Communications Engineering
(International Master's Program)

Technische Universität München

Master's Thesis

Wearable device-based Human Activity Recognition (HAR)

Mohammad Asif Ibna Mustafa

Communications Engineering (International Master's Program)

Technische Universität München

Master's Thesis

Wearable device-based Human Activity Recognition (HAR)

Author:	Mohammad Asif Ibna Mustafa
Examiner:	Univ.-Prof. Dr. Felix Dietrich
1st assistant advisor:	Prof. Dr. Roman Obermaisser
2nd assistant advisor:	Abu Shad Ahammed
Submission Date:	October 30th, 2024

I hereby declare that this thesis is entirely the result of my own work except where otherwise indicated. I have only used the resources given in the list of references.

October 30th, 2024 Mohammad Asif Ibna Mustafa

Acknowledgments

I would like to express my heartfelt gratitude to my examiner and assistant advisors for their invaluable support and mentorship throughout this thesis. Their constructive feedback, dedication, and insights were instrumental in guiding my work, and I am profoundly grateful for the time and effort they devoted to this thesis work.

My deepest appreciation also goes to Dr. Michael Weber, Lead Data Scientist at Cosinuss°, for his generous provision of the Cosinuss° device and continuous support throughout the research process. Dr. Weber's expertise was crucial in setting up the data collection portal and understanding sensor capabilities, which greatly strengthened the technical foundation of this study.

Abstract

Human Activity Recognition (HAR) is the process of identifying and monitoring physical activities through wearable sensor data, providing real-time insights that enhance patient care, personalized fitness planning, and lifestyle interventions. With the growing adoption of wearable devices, demand for reliable HAR systems capable of operating in diverse, free-living environments has increased, as these systems must handle varied data quality. Wearable device-based HAR, leveraging multimodal data, holds significant promise for applications like health monitoring, fall detection, and rehabilitation, where precise, continuous activity tracking is critical.

This study evaluates HAR performance using multimodal data from two wearable devices: the Cosinuss° C-Med° Alpha (an in-ear device) and the Garmin Venu 2 (a wrist-worn smartwatch). Data were collected from eight participants in a free-living setting, with Cosinuss° capturing continuous physiological metrics, including heart rate, blood oxygen saturation, body temperature, and accelerometer data. Garmin also recorded physiological data, such as heart rate and oxygen saturation, in addition to an extensive motion dataset comprising accelerometer, gyroscope, and GPS altitude readings. By independently assessing motion-only and motion plus physiological data, this study investigates how multimodal data integration, specifically adding physiological signals, impacts model performance, particularly for complex activities.

Various machine learning models (KNN, SVM, Random Forest, XGBoost) with handcrafted features and deep learning models (LSTM, ConvLSTM, Transformer) using raw features were evaluated. Machine learning models performed well with motion-only data for both simple and complex tasks, while deep learning models showed notable gains with the addition of physiological data. XGBoost achieved the highest accuracy among machine learning models, while Transformer led in deep learning across single and multi-subject datasets. The LSTM model showed a 30% improvement in weighted F1 score when using both motion and physiological data from Cosinuss°. Zero shot learning revealed that Cosinuss° performed better than Garmin in motion-only classification, while Garmin's combined motion and physiological data exhibited superior cross-subject generalization.

The results highlight the value of multimodal data integration in enhancing HAR for complex activities and suggest that Transformer and XGBoost models are particularly suited for robust healthcare and fitness applications. Future work may explore transfer learning and cross-subject training to broaden HAR's adaptability and generalizability.

Contents

1 Introduction

Throughout daily life, we engage in a diverse array of activities, from routine tasks like cleaning and driving to more structured activities such as playing games. These activities involve fundamental actions like standing, sitting, jogging, and running. While actions like sitting and standing may appear simple, more complex activities, such as eating, cooking, and commuting, pose greater challenges for accurate identification [29]. Precise recognition of these activities is essential for improving Human-Computer Interaction (HCI). Human Activity Recognition (HAR) automates the detection of physical actions by analyzing data from wearable devices and vision-based systems. It not only identifies movements but also provides insights into a person's goals and mental states, offering a deeper understanding of their behaviour [23].

The field of HAR is rapidly evolving and holds significant promise for transforming our understanding of human behaviour. It has diverse applications in healthcare [21], [33], athletic performance [35], and personal assistance systems [45], [31]. In healthcare, HAR is crucial for monitoring vital signs such as blood pressure, respiration rate, and Electrocardiogram (ECG) patterns, making it vital for telemedicine and personal health systems [66]. Its growing presence in robotics, security, and entertainment further enriches our ability to analyze and engage with human activities [42], [49].

Research on recognizing human activity can be divided into computer vision-based approaches [6] and sensor-based methods [12]. In computer vision-based human activity recognition, videos of activities are collected through cameras and then divided into sequences of frames. These frames are analyzed sequentially to detect specific activities [60], [41]. Although computer vision-based HAR offers high recognition accuracy, it is limited by environmental factors such as sensitivity to lighting conditions and has a restricted detection range, which can affect its performance in diverse settings [4].

Depending on the sensor arrangement, sensor-based activity recognition falls into two broad categories: multi-node and single-node frameworks. By capturing extensive motion data from several identical sensors, such as accelerometers, multi-node techniques usually achieve better accuracy [14]. However, the use of multiple sensors can be cumbersome, limiting ease of movement and reducing overall user comfort. To resolve this issue, recent methods aim to combine different sensor types into a single node, minimizing data loss while improving user comfort and practicality [40]. The development of sensor technology has dramatically improved the portability and capabilities of sensors [5].

The rise of the Internet of Things (IoT) has accelerated the integration of various sensor types, such as motion, optical, and temperature sensors, into wearable devices like smartwatches and in-ear health monitors, all while maintaining user comfort. These wearable

sensors capture a comprehensive array of multimodal data, including movement information from accelerometers and gyroscopes, as well as vital health metrics such as heart rate, blood oxygen levels, and body temperature. This combination of motion and physiological data is crucial for effective human activity recognition [38].

Different algorithms of machine learning have been widely used in various fields, including Human Activity Recognition (HAR), due to their ability to identify patterns and classify data. Such traditional machine learning includes decision trees [52], support vector machines (SVM) [3], and K-Nearest Neighbors (KNN) [47]. However, these approaches often face significant challenges, including the need for extensive domain knowledge, manual feature extraction, and the difficulty of real-time data processing [68]. To tackle these challenges, deep learning has emerged as a robust alternative. Unlike traditional machine learning approaches, deep learning models can automatically learn and extract features from raw sensor data, thus eliminating manual intervention. Various Convolutional Neural Network (CNN) architectures have been extensively explored in human activity recognition [30]. Recent advancements in recognizing complex activities have been made through Recurrent Neural Networks (RNNs), particularly Long Short-Term Memory (LSTM) networks, which excel at capturing temporal dependencies. Transformers, with their self-attention mechanisms, further enhance activity recognition by effectively modeling long-range dependencies and complex patterns within the data, making them well-suited for handling both temporal and spatial relationships in HAR [44]. However, these models still struggle with accurately identifying intricate activities, and data imbalance can adversely affect their performance.

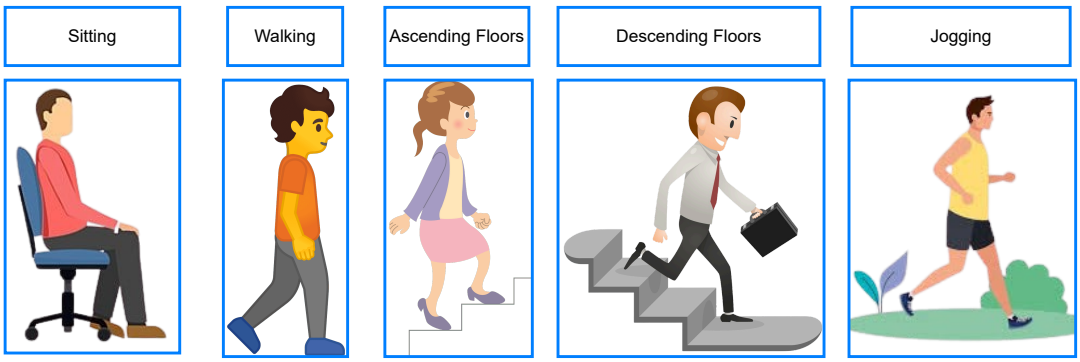


Figure 1.1: Typical day-to-day activities.

However, many existing public datasets for HAR are collected in controlled environments. To thoroughly evaluate HAR models, it is essential to gather data in free-living conditions where real-world variability can be introduced [34]. In this study, a multimodal dataset collected from two types of wearable devices, the Cosinuss° in-ear sensor and the Garmin smartwatch, is utilized in a free-living environment. These devices capture data from five crucial activities—walking, jogging, ascending stairs, descending stairs,

and sitting—as illustrated in Figure 1.1. This multimodal dataset combines movement and physiological data, which is vital for health monitoring, fitness tracking, and rehabilitation applications.

In this study, a structured approach to data collection, cleaning, and preprocessing is utilized to ensure dataset quality. The performance of traditional machine learning algorithms, including Support Vector Machines (SVM), k-Nearest Neighbors (KNN), Random Forest, and eXtreme Gradient Boosting (XGBoost), is compared with that of deep learning models, such as Long Short-Term Memory (LSTM), Convolutional LSTM (ConvLSTM), and Transformer architectures. Our analysis focuses on how model performance varies between motion-only data and multimodal data that includes physiological information, aiming to identify which device provides higher-quality data for HAR and which algorithms are most effective for complex activities with limited samples.

1.1 Research Questions

This study seeks to address the following research questions:

1. How do wearable sensors from Cosinuss° and Garmin compare in capturing data quality for HAR in a free-living environment, using both motion-only and motion plus physiological data?
2. What preprocessing techniques are necessary to build robust HAR datasets from wearable sensors, considering varying activity sample sizes?
3. How do machine learning and deep learning models (e.g., SVM, KNN, LSTM, ConvLSTM, Transformer) compare in recognizing complex activities when trained on both motion-only and motion plus physiological data?
4. How does model performance differ between individual models trained for each participant and models trained on the full dataset in both motion only and motion plus physiological dataset?
5. Do multimodal datasets (motion plus physiological) improve the recognition of complex activities with limited samples?

This research addresses these questions through an in-depth analysis of sensor data, model comparisons, and the impact of data quality and class imbalance on HAR outcomes, as outlined below:

In Section 2, related studies are reviewed, establishing foundational insights into wearable sensor technologies and introducing relevant machine learning and deep learning algorithms for activity recognition.

In Section 3, the experimental setup is described, and a comprehensive performance analysis of machine learning and deep learning models is provided for both the Cosinuss° and Garmin datasets, examining the effectiveness of motion-only and multimodal data.

In Section 4, the main findings of this research are summarized, and potential avenues for future work are outlined to further enhance activity recognition models.

2 State of the Art

In this section, we review related work and establish the theoretical foundations of classical machine learning and deep learning algorithms relevant to our study.

2.1 Related Work

The development of Human Activity Recognition (HAR) models relies heavily on high-quality datasets collected through wearable devices such as smartwatches, smartphones, and medical devices [53], [9]. Traditionally, motion data from inertial sensors such as accelerometers and gyroscopes, as well as image data from vision-based sensors, have been used for activity recognition. Prominent datasets like WISDM [36], ARAS [2], OPPORTUNITY [56], and UCI HAR [55] provide motion data from smartphones and wearable devices for time series analysis in HAR. These datasets have laid the foundation for early research into activity recognition. However, many of these datasets focus on controlled environments, making them less suited for evaluating real-world, free-living scenarios [34].

Recent research demonstrates that using multimodal datasets, which combine data from multiple sensor types, is more effective than single-modal datasets. Integrating data from different sensors, such as wearables and mobile devices, through data fusion techniques has been shown to reduce uncertainty and enhance HAR accuracy [50]. Different sensor modalities like Wearable Sensor-based HAR (WSHAR), Ambient Sensor-based HAR (ASHAR), and Hybrid Sensor-based HAR (HSHAR) for HAR were discussed [65]. The focus was on wearable sensor modality-centered HAR in healthcare, and the survey covered various steps involved in WSHAR, such as sensor selection and placement, data collection and pre-processing, feature extraction and selection, and classification algorithms [65].

Following up, emphasize the importance of incorporating physiological data into HAR models, particularly in healthcare settings. Systems such as Centinela integrated both accelerometer and physiological data, showcasing the ability of machine learning models to improve classification accuracy, especially for activities with similar movement patterns [38]. Although most HAR datasets focus primarily on motion data, those incorporating physiological metrics are still relatively rare. While motion sensors effectively capture movement patterns, they often miss the physiological and contextual aspects of human behaviour [15]. Notable multimodal datasets such as PAMAP2 [54], ETRI Lifelog [15], and EMG Physical Action [61] address this gap by incorporating both motion and physiological data, providing more comprehensive HAR systems.

Preprocessing techniques play a crucial role in ensuring the quality of sensor data for

HAR models. Sensor data is typically recorded at sampling rates between 20 and 50 Hz, depending on the sensor capacity and activity. After data collection, sensor signals must be cleaned and filtered to remove noise, spikes, and missing values, which is essential for real-world applications. Common methods include like nonlinear, low-pass, high-pass filters, bandpass filters [32], and Kalman filtering [39]. Data imputation is essential for handling missing values caused by sensor failures, improving model accuracy and minimizing the impact of incomplete data during analysis.

