with the requirements of deep learning algorithms.

3.4.2. Deep Learning Models

To classify fatigue from raw time-series data, this study employed several deep learning (DL) architectures, including LSTM, DCNN, Bi-LSTM, attention-based LSTM and a hybrid CNN-LSTM model. Figure 3.15 illustrates the data flow process within the proposed framework.

LSTM Model

The LSTM model employs sequential layering of components to analyze the raw sequential data. Batch normalization was applied post-LSTM layers to

standardize scattered data, followed by a dropout layer to reduce chances of overtraining.

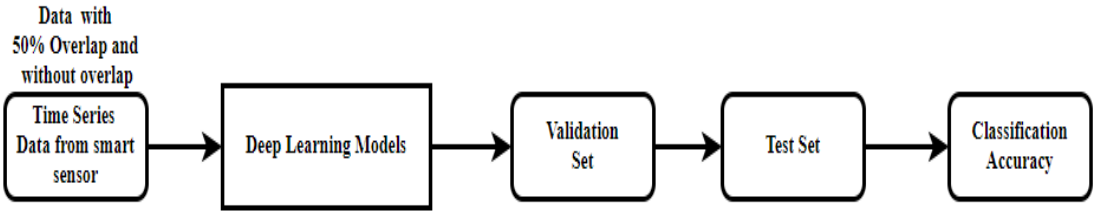


Figure 3.15 Data Flow Process for fatigue monitoring

Dense layers restricted weights to prevent excessive magnitudes, with 64 neurons utilized in the hidden layers. Inputs were defined with dimensions of (250, m), where 250 represents the sample rate and *m* the number of input features. L1 regularization, configured at 1%, was implemented to enhance robustness against outliers. The training process involved running the model for a total of 150 epochs, utilizing a batch size of 100 and applying a dropout rate of 0.5 to mitigate overfitting.

DCNN Model

The DCNN architecture included four convolutional layers, with two max-pooling layers alternatively placed between them. Convolutional layers extracted primary features from the time-series data, while pooling layers reduced feature dimensions and computed local sensitivities.

Table 3.5 Parameters of DCNN

Layer	Shape
Convolution layer	32
Max-pooling layer	32
Convolution layer I	64
Convolution layer II	128
Max-pooling layer	128
Convolution layer III	128
Flatten layer	200
Dense layer	2

Max-pooling replaced average pooling to optimize parameter adjustments. Following the convolutional and pooling layers, a flattening layer and dense layer were incorporated. ReLU served as the activation function for convolutional layers and softmax was utilized in the fully connected layer. Table 3.5 provides the architectural details of the DCNN model.

Bi-LSTM Model

The Bi-LSTM model extended the LSTM framework by incorporating bidirectional processing, enabling analysis of both past and future sequences. The forward LSTM layer produced a 3D tensor for input into the backward layers. Like the LSTM model, each hidden layer consisted of 64 neurons, with dropout layers applied to lessen overfitting problem. Dense layers and L1 regularization were similarly configured.

Attention-Based LSTM Model

This model introduced an attention mechanism, which alter feature weights to prioritize relevant information.

Table 3.6 Parameters of attention based LSTM

Layer	Shape
LSTM	128
Attention vector score layer	128
Dense layer	128
Output layer	2

Attention scores were computed using a bilinear method to normalize weights and convert them into probabilistic representations. The LSTM structure was enhanced with an attention layer post-dropout to refine feature selection. Table 3.6 provides the architectural details of the attention based LSTM model.

CNN-LSTM Hybrid Model

The hybrid model combined the temporal analysis capabilities of LSTM with the spatial feature extraction strengths of CNN. Accelerometer, gyroscope and magnetometer data were processed to account for both temporal and spatial correlations. Separate CNN and LSTM layers processed the initial signal arrays, with subsequent fusion enhancing feature representation. In the combined CNN-LSTM model (Figure 3.16), the LSTM component follows the previously described LSTM network, while the CNN component utilizes the architecture of a standalone CNN model to independently extract features. This study employs the cross-entropy loss function.

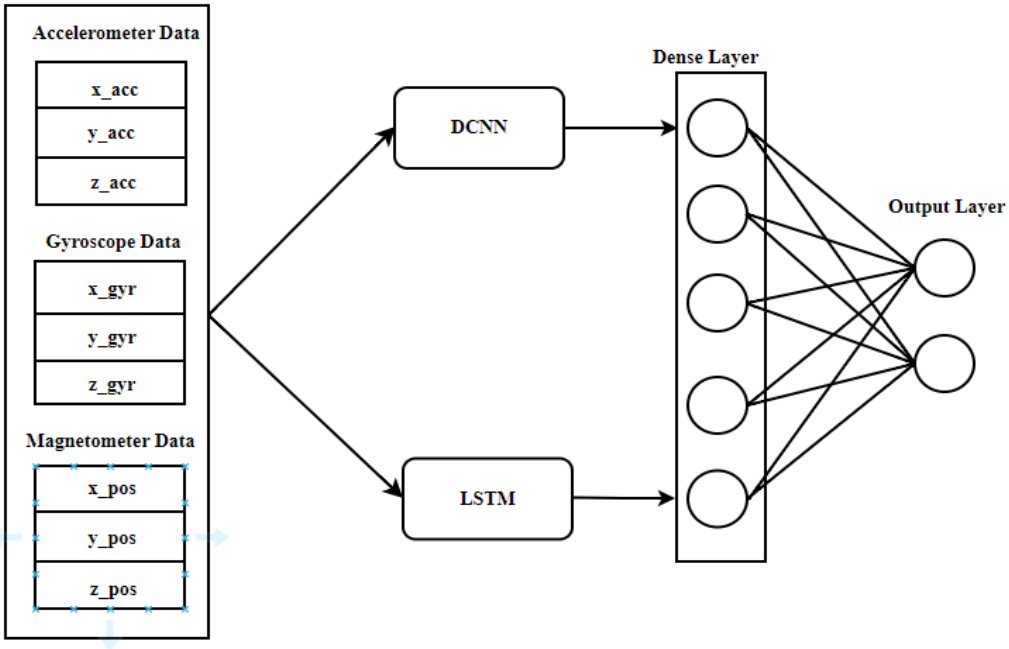


Figure 3.16 Combined CNN-LSTM model architecture

All models were trained using the TensorFlow 2.0 framework and Python 3.7.9. Data processing was facilitated by Pandas and NumPy and the Keras library supported model construction. Model training was performed on hardware featuring an Intel i7-6600U CPU, 16 GB of RAM and an AMD Radeon PRO W6400 GPU.

3.5. Summary

This chapter presents the methodology for boxing activity recognition using deep learning and sensor-based approaches. Traditional motion tracking methods are costly and inefficient, prompting the use of IMU sensors and smartphone-based systems for real-time punch classification and fatigue detection. The study employs a multi-sensor fusion approach, integrating IMU sensors on the wrists and upper back, processed using a DCNN for improved punch recognition. Optimal sliding window sizes are explored to enhance recognition speed and accuracy. A smartphone-based HAR system utilizing an AT-DCNN-BL model captures spatial and temporal dependencies for better classification. Real-time fatigue monitoring is conducted using a CNN-LSTM hybrid model, analyzing variations in punch force, speed and consistency. Various deep learning models, including DCNN, Bi-LSTM, MLPNN and RF, are compared, with hyperparameter tuning optimizing performance for portable deployment. By integrating multi-sensor data fusion and deep learning, this study enhances real-time sports monitoring for improved training, injury prevention and feedback.