Machine learning (ML) and deep learning (DL) techniques have both played pivotal roles in advancing HAR systems. Traditional ML algorithms like K-Nearest Neighbors (KNN), Support Vector Machines (SVM), and Random Forests have been widely used for classifying activities, relying heavily on handcrafted features [57], [19] and extensive domain knowledge [46]. While effective, these approaches often struggle with scalability and generalization, particularly in the face of diverse sensor modalities and complex activity types.

The advent of deep learning has addressed many of these limitations. Deep learning models, particularly Convolutional Neural Networks (CNNs) and hybrid CNN-LSTM models, have been shown to outperform traditional ML methods by automatically extracting spatial and temporal features from raw sensor data. These models eliminate the need for manual feature engineering and have demonstrated significant improvements in HAR accuracy [63]. In recent years, Transformer models have shown significant potential in Human Activity Recognition (HAR) due to their ability to capture long-range dependencies in time-series data. Unlike RNNs and CNNs, Transformers rely on attention mechanisms, allowing them to process sequences in parallel and focus on the most relevant parts of the data. This has led to improved performance, especially in complex tasks. Studies suggest that Transformer-based architectures outperform traditional deep learning models in Natural Language Understanding and Computer Visision and Time Series [62], [22], [11], [24]. Finally, A comprehensive review of both ML and DL techniques in HAR is provided by recent survey [67]. It is emphasized that while traditional ML approaches remain relevant for computationally efficient and interpretable applications, the availability of large-scale sensor data has made deep learning the preferred method for capturing complex dependencies in HAR.

2.2 Sensor Information

This section describes the data collection methodologies for the Cosinuss° C-Med° Alpha in-ear sensor and the Garmin Venu 2 smartwatch. Both devices capture motion and physiological data, each with unique advantages and limitations.

2.2.1 Ear-Worn Sensor for Activity and Vital Metrics

The C-Med° Alpha by Cosinuss° is a compact and lightweight in-ear wearable device, designed for continuous monitoring of vital signs and motion data. Weighing just 7 grams, it features dimensions of 55.2 mm × 58.6 mm × 10.0 mm, making it comfortable for extended wear in various settings, such as hospitals, clinics, and daily use. The device includes a silicone earplug, available in three sizes (S, M, and L), ensuring a secure and customized fit that helps to minimize motion artifacts and ensures accurate data collection. The C-Med° Alpha continuously tracks essential health metrics, including heart rate, blood oxygen saturation (SpO_2), and body temperature, with a sampling rate of 1 Hz.

Figure 2.1: C-Med° Alpha in-ear sensor architecture: (1) Sensor head containing the photoplethysmography (PPG) sensor and temperature sensors, (2) Preprocessing block responsible for real-time signal processing, and (3) Triaxial accelerometer for capturing motion data, adapted from [17].

As shown in Figure 2.1, the core of the device features a photoplethysmography (PPG) sensor, housed in the sensor head, which utilizes alternating red (655 nm) and infrared (940 nm) light-emitting diodes to measure blood flow. This setup allows for precise, beat-by-beat readings of heart rate and SpO_2, providing reliable tracking even during physical activities. The sensor head also contains both an infrared thermometer and a contact thermometer to accurately measure body temperature from the outer ear canal.

Additionally, integrated within the main body of the device is a triaxial accelerometer, which captures three-dimensional motion data at a 100 Hz sampling rate, enabling precise monitoring of physical movements and body orientation. The preprocessing block processes all collected data in real time, transmitting it via Bluetooth Low Energy to connected mobile devices or the online health portal for instant monitoring and analysis.

2.2.2 Wrist-Worn Smart Watch for Activity and Vital Metrics

The Garmin Venu 2 is a versatile smartwatch, designed for comprehensive fitness and health tracking. Weighing just 49 grams, it is lightweight and comfortable for all-day wear. The watch features a 1.3-inch AMOLED touchscreen with a resolution of 416 × 416 pixels. It boasts a battery life of up to 11 days in smartwatch mode and 8 hours with GPS and music enabled. For location-based activities, it comes equipped with GPS/GLONASS/Galileo receivers, enabling accurate tracking of running, cycling, hiking, and other outdoor activities.

(a) Front view of the Garmin Venu 2 Smartwatch.

(b) Rear view of Garmin Venu 2: (1) optical heart rate sensor and (2) the charging port.

Figure 2.2: Overview of the Garmin Venu 2 Smartwatch: (a) displays the smartwatch's front interface and design, while (b) highlights the placement of the optical heart rate sensor and charging port on the back of the device, adapted from [28].

At the core of the Venu 2's health monitoring capabilities is the Elevate V4 sensor package as shown in Figure 2.2. The sensor package includes advanced systems for both heart rate and blood oxygen saturaton (SpO_2) measurement. Heart rate is measured continuously using green LEDs through photoplethysmography (PPG), while SpO_2 is monitored using red and infrared (IR) LEDs. The device has an upgraded system with two sets of red/IR LEDs and four signal paths, improving the accuracy of SpO_2 readings. Additionally, the four black rectangles surrounding the central sensor assist specifically with SpO_2 measurements by analyzing how much red and infrared light is absorbed by the blood, ensuring real-time monitoring of blood oxygen saturation during both activity and rest.

Beyond heart rate and SpO_2, the Venu 2 includes a triaxial accelerometer and gyroscope for tracking movement and orientation, providing accurate data for activity recognition. The smartwatch also monitors advanced health metrics such as heart rate variability (HRV), sleep stages, and stress levels, giving users a holistic view of their health. While

the standard software does not allow direct access to raw accelerometer and gyroscope data, a custom SDK enables data collection at a 20 Hz sampling rate, making the Venu 2 suitable for Human Activity Recognition (HAR) tasks. The collected data is stored in FIT files for further analysis, making the Garmin Venu 2 an ideal tool for both everyday fitness tracking and more detailed health studies.

2.3 Machine Learning

Machine Learning (ML) is a branch of artificial intelligence (AI) that focuses on developing systems capable of learning from data to make decisions without being explicitly programmed for specific tasks [26]. Machine learning algorithms use sample data, known as training data, to build models that are capable of making predictions based on unseen new input data.

Machine learning tasks can be classified into two primary classes: supervised learning and unsupervised learning. The most commonly utilized approach in practical applications is supervised learning. Both supervised and unsupervised learning utilize an input variable X, but in the case of supervised learning, it has an associated output variable Y [26]. Supervised learning works by learning a function that maps inputs to outputs, denoted as:

$$Y = f(X), \tag{2.1}$$

where Y is the predicted output and X is the input data. The objective of supervised learning is to approximate this mapping function closely, allowing the model to predict the output Y for any new input data X with high accuracy.

On the other hand, unsupervised learning does not have predefined output labels Y. Instead, its purpose is to explore the hidden structure or distribution within the data to uncover patterns or relationships that are not immediately apparent [10]. This type of learning is often used for clustering or association tasks, where the algorithm identifies groups or associations between the data points.

2.3.1 Support Vector Machine (SVM)

The Support Vector Machine (SVM) is a supervised learning model that is widely used for classification tasks [16], [59]. It works by finding a hyperplane in the feature space that maximally separates data points from different classes. For multi-class classification tasks, such as in Human Activity Recognition (HAR), the SVM can be extended using strategies like One-vs-Rest (OvR) or One-vs-One (OvO), allowing it to classify multiple activities.

Formally, the training dataset is denoted as:

$$T = \{(x_i, y_i), i = 1, 2, \ldots, N\}, \tag{2.2}$$

where $x_i \in \mathbb{R}^n$ represents the feature vector of the i-th training sample, and $y_i \in \{1, 2, \ldots, M\}$ represents the corresponding class label, with M being the number of classes and

N the number of training samples. The feature vector x_i is described as:

$$x_i = \left(x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n)}\right), \tag{2.3}$$

where n is the number of features.

In the linear SVM case, the model seeks to find a hyperplane defined by the vector w and bias term b that separates the classes. The decision boundary is given by:

$$w^T x + b = 0, \tag{2.4}$$

where the data points are classified according to the sign of the resulting value. To correctly classify the data points, the following constraints must be satisfied:

$$w^T x_i + b \geq 1 \quad \text{for } y_i = 1, \tag{2.5}$$

$$w^T x_i + b \leq -1 \quad \text{for } y_i = -1, \tag{2.6}$$

which ensures that data points from different classes are separated by a margin. The margin can be computed as:

$$\frac{2}{\|w\|}, \tag{2.7}$$

with the goal of maximizing this margin.

Thus, the SVM optimization problem can be formulated as minimizing the norm of w, subject to the classification constraints:

$$\min_w \frac{\|w\|^2}{2}, \quad y_i(w^T x_i + b) \geq 1, \quad \forall i. \tag{2.8}$$

This optimization problem is typically solved using Lagrange multipliers, leading to a quadratic programming formulation where the final decision function for classifying a new sample x_t is given by:

$$f(x_t) = \sum_{k=1}^{N} \alpha_k y_k x_k^T x_t + b, \tag{2.9}$$

where α_k are the Lagrange multipliers, and the predicted class for x_t is determined by the sign of $f(x_t)$, represented as $\hat{y}_t$:

$$\hat{y}_t = \text{sign}(f(x_t)). \tag{2.10}$$

2.3.2 K-Nearest Neighbors (KNN)

The K-Nearest Neighbors (KNN) algorithm is a widely used classification technique based on assigning a class label to a test sample by considering the majority class among its k-nearest neighbors in the feature space [64]. The parameter k denotes the number of

neighbors considered for classification, typically chosen as a small odd integer to avoid ties.

The training dataset used in the KNN algorithm follows the same notation as defined in Equation 2.2, where T represents the set of feature vectors and corresponding class labels. As before, $x_i \in \mathbb{R}^n$ represents the feature vector of the i-th training sample, and $y_i \in \{1, \ldots, M\}$ represents the corresponding class label, with M and N being the number of classes and training samples, respectively.

Given a test sample $x_t \in \mathbb{R}^n$, the algorithm computes the distance between the test sample and each training sample using a distance metric. For example, using Euclidean distance, the distance between x_t and a training sample x_i is calculated as:

$$d_{t,i} = \sqrt{\sum_{j=1}^{n} \left(x_t^{(j)} - x_i^{(j)} \right)^2}, \tag{2.11}$$

where $x_t^{(j)}$ and $x_i^{(j)}$ are the j-th features of the test sample and training sample, respectively. Once the distances $d_{t,i}$ are calculated, they are sorted in ascending order, and the top k-nearest neighbors are selected.

The corresponding class labels of these k-nearest neighbors are stored in an array:

$$C_k = [y_{t,1}, y_{t,2}, \ldots, y_{t,k}],$$

where $y_{t,i}$ denotes the class label of the i-th nearest neighbor. To classify the test sample, the algorithm performs majority voting based on the class labels of the nearest neighbors. The predicted class $\hat{y}_t$ is determined by the most frequent class among the k-nearest neighbors:

$$\hat{y}_t = \text{mode}(C_k), \tag{2.12}$$

where $\text{mode}(C_k)$ returns the class that appears most frequently in C_k.

By following this procedure, KNN assigns the test sample to the most common class among its k-nearest neighbors. The performance of the KNN algorithm depends significantly on the choice of k, the distance metric used, and the distribution of the training data.

2.3.3 Random Forest (RF)

Random Forest (RF) is an ensemble learning method that builds multiple decision trees, each trained on a random subset of the data and features [37]. For Human Activity Recognition (HAR), Random Forest (RF) is effective at handling complex classification tasks by combining the predictions from all trees through majority voting, reducing the risk of overfitting that is common in individual decision trees.

Each decision tree uses a criterion like Gini impurity to select the best feature splits at each node. The Gini impurity at node t for a feature X is defined as:

$$I(t, X) = 1 - \sum_{y=1}^{M} \left(\frac{n_y}{m}\right)^2, \tag{2.13}$$

where M is the number of classes, n_y is the number of samples belonging to class y, and m is the total number of samples at node t.

The Random Forest (RF) model selects the feature that minimizes the Gini impurity after the split. The Gini split index $G(r, X)$ is the weighted average of the Gini impurity across all child nodes, calculated as:

$$G(r, X) = \sum_{i=1}^{j} \frac{m_i}{N} I(t, x_i), \tag{2.14}$$

where j is the number of child nodes, m_i is the number of samples at the i-th node, and N is the total number of samples in the parent node.

Feature importance in Random Forest (RF) measures how much each feature contributes to reducing impurity across all trees. For a feature i in a decision tree, the importance is defined as:

$$FI_i = \frac{\sum_j G_{ij}}{\sum_k G_{ik}}, \tag{2.15}$$

where G_{ij} is the Gini impurity reduction for feature i at the j-th split, and G_{ik} is the total Gini impurity reduction across all splits in the tree.

The overall feature importance for feature i across all trees, denoted $RFFI_i$, is the average feature importance across all decision trees, which is given by:

$$RFFI_i = \frac{\sum_{j=1}^{T} FI_{ij}}{T}, \tag{2.16}$$

where T is the number of trees in the forest.

For multi-class HAR tasks, Random Forest (RF) predicts the activity by aggregating the votes from all decision trees. Each tree $h_t(x)$ casts a vote for a class y, and the final predicted class is determined by majority voting, expressed as:

$$\hat{y} = \arg\max_y \sum_{t=1}^{T} \mathbb{I}(h_t(x) = y), \tag{2.17}$$

where $\mathbb{I}$ is the indicator function.

By combining predictions from multiple trees and calculating feature importance, Random Forest (RF) offers a robust classification method, effectively handling multi-class problems and reducing overfitting through its ensemble approach.

2.3.4 eXtreme Gradient Boosting (XGBoost)

eXtreme Gradient Boosting (XGBoost) is an advanced ensemble learning algorithm that builds multiple decision trees iteratively to minimize residual errors [13]. For multi-class classification, the model is represented as a sum of K decision trees:

$$\hat{y}_i = \phi(x_i) = \sum_{k=1}^{K} f_k(x_i), \quad f_k \in \mathcal{F}, \tag{2.18}$$

where $f_k(x_i)$ is the score assigned by the k-th tree, and $\mathcal{F}$ represents the space of decision trees. XGBoost optimizes a regularized objective function:

$$L(\phi) = \sum_{i=1}^{N} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k), \tag{2.19}$$

where $l(y_i, \hat{y}_i)$ is the loss function (e.g., logistic loss for classification), and the regularization term $\Omega(f_k)$ is defined as:

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \|w\|^2, \tag{2.20}$$

with T as the number of leaves, w the leaf weights, and λ and γ regularization parameters.

At each iteration, XGBoost adds a new tree to minimize the following objective:

$$L^{(j)} = \sum_{i=1}^{N} l(y_i, \hat{y}_i^{(j-1)} + f_j(x_i)) + \Omega(f_j), \tag{2.21}$$

where the optimization leverages second-order Taylor expansion, and the approximate loss reduction after a split is given by:

$$L_{\text{split}} = \frac{1}{2}\left[\frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{\left(\sum_{i\in I_R} g_i\right)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{\left(\sum_{i\in I} g_i\right)^2}{\sum_{i\in I} h_i + \lambda}\right] - \gamma, \tag{2.22}$$

where g_i and h_i are the first and second derivatives of the loss function, and I_L and I_R denote the data subsets after splitting.

The final classification is obtained by applying the softmax function:

$$P(y|x) = \frac{e^{\hat{y}_y}}{\sum_{j=1}^{M} e^{\hat{y}_j}}, \tag{2.23}$$

and the predicted class $\hat{y}$ is:

$$\hat{y} = \arg\max_y P(y|x). \tag{2.24}$$

2.4 Artificial Neural Network (ANN)

Artificial Neural Networks (ANNs) are machine learning models inspired by biological neurons, designed to imitate brain functions. Rather than functioning as standalone algorithms, ANNs provide a structural framework used by various learning algorithms. An ANN consists of layers (input, hidden, and output) and weighted connections between neurons [26].

2.4.1 Multilayer Perceptron (MLP)

The Multilayer Perceptron (MLP) is a common ANN architecture with multiple layers, including an input layer, one or more hidden layers, and an output layer. The hidden layers enable the MLP to model complex non-linear relationships. Each layer consists of neurons, and connections between neurons are weighted by ω_{ij}^{l}, where ω_{ij}^{l} denotes the weight between neuron i in layer l and neuron j in layer $l+1$.

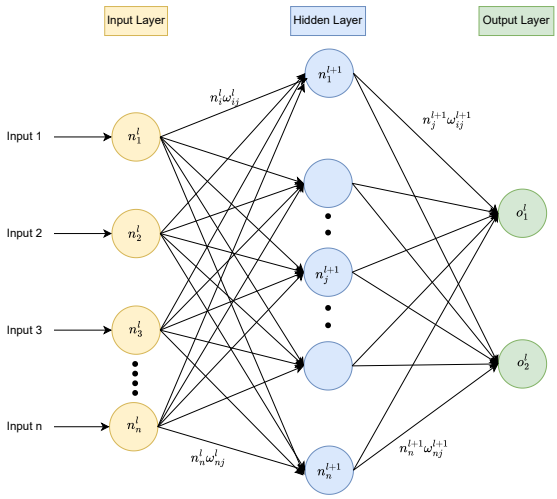


Figure 2.3: Architecture of a Multi Layer Perceptron (MLP) Network.

For neuron n_j^{l+1} in layer $l+1$, the total input v_j^{l+1} is the weighted sum of the outputs of neurons in the previous layer, as illustrated in Figure 2.3:

$$v_j^{l+1} = \sum_{i=0}^{n} o_i^l \omega_{ij}^l, \tag{2.25}$$

where o_i^l is the output of neuron n_i^l in layer l. This weighted sum is passed through an activation function $\phi(\cdot)$ to introduce non-linearity:

$$o_j^{l+1} = \phi(v_j^{l+1}), \tag{2.26}$$

where a commonly used activation function is the Rectified Linear Unit (ReLU), defined as:

$$\phi(v) = \max(0, v). \tag{2.27}$$

During training, the network minimizes the error between predicted and true outputs using a loss function. The weights are adjusted iteratively through backpropagation, which calculates the gradient of the error with respect to each weight, updating them to reduce the overall error [26]. The loss function measures this error, guiding the weight updates to improve performance across the network.

2.5 Deep Learning (DL)

Deep Learning (DL) is a subset of machine learning architectures that relies on artificial neural networks (ANNs) with multiple hidden layers to learn complex patterns from large datasets. These deep models, such as the Multilayer Perceptron (MLP), are capable of capturing intricate features but require significant computational resources and large amounts of data for effective training [48].

2.5.1 Recurrent Neural Network (RNN)

A Recurrent Neural Network (RNN) is designed to retain information over time, making it well-suited for processing sequential data [26]. Unlike feedforward neural networks, which assume independence between inputs, RNNs introduce loops in their architecture, allowing information to persist across different time steps. This is achieved by maintaining a hidden state h_t at each time step t, which serves as a memory of the previous inputs.

At each time step t, the hidden state h_t is updated based on the previous hidden state h_{t-1} and the current input x_t. The update is computed as:

$$h_t = \sigma(W_h h_{t-1} + W_x x_t + b_h), \tag{2.28}$$

where:

- h_{t-1} is the hidden state from the previous time step,
- x_t is the input at the current time step,
- W_h is the weight matrix applied to the hidden state,
- W_x is the weight matrix applied to the input,
- b_h is the bias term, and
- $\sigma(\cdot)$ is the activation function (commonly a sigmoid or tanh function) that introduces non-linearity into the model.

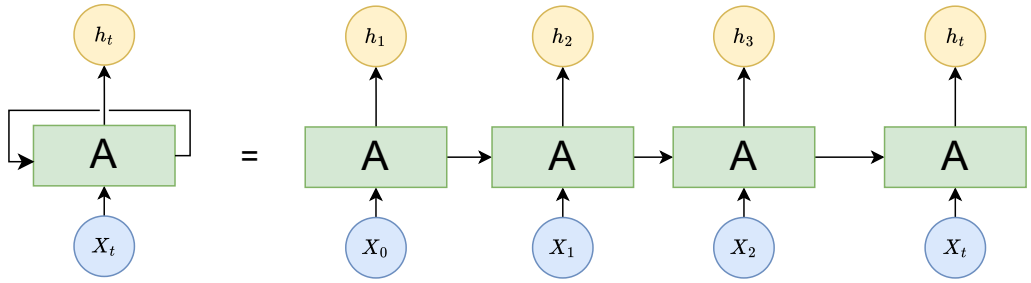


Figure 2.4: Unrolled Recurrent Neural Network (RNN) loop over time steps, showing the flow of information from one time step to the next via hidden states h_t and inputs x_t.

In Figure 2.4, the RNN loop is unrolled across multiple time steps, showing how the hidden state h_t is updated iteratively. Each block labeled A represents the computation performed at each time step, including the updating of the hidden state and the processing of the current input. The figure illustrates how the hidden states $h_1, h_2, \ldots, h_t$ are sequentially computed, and how the input $x_0, x_1, \ldots, x_t$ influences the evolution of the hidden states over time.

2.5.2 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM), shown in Figure 2.5, is an extension of RNNs designed to handle long-range dependencies in sequential data [27]. LSTMs introduce gates to control the flow of information, namely the input gate i_t, forget gate f_t, and output gate o_t, which are computed as:

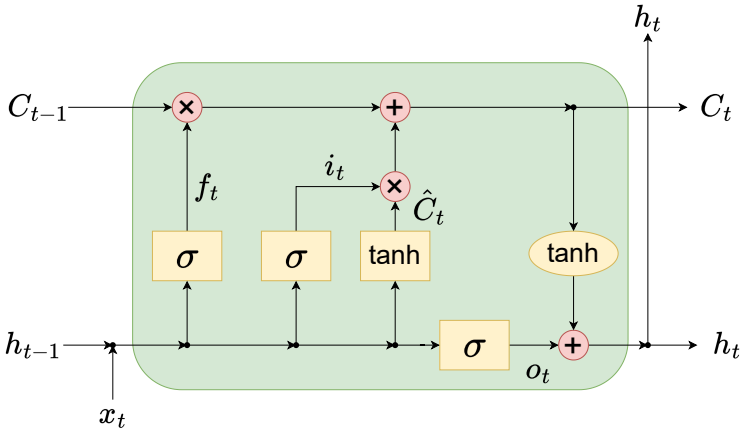


Figure 2.5: Structure of a Long Short-Term Memory (LSTM) Cell.

The input gate i_t is calculated as:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i), \tag{2.29}$$

while the forget gate f_t is computed as:

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f), \tag{2.30}$$

and the output gate o_t is given by:

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o). \tag{2.31}$$

The cell state C_t is updated using the forget gate and input gate as:

$$C_t = f_t \circ C_{t-1} + i_t \circ \tilde{C}_t, \tag{2.32}$$

where $\tilde{C}_t = \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$ is the candidate cell state. The hidden state h_t is then updated as:

$$h_t = o_t \circ \tanh(C_t), \tag{2.33}$$

where $\circ$ denotes element-wise multiplication, and $\sigma(\cdot)$ represents the sigmoid activation function.

2.5.3 Convolutional LSTM (ConvLSTM)

Convolutional LSTM (ConvLSTM) integrates convolutional layers with LSTM units to capture both spatial and temporal dependencies, making it ideal for tasks like human activity recognition [51]. In ConvLSTM, the gate operations are similar to those in the standard LSTM, but with convolutional operations ($*$) instead of matrix multiplications, as illustrated in Figure 2.6.

In ConvLSTM, each gate is computed as:

$$i_t = \sigma(W_{xi} * x_t + W_{hi} * h_{t-1} + b_i), \tag{2.34}$$

$$f_t = \sigma(W_{xf} * x_t + W_{hf} * h_{t-1} + b_f), \tag{2.35}$$

$$o_t = \sigma(W_{xo} * x_t + W_{ho} * h_{t-1} + b_o), \tag{2.36}$$

where $*$ denotes the convolution operation.

The cell state C_t is updated as:

$$C_t = f_t \circ C_{t-1} + i_t \circ \tilde{C}_t, \tag{2.37}$$

where the candidate cell state $\tilde{C}_t$ is computed as:

$$\tilde{C}_t = \tanh(W_{xc} * x_t + W_{hc} * h_{t-1} + b_c). \tag{2.38}$$

Finally, the hidden state h_t is updated as:

$$h_t = o_t \circ \tanh(C_t), \tag{2.39}$$

where $\circ$ denotes element-wise multiplication, and $\sigma(\cdot)$ and $\tanh(\cdot)$ represent the sigmoid and hyperbolic tangent activation functions, respectively. Here, W_{xi}, W_{hi}, etc., represent convolutional filters that capture spatial dependencies in the data.

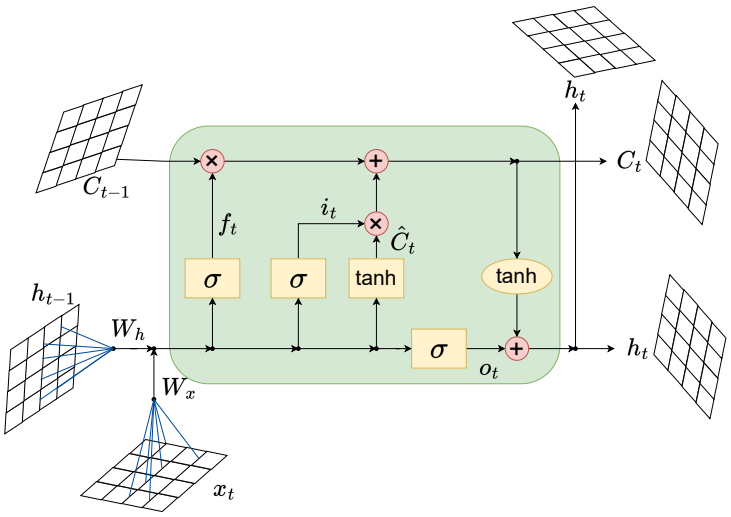


Figure 2.6: Convolutional LSTM (ConvLSTM). The convolutional filters (W_x and W_h) are shown as grid-like structures, representing their role in capturing spatial patterns, in contrast to the matrix multiplications in standard LSTMs.

2.5.4 Transformer

The Transformer model is a powerful sequence-to-sequence architecture that has been widely adopted for various tasks, including classification [43]. In this work, time series classification is modeled using an encoder-only Transformer approach, which maps sequences of sensor measurements to class labels, as illustrated in Figure 2.7 [58]. A crucial advantage of this approach is its ability to capture long-range dependencies in the time series data, making it particularly effective for tasks like Human Activity Recognition, where capturing both short-term and long-term patterns is essential.

The training dataset follows the same notation as defined in Equation 2.2, where $x_i \in \mathbb{R}^{k \times n}$ is the feature matrix for the i-th training sample, with k representing the sequence length and n the number of features, and $y_i \in \{1, 2, \ldots, M\}$ is the corresponding class label. The task is to map the sequence x_i to a class label y_i through the Transformer architecture.

2.5.4.1 Input Embedding and Positional Encoding

Each sequence $x_i \in \mathbb{R}^{k \times n}$ is first embedded into a higher-dimensional space through a learned linear transformation. This results in an embedding $E(x_i) \in \mathbb{R}^{k \times d}$, where d is the embedding dimension. Additionally, to encode the temporal order of the sequence, a positional encoding $P \in \mathbb{R}^{k \times d}$ is added to the input embeddings, giving the final input to

the Transformer as:

$$Z_0 = E(x_i) + P, \tag{2.40}$$

where $Z_0 \in \mathbb{R}^{k \times d}$ represents the input to the Transformer layers.

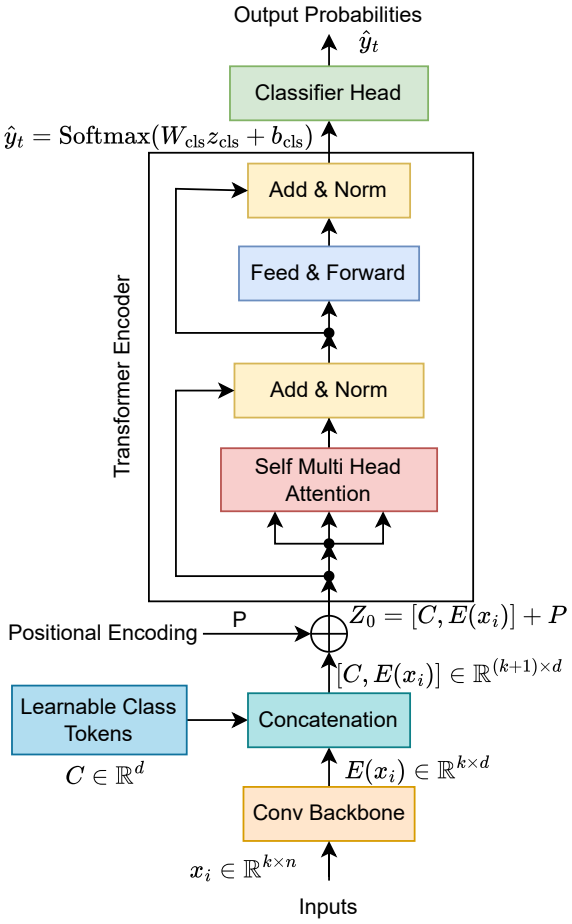


Figure 2.7: Transformer Architecture for Human Activity Recognition.

2.5.4.2 Transformer Encoder

The Transformer Encoder consists of L layers, where each layer comprises two main components: multi-head self-attention (MHA) and a feedforward network. The MHA mechanism is responsible for capturing relationships between different positions in the sequence. For a given input sequence Z, the self-attention mechanism computes the attention scores as follows:

$$A(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) V, \tag{2.41}$$

where $Q, K, V \in \mathbb{R}^{k \times d}$ are the query, key, and value matrices, and d is the dimension of the embedding space. The attention mechanism allows the model to weigh the importance of different time steps in the sequence. In multi-head attention, this operation is repeated for multiple attention heads, each focusing on different parts of the sequence:

$$\text{MHA}(Q, K, V) = \text{concat}(A_1(Q, K, V), \ldots, A_h(Q, K, V)), \tag{2.42}$$

where h is the number of attention heads, and each head A_i computes the attention independently.

After the attention block, the output is passed through a feedforward network (FFN) consisting of two fully connected layers with a non-linearity in between:

$$\text{FFN}(Z) = \text{GELU}(ZW_1 + b_1)W_2 + b_2, \tag{2.43}$$

where W_1, W_2 are learned weight matrices, and b_1, b_2 are biases.

The complete Transformer layer is then computed as:

$$Z' = \text{LayerNorm}(Z + \text{MHA}(Z, Z, Z)), \tag{2.44}$$

$$Z = \text{LayerNorm}(Z' + \text{FFN}(Z')), \tag{2.45}$$

where Z' represents the output of the attention block, and Z is the final output of the layer after the feedforward network and layer normalization.

2.5.4.3 Classification Head

Once the input passes through the Transformer encoder, the output corresponding to the first position (class token) is extracted. This output, $z_{\text{cls}} \in \mathbb{R}^d$, represents the entire sequence. The classification is performed by passing z_{cls} through a fully connected layer followed by a softmax activation:

$$\hat{y}_t = \text{Softmax}(W_{\text{cls}} z_{\text{cls}} + b_{\text{cls}}), \tag{2.46}$$

where $W_{\text{cls}} \in \mathbb{R}^{d \times M}$ is the weight matrix mapping the latent vector to the class probabilities, and $\hat{y}_t$ is the predicted class distribution.

2.6 Hyperparameter Optimization

Hyperparameter optimization is a crucial step in both Machine Learning (ML) and Deep Learning (DL) tasks, as it directly impacts the model's performance. Hyperparameters are the parameters set before the training process begins and are not learned from the data. These include learning rates, batch sizes, number of layers, and many other model-specific parameters. Optimizing hyperparameters can lead to better generalization and performance on unseen data, and there are various techniques available to efficiently search for the best hyperparameter combinations.

For machine learning models, simpler optimization techniques like Random Search are often effective due to the lower computational cost and smaller search space [8]. However, for deep learning models, which tend to have more complex architectures and larger hyperparameter spaces, more sophisticated optimization methods like Bayesian Optimization with Tree-structured Parzen Estimators (TPE) are employed to efficiently navigate the high-dimensional search space [8], [1].

2.6.1 Random Search for Machine Learning Models

Random Search is a hyperparameter optimization technique where hyperparameter values are randomly sampled from predefined distributions, and the model performance is evaluated for each sampled set of hyperparameters. Unlike grid search, which systematically explores all combinations of hyperparameters, random search selects a subset of combinations based on randomness, which often leads to more efficient exploration of the hyperparameter space [8].

Formally, the optimization problem is to find the set of hyperparameters x^* that maximizes the objective function $f(x)$, where $x \in \mathcal{X}$ represents a combination of hyperparameters. The objective is expressed as:

$$x^* = \arg\max_{x \in U} f(x), \tag{2.47}$$

where U is the set of candidate hyperparameter configurations, and $f(x)$ represents the model performance, typically measured by a validation metric such as accuracy, F1-score, or mean squared error.

In Random Search, the hyperparameter values x are sampled independently from predefined distributions. For each trial k, the process of Random Search is as follows:

1. Randomly sample a hyperparameter set x_k from the search space:

$$x_k \sim P(x), \quad x_k \in U, \tag{2.48}$$

2. Evaluate the model performance using the hyperparameter set x_k, denoted as $f(x_k)$.
3. Repeat the process for a predefined number of trials T, storing the performance $f(x_k)$ for each sampled hyperparameter set.

At the end of the search process, the best hyperparameter set x^* is the one that maximizes the objective function:

$$x^* = \arg\max_{x_k, k=1,2,\ldots,T} f(x_k). \tag{2.49}$$

Random search is simple to implement and often results in good hyperparameter configurations, especially for smaller search spaces, as it can avoid exploring suboptimal regions in a grid-like manner [8].

2.6.2 Tree-structured Parzen Estimators (TPE) for Deep Learning Models

For deep learning models, the search space of hyperparameters is typically much larger and more complex. As a result, Random Search may not be efficient enough. Instead, advanced optimization methods such as Bayesian Optimization are employed to systematically explore the search space. One popular Bayesian method is Tree-structured Parzen Estimators (TPE), which is used in the present study for deep learning model optimization [7].

TPE constructs a probabilistic model mapping hyperparameters to the objective function, based on past evaluations. The objective is to maximize the function $f(x)$, where $x \in \mathcal{X}$ represents the hyperparameter values, and the best set of hyperparameters x^* is defined as:

$$x^* = \arg\max_{x \in U} f(x), \tag{2.50}$$

where U represents the set of all candidate hyperparameter sets.

An important step in TPE is the Expected Improvement (EI) criterion, which is used to choose the next set of hyperparameters to evaluate. The EI criterion can be expressed as:

$$EI_{y^*}(x) = \int_{-\infty}^{\infty} \max(y^* - y, 0) p_M(y|x)\, dy, \tag{2.51}$$

where y^* is the threshold for improvement, y is the objective function value, and $p_M(y|x)$ is the probability density of obtaining value y given hyperparameter set x.

TPE models the probability $p(x|y)$ as a combination of two densities:

$$p(x|y) = \begin{cases} l(x), & \text{if } y < y^* \\ g(x), & \text{if } y \geq y^* \end{cases} \tag{2.52}$$

where $l(x)$ represents the probability distribution of hyperparameters resulting in values lower than y^*, and $g(x)$ represents the probability distribution for values higher than y^*.

The Expected Improvement is maximized using these densities:

$$EI_{y^*}(x) = \gamma l(x) - (1-\gamma) g(x) \left(\frac{\gamma}{1-\gamma} + \frac{g(x)}{l(x)} \right)^{-1}, \tag{2.53}$$

where γ represents the ratio of the number of evaluations with $y < y^*$ to the total number of evaluations.

TPE efficiently selects the most promising hyperparameters to evaluate in the next iteration, reducing the overall number of evaluations required to find the optimal configuration [7].

3 Wearable device-based Human Activity Recognition (HAR)

This section outlines the experimental design employed for Human Activity Recognition (HAR) in this thesis, leveraging data from two wearable devices: the Cosinuss° C-Med° Alpha in-ear sensor and the Garmin Venu 2 smartwatch. These devices independently capture physiological and motion data during activities such as sitting, walking, jogging, and stair navigation. The experimental setup is illustrated in Figure 3.1.

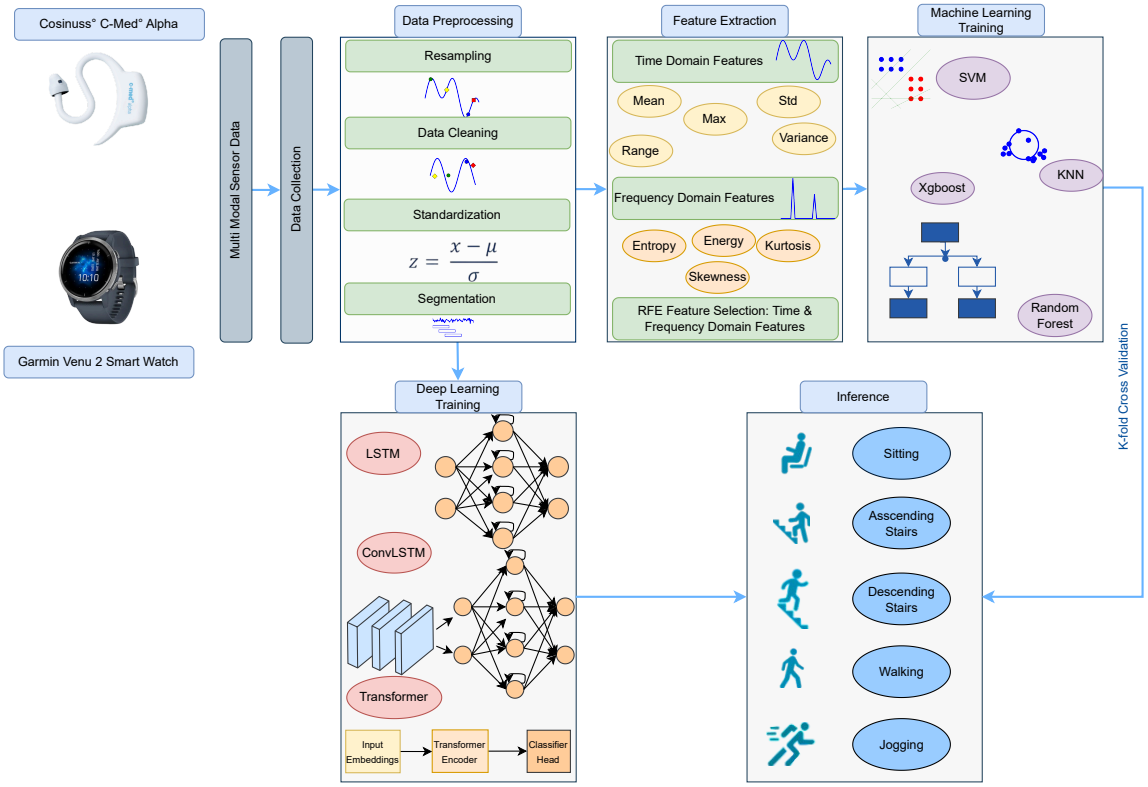


Figure 3.1: Experimental setup for Human Activity Recognition (HAR) using Cosinuss° C-Med° Alpha in ear device and Garmin Venu 2 smart watch.

The data collection process is thoroughly explained in Section 3.1, followed by a description of the dataset and exploratory data analysis (EDA) in Section 3.2. After collecting data,

the dataset goes through a thorough preprocessing pipeline that includes data cleaning, resampling, and noise filtering, as discussed in Section 3.4. Handcrafted features derived from both the time and frequency domains are introduced in Section 3.5.

To improve model performance, Recursive Feature Elimination (RFE) is employed to select the most relevant features, as detailed in Section 3.6. Techniques for windowing and segmentation, which are crucial for preparing time-series data for machine learning and deep learning models, are presented in Section 3.7.

Finally, model performance is evaluated using various metrics described in Section 3.9. The impact of incorporating both motion and physiological data on human activity recognition (HAR) is evaluated by comparing traditional machine learning models—SVM, KNN, Random Forest, and XGBoost—with deep learning models such as LSTM, ConvLSTM, and Transformer. Detailed results are provided in Section 3.10.

3.1 Data Acquisition Workflows for Wearable Devices

This section summarizes the data collection workflows for the Cosinuss° C-Med° Alpha in-ear sensor and the Garmin Venu 2 smartwatch. Each device captures both physiological and motion data using unique methods for transmission and analysis.

3.1.1 Data Workflow for Cosinuss° C-Med° Alpha Using the Health Portal

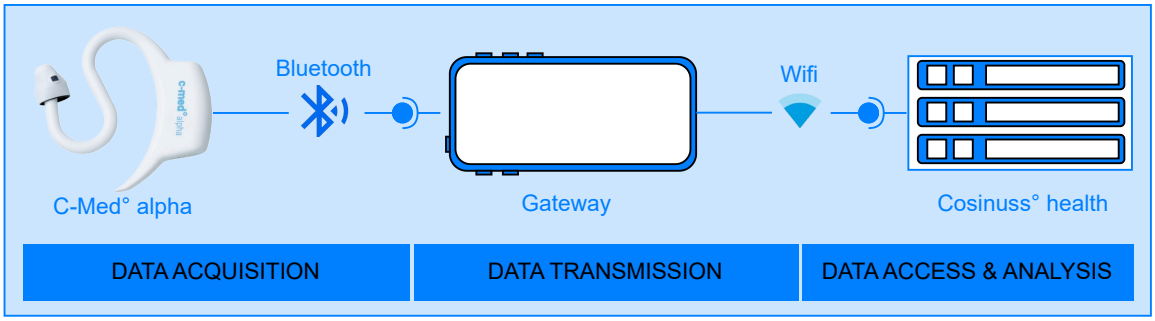


Figure 3.2: Cosinuss° C-Med° Alpha data collection workflow.

The data collection process for the Cosinuss° C-Med° Alpha device involves multiple stages, as illustrated in Figure 3.2. The device acquires physiological data such as body temperature, heart rate, and blood oxygen saturation (SpO_2), as well as motion data from the embedded accelerometer. This data is transmitted via Bluetooth Low Energy (BLE) to a gateway, which could be a smartphone or tablet, where it is further relayed through a Wi-Fi connection to the Cosinuss° Health server. The server provides access to real-time data monitoring, and analysis. Data is stored and made accessible for download via the Cosinuss° Health portal for further preprocessing and analysis. The accelerometer

captures motion at 100Hz, while vital signs are recorded at 1Hz, enabling a comprehensive dataset for analysis.

3.1.2 Data Collection Workflow for Garmin Venu 2 Using Custom Software Development Kit (SDK)

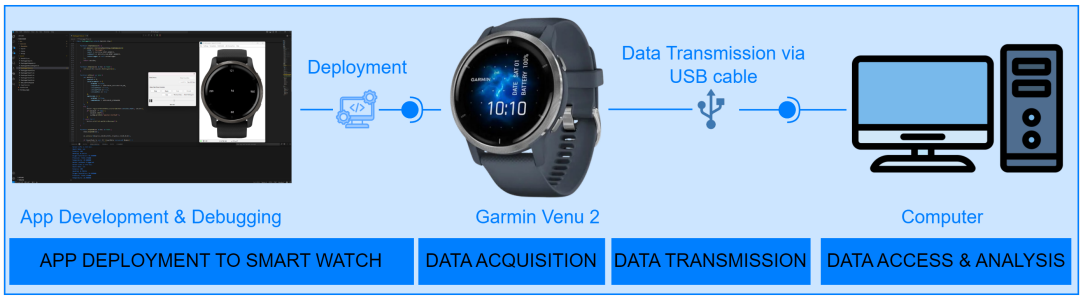


Figure 3.3: Garmin data collection workflow.

The Garmin Venu 2 smartwatch data collection workflow, shown in Figure 3.3, overcomes the default limitation of accessing raw accelerometer and gyroscope data by deploying a custom SDK built using Garmin's Connect IQ platform. The Software Development Kit (SDK), built using the Monkey C programming language, utilizes APIs from the Toybox framework to enable high-frequency data collection at 20Hz for motion data (accelerometer and gyroscope) and 1Hz for physiological metrics, including heart rate, blood oxygen saturation (SpO_2), and temperature. The custom application is deployed to the smartwatch, allowing it to record data in real-time. Once data is collected, it is transmitted to a computer via USB, where it is stored in the .FIT format for efficient handling. Tools like FitSDK are used to decode the data into readable formats, enabling further analysis and insights into the motion and physiological metrics collected.

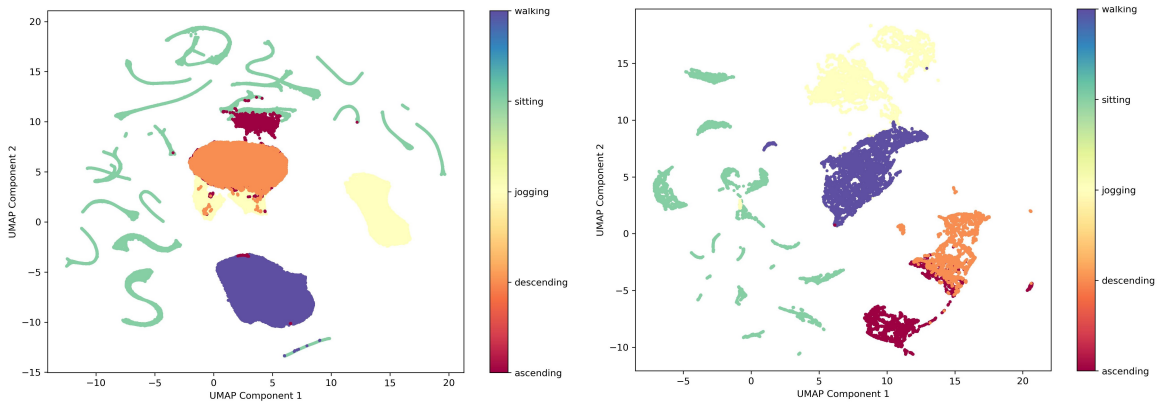
3.2 Human Activity Recognition Dataset

The study initially involved 9 participants; however, data from one participant was excluded due to a Cosinuss° server issue. As a result, the final analysis was conducted with 8 healthy participants, with an average age of 27, comprising 7 males and 1 female. Each participant was assigned a unique user ID and wore both Cosinuss° and Garmin devices during data collection to capture synchronized physiological and motion data. Data was recorded during five specific activities: sitting, walking, jogging, ascending, and descending stairs. For sitting, walking, and jogging, approximately 5 minutes of continuous recording was performed. The duration of recording the ascending and descending stairs

varied based on the number of floors at each of the four data collection locations. In one case, a participant performed only four activities (sitting, ascending stairs, descending stairs, walking) due to scheduling constraints. During jogging sessions, if a participant experienced fatigue, recording was paused and resumed after a rest period; otherwise, continuous 5-minute sessions were maintained. Both Cosinuss° C-Med° Alpha and Garmin Venu 2 devices were used simultaneously to ensure accurate timing across both platforms. Activity timing was managed using the ATracker mobile app for time tracking, with the researcher instructing the participants when to start and stop each activity. The recorded times and corresponding activities were logged in the app and later used to annotate the datasets.

Table 3.1: Distribution of Activities and Duration Statistics for Cosinuss° C-Med° Alpha and Garmin Venu 2.

Activity	Percentage (%)	Total Time (s)	Mean (s)	Std (s)	Min (s)	Max (s)
sitting	28.00	2623.00	327.88	39.00	301.00	401.00
ascending	10.94	1025.00	128.13	42.43	72.00	179.00
descending	10.15	951.00	118.88	45.30	45.00	174.00
walking	27.67	2592.0	324.00	21.22	305.00	371.00
jogging	23.23	2175.95	310.85	18.76	277.95	334.00



(a) UMAP visualization for Cosinuss° dataset.

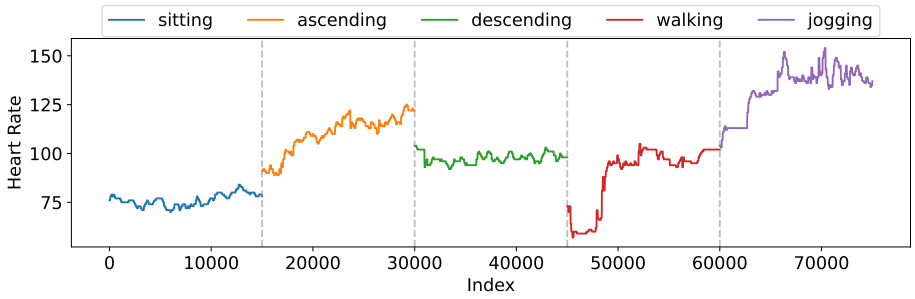
(b) UMAP visualization for Garmin dataset.

Figure 3.4: Uniform Manifold Approximation and Projection (UMAP) visualizations comparing Cosinuss° and Garmin devices for subject S4.

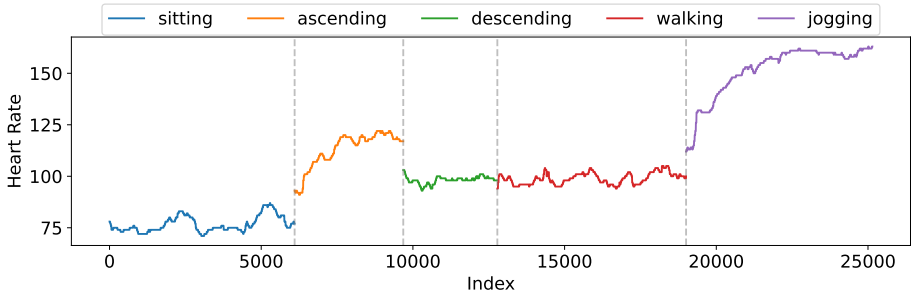
Table 3.1 provides an overview of the distribution and duration statistics for each activity, displaying the total time, mean, standard deviation, and range (minimum and maximum) of recording durations across participants. The dataset contains a total of 9,06,585 samples for the Cosinuss° device and 186,620 samples for the Garmin device. The total

recording time across all activities was approximately 156.1 minutes. Sitting and walking constituted the highest percentages of total recording time, while ascending and descending activities were recorded for shorter, variable durations due to differences in available stairs at each location. These statistics offer insight into the variability and consistency of the dataset, forming the basis for evaluating activity-specific performance in later analyses.

Figure 3.4 presents a comparison of Uniform Manifold Approximation and Projection (UMAP) visualizations for subject S4, generated from C-Med° Alpha and Venu 2 devices. Figure 3.4a displays data from the Cosinuss° device, which includes an accelerometer and physiological features, while Figure 3.4b visualizes data from the Garmin device, encompassing a broader set of sensors, including accelerometer, gyroscope, and altitude data, etc. The color-coded clusters correspond to different activities, such as walking, sitting, jogging, ascending, and descending.



(a) Heart rate time series samples for subject S1 using the Cosinuss° C-Med° Alpha for different activities.

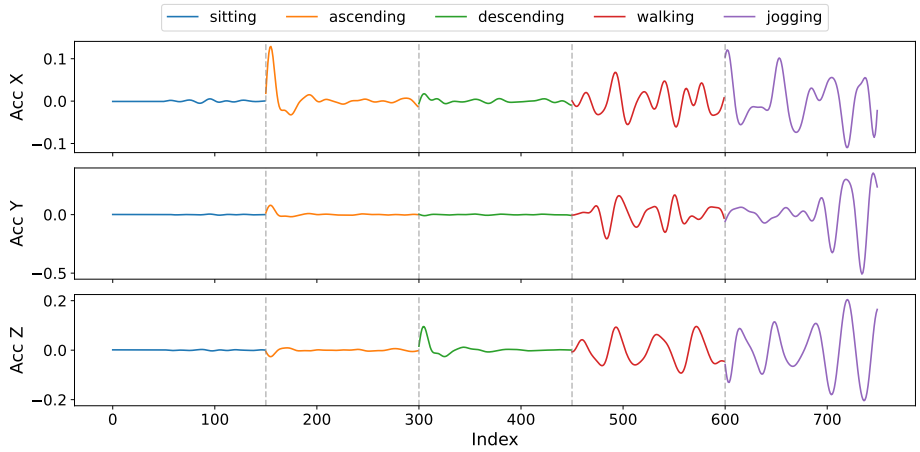


(b) Heart rate time series samples for subject S1 using the Venu 2 for different activities.

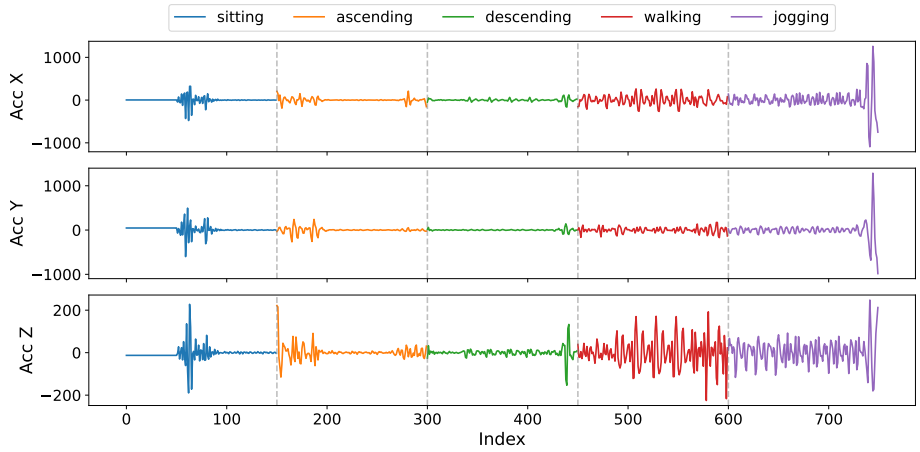
Figure 3.5: Comparison of heart rate time series samples for subject S1 using the Cosinuss° C-Med° Alpha and Garmin Venu 2 devices during different activities.

Figure 3.5 compares heart rate time series samples for subject S1 using the Cosinuss° C-Med° Alpha and Garmin Venu 2 sensors during different activities. These plots show a subset of the full time series data, highlighting trends during various activities. In both cases, heart rate remains relatively stable during sitting, rises sharply during ascending,

slightly decreases during descending, and peaks during jogging, reflecting the varying intensity levels of each activity. The heart rate samples captured by the Cosinuss° sensor in Figure 3.5a demonstrate a clear increase during ascending, with the highest rates observed during jogging. Similarly, the Garmin sensor data in Figure 3.5b shows comparable trends, confirming the consistency in physiological responses captured by both devices.



(a) Accelerometer time series samples for subject S1 using the Cosinuss° device for different activities.



(b) Accelerometer time series samples for subject S1 using the Garmin device for different activities.

Figure 3.6: Comparison of accelerometer time series samples for subject S1 using the C-Med° Alpha and Garmin Venu 2 devices during different activities.

Figure 3.6 presents accelerometer time series samples for subject S1, recorded using the C-Med° Alpha and Garmin Venu 2 devices during different activities. Both plots highlight

distinct movement patterns for each activity. The Cosinuss° data in Figure 3.6a shows smoother transitions between activities and lower amplitude variations, likely due to its in-ear placement, which is more sensitive to head movements. The jogging phase displays periodic peaks, while the sitting phase remains nearly flat. In contrast, the Garmin data in Figure 3.6b captures larger fluctuations, particularly in the z-axis, reflecting its wrist-worn placement and sensitivity to full-body and arm movements. The higher intensity during jogging and walking is evident in the pronounced z-axis spikes.

3.3 Feature Definition

Data from both the Cosinuss° C-Med° Alpha and Garmin Venu 2 devices were collected and processed using a standardized methodology to ensure consistency across the dataset. The C-Med° Alpha device primarily recorded physiological metrics, including heart rate, SpO_2, body temperature, and triaxial accelerometer data, while the Venu 2 captured motion data from its accelerometer and gyroscope, along with heart rate, SpO_2, and GPS-based activity metrics. Below is a summary of the key features extracted from both devices:

- **Accelerometer**: Captures raw motion along the x, y, and z axes, offering insights into the subject's physical activity levels and movement patterns.
- **Gyroscope**: Measures rotational movement along the x, y, and z axes, indicating changes in orientation and angular velocity.
- **Heart Rate**: Recorded using photoplethysmography (PPG), this feature represents beats per minute (BPM), providing information on cardiovascular activity.
- **Oxygen Saturation (SpO_2)**: Indicates the percentage of oxygen-saturated hemoglobin in the blood, a key metric for assessing respiratory function and physical exertion.
- **Temperature**: Monitors both core body temperature and ambient temperature, offering valuable insight into the subject's physiological state.
- **Perfusion Index**: Reflects the strength of the PPG signal and serves as an indicator of blood flow in the measured area.
- **Distance**: Tracks the total distance traveled in meters, relevant for assessing activity levels during exercises such as walking or running.
- **Enhanced Altitude**: Provides precise altitude readings in meters above sea level, using GPS or barometric sensors for accuracy.
- **Cadence**: Measures the rate of steps or pedaling, usually expressed as revolutions per minute (RPM), offering insights into locomotion efficiency.

- **Fractional Cadence**: A refined version of cadence, which includes fractional values for greater precision in RPM measurements.
- **Pressure**: Records atmospheric pressure in Pascals (Pa), which can be used to provide additional context for altitude and environmental conditions.
- **Heading**: Captures the direction of movement in degrees, valuable for navigation and orientation during outdoor activities.
- **Enhanced Speed**: Measures velocity in meters per second (m/s), offering precise information about the subject's movement speed during activities.

Table 3.2: Summary of Features, Device Support, and Sampling Rates.

Feature	Cosinuss°	Garmin	Cosinuss° Rate (Hz)	Garmin Rate (Hz)
Heart Rate	Yes	Yes	1	1
Oxygen Saturation (SpO_2)	Yes	Yes	1	1
Body Temperature	Yes	No	1	N/A
Accelerometer	Yes	Yes	100	20
Gyroscope	No	Yes	N/A	20
Perfusion Index	Yes	No	1	N/A
Distance	No	Yes	N/A	Variable
Enhanced Altitude	No	Yes	N/A	Variable
Cadence	No	Yes	N/A	1
Fractional Cadence	No	Yes	N/A	1
Pressure	No	Yes	N/A	1
Heading	No	Yes	N/A	1
Latitude/Longitude (GPS)	No	Yes	N/A	Variable
Enhanced Speed	No	Yes	N/A	Variable

Table 3.2 summarizes the key features supported by the Cosinuss° C-Med° Alpha and Garmin Venu 2 devices, along with their respective sampling rates. While both devices support physiological metrics like heart rate and SpO_2, only Garmin provides additional motion-related features such as gyroscope data, GPS, and altitude tracking at variable sampling rates.

3.4 Data Preprocessing

This section outlines the steps involved in preprocessing raw sensor data and the subsequent feature engineering process. These steps transform the data into a format suitable for machine learning and deep learning models. Both Cosinuss° and Garmin sensor data were processed through a comprehensive pipeline that includes filtering, noise removal, outlier removal, and feature extraction.

3.4.1 Data Alignment and Re-Sampling

To align data streams with differing sampling rates, as shown in Table 3.2, from Cosinuss° and Garmin sensors, each device's data was resampled to a consistent frequency: 100 Hz for Cosinuss° and 20 Hz for Garmin. For example for Garmin device, lower-frequency signals like heart rate (1 Hz) were upsampled to 20 Hz to match its accelerometer and gyroscope readings. This alignment was achieved through forward-filling, ensuring continuous and consistent data points without altering the underlying sensor signals. By standardizing sampling rates across all features, this process maintained data integrity and provided a synchronized dataset for further analysis.

3.4.2 Data Cleaning

Data cleaning is essential for improving data quality by identifying and handling inconsistencies, errors, and anomalies that could distort analysis or model training. For this dataset, outlier detection focused on removing sensor errors, particularly saturated values.

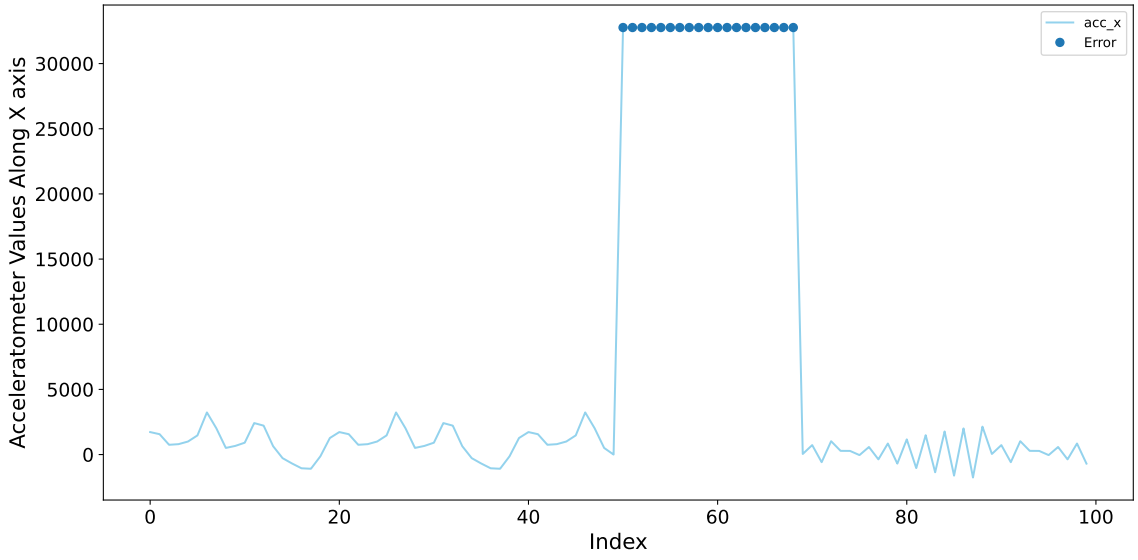


Figure 3.7: Removing Sensor Errors for Garmin Smart Watch.

As shown in Figure 3.7, the Garmin smart watch occasionally produced abnormally high accelerometer and gyroscope readings, reaching a maximum value of 32767. This saturation error indicates that the sensor exceeded its recording capacity, likely due to hardware limitations or transmission issues. Such unrealistic outliers were flagged and removed to retain only meaningful data. By filtering out these extreme values, the dataset

better represents true activity signals, ensuring accuracy in subsequent analysis and model training.

3.4.3 Filtering and Noise Removal

Raw sensor signals often contain noise due to the limitations of wearable devices, which can degrade the performance of machine learning models. This noise may stem from sensor inaccuracies, environmental interference, or irrelevant body movements. To address these issues, a Butterworth bandpass filter was applied to the accelerometer data, as illustrated in Figure 3.8.

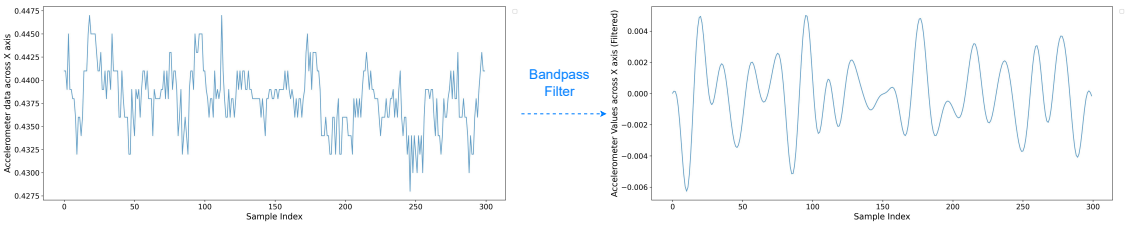


Figure 3.8: Comparison of Raw and Preprocessed Accelerometer Data for Cosinuss° Sensor.

The left side of Figure 3.8 shows the raw accelerometer data, which is prone to low-frequency drift and high-frequency noise. These artifacts can obscure meaningful movement signals, such as walking or running. In contrast, the right side displays the filtered accelerometer data after applying a bandpass filter with a frequency range of 2 to 8 Hz. The filtering process, described by the transfer function in Equation 3.1, selectively retains the relevant frequency components while attenuating noise:

$$H(s) = \frac{s^2}{s^2 + \left(\frac{\omega_2 - \omega_1}{Q}\right) s + \omega_1 \omega_2}, \tag{3.1}$$

where, ω_1 and ω_2 are the lower and upper cutoff angular frequencies, and Q is the quality factor. This filtering step significantly improves data quality by removing noise outside the relevant range of human activity, ensuring that only meaningful accelerometer signals are preserved for subsequent machine learning analysis.

The first 300 samples of filtered accelerometer data across all three axes for each activity, shown in Figure 3.9, reveal distinct movement patterns. Periodic activities such as walking and jogging are clearly identifiable, while sitting produces stable, low-motion signals. Although the patterns for ascending and descending are more complex, they remain distinguishable, demonstrating the effectiveness of the preprocessing in enhancing activity-specific signals.

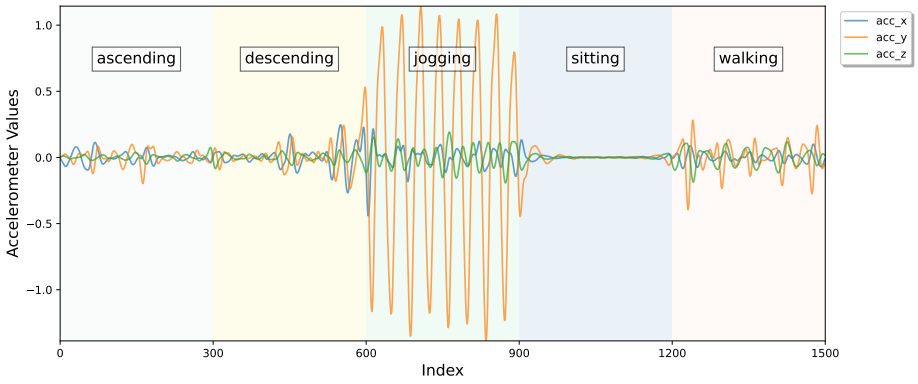


Figure 3.9: Preprocessed Accelerometer Data for Cosinuss° Sensor for different activities.

3.4.4 Simple Moving Average

The moving average is a simple yet effective technique for smoothing time series data by averaging consecutive values within a defined window, which helps reduce random noise and highlight underlying trends. Given the fluctuations in heart rate data, a moving average filter with a window size of 3 was applied to smooth the signal. This technique averages each data point with its two neighboring values, reducing rapid fluctuations while maintaining the overall trend, thereby making the signal more suitable for further analysis and feature extraction. The simple moving average at time t with a window size N is defined as:

$$MA_t = \frac{1}{N} \sum_{i=0}^{N-1} x_{t-i}, \tag{3.2}$$

where MA_t represents the moving average at time t, N is the window size, and x_{t-i} is the data value at time $t - i$. With $N = 3$, this moving average effectively smooths short-term fluctuations, enhancing the heart rate signal for more robust analysis and feature extraction.

3.4.5 Scaling and Transformation

Following filtering and outlier detection, the sensor signals are standardized using a StandardScaler to ensure that each feature contributes equally to the learning process. This transformation normalizes the data by adjusting it to have a mean of zero and unit variance, thereby reducing any bias caused by differing feature scales [25]. The standard scaling of a feature x is calculated using Equation 3.3,

$$z = \frac{x - \mu}{\sigma}, \tag{3.3}$$

where z is the standardized (scaled) value, x is the original feature value, μ is the mean of the feature, and σ is the standard deviation of the feature. Standardization using this formula enables consistent feature representation across the dataset, facilitating balanced learning and improving model performance by ensuring that all features are on a comparable scale. This step is especially critical for models that are sensitive to input magnitudes, as it helps stabilize the training process and optimizes the effectiveness of the learning algorithm.

3.5 Feature Engineering

Once the sensor data is preprocessed, feature engineering is applied to transform the raw data into meaningful representations suitable for classical machine learning models that heavily rely on handcrafted features [57]. This process involves extracting both time-domain and frequency-domain features. Time-domain features capture statistical properties within segmented windows, helping to distinguish between activities based on patterns like central tendency, variability, and signal intensity. Frequency-domain features, extracted using the Fast Fourier Transform (FFT), reveal periodic patterns and energy distribution across frequency bands, adding further insights into the characteristics of each activity.

3.5.1 Time-Domain Feature Extraction

Time-domain features are computed from segmented windows of sensor data to capture statistical characteristics that distinguish between activities. A total of 143 time-domain features were extracted from the Cosinuss° data, and 274 from the Garmin data. Key features include the mean and standard deviation, representing the central tendency and variability, and additional measures such as variance (dispersion), sum (total segment value), magnitude (Euclidean norm), and motion variation, which captures movement fluctuations across successive readings. Features like mean absolute deviation (MAD), maximum and minimum values, root mean square (RMS), signal magnitude area (SMA), energy, and zero-crossing rate (ZCR) provide insights into the signal's intensity and fluctuations. Skewness and kurtosis describe the shape of the data distribution, while autoregressive (AR) coefficients capture temporal dependencies within the signal. Correlation coefficients and inter-axis angles add spatial context, while the interquartile range (IQR) and empirical cumulative distribution function (ECDF) percentiles summarize data spread and distribution characteristics.

3.5.2 Frequency-Domain Feature Extraction

Frequency-domain features are extracted using the Fast Fourier Transform (FFT) to capture periodic patterns in sensor data. From the Cosinuss° device, 45 frequency-domain

features were extracted, and 90 from the Garmin data. Key features include the maximum frequency index, identifying the dominant frequency in the FFT spectrum, and mean frequency, representing the weighted average of frequencies. Spectral skewness and kurtosis describe the shape of the frequency distribution, while spectral entropy quantifies the randomness in the signal. Additionally, band energy is calculated by dividing the spectrum into predefined frequency bins, providing insight into the distribution of energy across frequencies.

3.6 Recursive Feature Elimination (RFE)

Feature selection is crucial in machine learning, especially for high-dimensional datasets, as it enhances model interpretability and mitigates overfitting [18]. In this study, Recursive Feature Elimination (RFE) is used to identify significant features for classification tasks across the Cosinuss° and Garmin datasets. RFE iteratively removes the least important features, refitting the model until the most relevant subset remains [18].

Using RFE, 100 of the 188 time and frequency-domain features were selected from the Cosinuss° dataset, while 200 of 364 features were chosen from the Garmin dataset, enhancing contextual information for Human Activity Recognition. The RFE process begins by splitting the data into 70% training and 30% testing subsets, with a Random Forest Classifier as the base estimator, as shown in Figure 3.10. RFE ranks features by importance, eliminates the least impactful, and refits the model iteratively until only the top features remain. In fixed mode, RFE retains a specified number of features, whereas in variable mode, it evaluates subsets from 1 to n, selecting the subset with the highest classification accuracy.

The feature subset at iteration i is:

$$\mathcal{F}_i = \{f_1, f_2, \ldots, f_n\}, \quad i = 1, 2, \ldots, n. \tag{3.4}$$

Feature ranking is computed as:

$$\mathcal{R}(f_i) = \text{Rank}(f_1, f_2, \ldots, f_n), \tag{3.5}$$

with the model trained on the selected features:

$$\hat{y} = \text{Model}(X_{\text{train}}|\mathcal{F}_n), \tag{3.6}$$

where X_{train} is the training data and $\mathcal{F}_n$ is the feature subset. Classification accuracy is computed on the test set as:

$$\text{Accuracy} = \frac{\sum_{i=1}^{m} I(\hat{y}_i = y_i)}{m}, \tag{3.7}$$

where m is the number of test samples. Tracked features and accuracy at each iteration are:

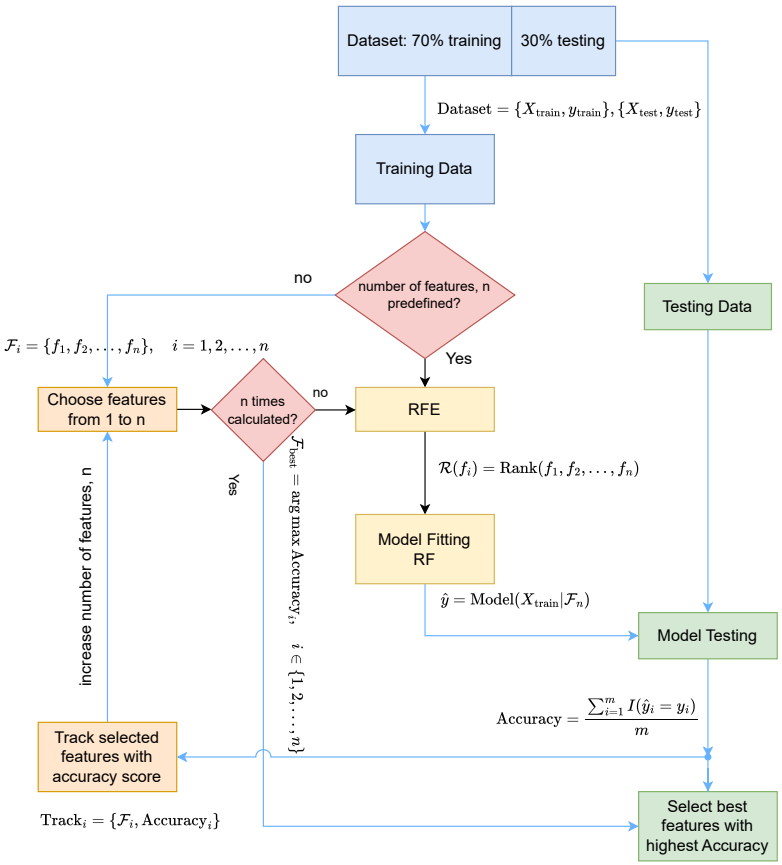


Figure 3.10: Process flow diagram of Recursive Feature Elimination (RFE) for feature selection.

$$\text{Tracked}_i = \{\mathcal{F}_i, \text{Accuracy}_i\}. \tag{3.8}$$

In *variable* mode, the optimal subset $\mathcal{F}_{\text{best}}$ is selected based on the highest accuracy:

$$\mathcal{F}_{\text{best}} = \arg\max \text{Accuracy}_i, \quad i \in \{1, 2, \ldots, n\}. \tag{3.9}$$

By applying RFE, the most relevant features are identified, improving classification performance while optimizing the computational efficiency for both datasets.

3.7 Windowing and Segmentation

In this experimental design, after preprocessing the sensor data was segmented into fixed-size, overlapping time windows to optimize training for machine learning and deep learn-

ing models. A consistent 1 second window with a 90% overlap was applied to both Cosinuss° and Garmin data, allowing for finer temporal resolution and ensuring sufficient training samples. For Cosinuss° data (sampled at 100 Hz), each 1 second window contained 100 samples with a 10 sample step size, achieving 90% overlap. Similarly, Garmin data (sampled at 20 Hz) used a 1 second window with 20 samples and a 2 sample step size. This overlap facilitates smooth transitions between activity segments, enhancing the model's ability to capture detailed temporal patterns. This segmentation approach was tailored for deep learning models, aiming to generate at least 10,000 samples, thereby providing ample data to capture activity-specific features effectively.

3.8 Data Splitting and Cross-Validation

In this experimental design, the dataset was split into training, validation, and testing sets. First, 20% of the entire dataset was set aside exclusively as the test set for final evaluation, providing an unbiased assessment of model generalization after training. The remaining 80% was split into training and validation sets in an 80:20 ratio, ensuring sufficient data for model training while reserving a validation set for hyperparameter tuning and performance monitoring.

For machine learning models, 5-fold cross-validation was applied on the training set, where the data was divided into five subsets, each used once for validation while the others were used for training. This approach ensured robust hyperparameter tuning and minimized overfitting risks. For deep learning models, cross-validation was not utilized due to computational demands. Instead, these models were trained on the training set and validated on the 16% validation set. After optimal tuning, both model types were evaluated on the 20% test set. This data-splitting strategy was applied consistently across subject-specific and combined evaluations, ensuring balanced model assessments.

3.9 Evaluation Metrics

The performance of the models was assessed using standard evaluation metrics commonly used in classification tasks. These metrics include accuracy, precision, recall, and F1 score, each defined as follows.

Accuracy (Acc) measures the overall effectiveness of the model, representing the ratio of correctly predicted activities to the total number of predictions. It is defined as

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}, \tag{3.10}$$

where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives.

Precision (P) quantifies the model's ability to correctly identify positive instances. It is calculated as the ratio of true positives (TP) to the total number of predicted positives

($TP + FP$):

$$P = \frac{TP}{TP + FP}. \tag{3.11}$$

A higher precision indicates a lower rate of false positives.

Recall (R), also known as sensitivity, measures the model's effectiveness in identifying all relevant instances. It is defined as the ratio of true positives (TP) to the total actual positives ($TP + FN$):

$$R = \frac{TP}{TP + FN}. \tag{3.12}$$

High recall indicates a lower rate of false negatives.

F1 score (F_1) is the harmonic mean of precision and recall, providing a balanced metric when there is an uneven class distribution. It is given by

$$F_1 = 2 \times \frac{P \times R}{P + R}. \tag{3.13}$$

A higher F1 score indicates good performance in terms of both precision and recall.

For multi-class classification tasks, two F1 score variants, macro and weighted, were calculated. The **Macro F1 score** averages the F1 scores across all classes, treating each class equally. It is defined as

$$F_{1\ macro} = \frac{1}{C} \sum_{c=1}^{C} F_{1c} = \frac{1}{C} \sum_{c=1}^{C} 2 \times \frac{\text{Precision}_c \times \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c}, \tag{3.14}$$

where C is the total number of classes, and Precision_c and Recall_c are the precision and recall for class c.

The **Weighted F1 score** adjusts for class imbalances by weighting each class's F1 score by its sample size. It is calculated as

$$F_{1\ weighted} = \frac{1}{C} \sum_{c=1}^{C} \frac{N_c}{N} \times F_{1c}, \tag{3.15}$$

where N_c is the number of samples in class c, and N is the total number of samples in the dataset.

The macro F1 score is ideal for balanced datasets, while the weighted F1 score better represents performance on imbalanced datasets by giving more weight to larger classes. In this study, the weighted F1 score is used as the primary evaluation metric due to class imbalance.

3.10 Results and Discussion

This section provides a detailed analysis of machine learning (ML) and deep learning (DL) model performance on the Cosinuss° and Garmin datasets, emphasizing the comparative effectiveness of motion-only (M) features versus combined motion and physiological

(M+P) features for human activity classification. The impact of incorporating physiological data on the accuracy of classifying activities, ranging from sitting and walking to more complex actions like ascending and descending stairs, is assessed across both datasets and models.

In total, four machine learning models (KNN, Random Forest, SVM, and XGBoost) were trained and evaluated on both datasets using motion-only and motion plus physiological features, spanning 9 distinct configurations (8 individual subjects plus a combined dataset encompassing all subjects). This resulted in 144 machine learning models being evaluated. Additionally, 3 deep learning models (LSTM, ConvLSTM, and Transformer) were trained on both datasets, again using both motion-only and motion plus plysiological (M+P) features, across the same 9 settings, leading to the evaluation of 108 deep learning models.

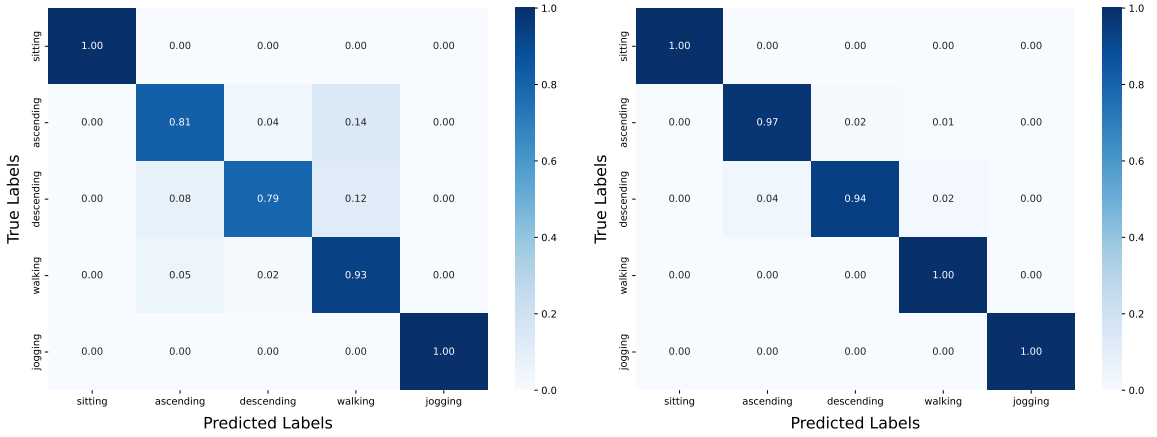
Overall, the total number of model evaluations reached 252, covering both single-subject and multi-subject scenarios. This thorough evaluation highlights the impact of adding physiological data and demonstrates the varying performance of different machine learning and deep learning models when classifying both simple and complex human activities.

3.10.1 Cosinuss° Results with Machine Learning

Table 3.3: Machine Learning Model Performance on Cosinuss° Dataset.

Subject	KNN		Random Forest		SVM		XGBoost	
	M	M+P	M	M+P	M	M+P	M	M+P
S1	0.9783	0.9921	0.9925	0.9992	0.9865	0.9893	0.9964	1.0000
S2	0.9779	0.9963	0.9927	1.0000	0.9825	0.9996	0.9935	1.0000
S3	0.9737	0.9988	0.9928	0.9992	0.9794	0.9992	9956	1.0000
S4	0.9724	0.9889	0.9910	0.9984	0.9712	0.9942	0.9955	1.0000
S5	0.9229	0.9807	0.9636	0.9994	0.9077	1.0000	0.9819	0.9994
S6	0.9292	0.9883	0.9745	0.9991	0.9154	0.9922	0.9858	1.0000
S7	0.9550	0.9871	0.9849	0.9956	0.9759	0.9908	0.9939	0.9982
S8	0.9716	0.9869	0.9859	0.9980	0.9763	0.9965	0.9915	0.9985
Multi-Subject Dataset	0.9597	0.9907	0.9814	0.9990	0.9417	0.9834	0.9891	0.9996

Table 3.3 shows the performance of four machine learning models (KNN, Random Forest, SVM, and XGBoost) on Cosinuss° dataset using both motion (M) and motion plus physiological (M+P) features across individual subject and multi-subject scenario. Incorporating physiological (M+P) features significantly improves classification performance, with near-perfect weighted F1 scores, particularly for Random Forest and XGBoost reaching 1.0000 in the M+P settings. Even without physiological features (M-only), the results are also very satisfactory, indicating that motion features alone provide strong classification performance across models. The multi-subject dataset also benefits from motion plus physiological features, where Random Forest and XGBoost achieve the highest results, highlighting the value of combining motion and physiological data for enhanced activity recognition.



(a) KNN Confusion Matrix using motion (M) data.

(b) KNN Confusion Matrix using motion and physiological (M+P) data.

Figure 3.11: Comparison of KNN Confusion Matrices for subject S6 using motion (M) data vs. motion plus physiological (M+P) data from Cosinuss° device.

Figure 3.11 shows the KNN confusion matrices for subject S6 using motion (M) data (Figure 3.11a) and motion plus physiological (M+P) (Figure 3.11b) features. In the motion-only setting, KNN performs well for activities like sitting and jogging with perfect classification, but struggles with more complex activities such as ascending and descending, with only 81% and 79% accuracy, respectively, and notable confusion between these two activities. When physiological data is added with motion data, the model's performance improves significantly, with ascending accuracy increasing to 97% and descending to 94%, while maintaining perfect accuracy for sitting and jogging. These results demonstrate that incorporating physiological features enhances the model's ability to classify complex activities, reducing misclassifications and improving overall accuracy.

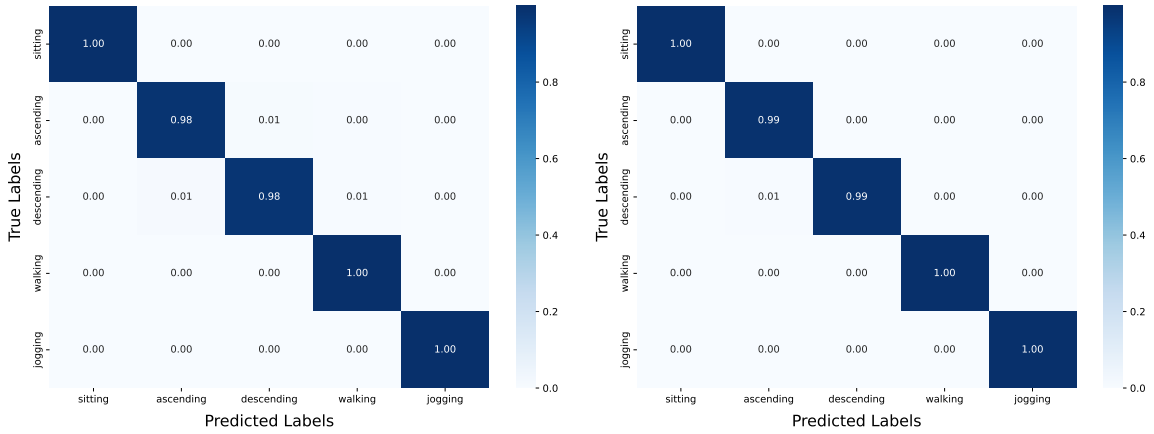
3.10.2 Garmin Results with Machine Learning

Table 3.4 shows how four machine learning models (KNN, Random Forest, SVM, and XGBoost) perform on Garmin data, utilizing motion (M) and motion plus physiological (M+P) features in both individual and multi-subject scenarios. Incorporating physiological (M+P) data slightly improves or maintains near-perfect classification performance across all subjects and models, with Random Forest and XGBoost consistently achieving accuracies close to 1.0000. KNN and SVM also show high performance, with slight improvements when adding physiological features, particularly for more complex activities. The multi-subject dataset follows the same trend, emphasizing that combining physiological data with motion features enhances classification accuracy across all models. Despite the addition of physiological features, even the motion only results remain highly satisfactory, demon-

strating that motion features alone provide solid classification performance with Garmin dataset.

Table 3.4: Machine Learning Model Performance on Garmin Dataset.

Subject	KNN		Random Forest		SVM		XGBoost	
	M	M+P	M	M+P	M	M+P	M	M+P
S1	0.9968	0.9972	0.9984	0.9984	0.9984	0.9984	1.0000	0.9996
S2	0.9980	0.9988	0.9972	1.0000	0.9944	0.9964	0.9988	1.0000
S3	0.9996	0.9996	0.9988	1.0000	0.9980	1.0000	0.9996	1.0000
S4	0.9996	0.9992	0.9988	0.9992	0.9975	0.9992	0.9996	0.9996
S5	0.9981	0.9981	0.9994	0.9994	0.9994	0.9994	0.9994	0.9994
S6	0.9993	0.9993	0.9996	0.9993	0.9996	0.9993	0.9993	0.9993
S7	0.9996	1.0000	0.9996	1.0000	0.9991	0.9996	0.9996	0.9991
S8	0.9986	0.9995	0.9995	0.9995	0.9995	0.9976	0.9991	0.9995
Multi-Subject Dataset	0.9979	0.9987	0.9991	0.9994	0.9942	0.9968	0.9994	0.9994



(a) SVM Confusion Matrix for Multi-Subject using motion (M) data.

(b) SVM Confusion Matrix for Multi-Subject using motion and physiological (M+P) data.

Figure 3.12: Comparison of SVM Confusion Matrices for Multi-Subject using motion (M) data vs. motion plus physiological (M+P) data from Garmin device.

Figure 3.12 shows the confusion matrices for SVM on the Garmin dataset in a multi-subject context using motion (M) (Figure 3.12a) and motion plus physiological (M+P) features (Figure 3.12b). In the motion-only (M) setting, SVM performs well for sitting, walking, and jogging with perfect classification but shows some confusion between ascending and descending activities, with both achieving 98% accuracy. When physiological data is added, the model's performance improves, increasing ascending and descending accuracy to 99%, while maintaining perfect classification for sitting, walking, and jogging.

These results highlight the positive impact of adding physiological data, which enhances SVM's ability to differentiate between more complex activities, particularly ascending and descending, and reduces misclassification in the multi-subject dataset.

3.10.3 Cosinuss° Results with Deep Learning

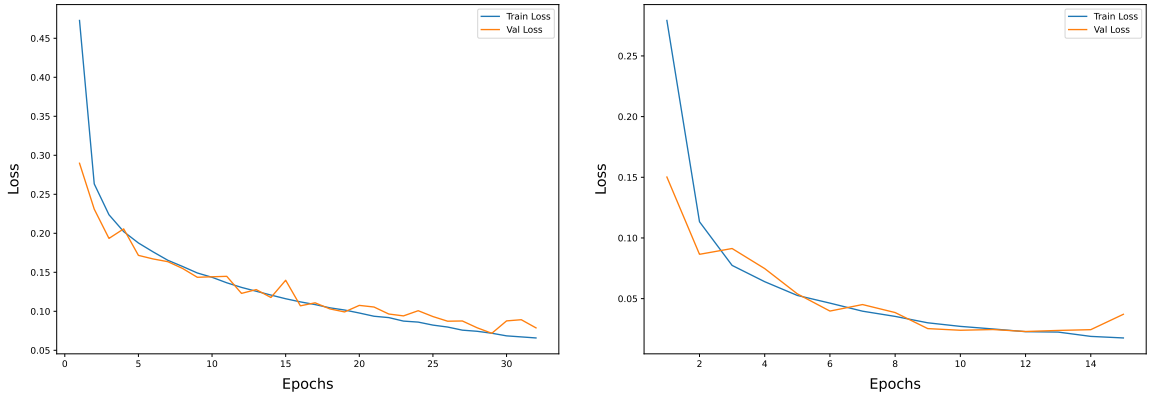
Table 3.5: Deep Learning Model Performance on Cosinuss° Dataset.

Subject	LSTM		ConvLSTM		Transformer	
	M	M+P	M	M+P	M	M+P
S1	0.7158	0.9718	0.9570	0.9932	0.9881	0.9928
S2	0.7250	0.9670	0.9728	0.9992	0.9793	0.9980
S3	0.7009	0.9886	0.8443	0.9778	0.9652	0.9980
S4	0.8147	0.9534	0.7159	0.9288	0.9772	0.9921
S5	0.7776	0.9917	0.7297	0.9817	0.9295	0.9955
S6	0.7014	0.9765	0.7106	0.9905	0.9592	0.9926
S7	0.7167	0.9519	0.7063	0.9202	0.9881	0.9819
S8	0.7128	0.9521	0.8718	0.9833	0.9614	0.9921
Multi-Subject Dataset	0.7678	0.9306	0.7895	0.9462	0.9754	0.9945

Table 3.5 shows the performance of deep learning models (LSTM, ConvLSTM, and Transformer) on the Cosinuss° dataset using motion-only (M) and motion plus physiological (M+P) features. Adding physiological features (M+P) significantly boosts performance across all models, with Transformers achieving the highest scores, consistently exceeding 0.99 in weighted F1 score. LSTM models demonstrate the largest improvement with M+P features, averaging a 30.92% increase over the M-only setting. ConvLSTM also sees steady gains with M+P features. In the multi-subject dataset, the combination of M+P features results in nearly perfect performance, particularly with Transformers. This highlights the significant benefit of incorporating physiological data to improve classification accuracy across various models and subjects.

Figure 3.13 illustrates the loss curves for the Transformer model on the multi-subject Cosinuss° dataset using motion (M) (Figure 3.13a) and motion plus physiological (M+P) (Figure 3.13b) features. In both cases, the model shows a consistent decrease in training and validation loss over time, indicating effective learning. However, when physiological features are included (M+P), the initial loss is lower, and the model converges more quickly, achieving better performance in fewer epochs. The close alignment between training and validation loss in both cases suggests that the model generalizes well, but the addition of physiological features helps the model reach lower overall loss and faster convergence, highlighting the benefit of incorporating this additional data.

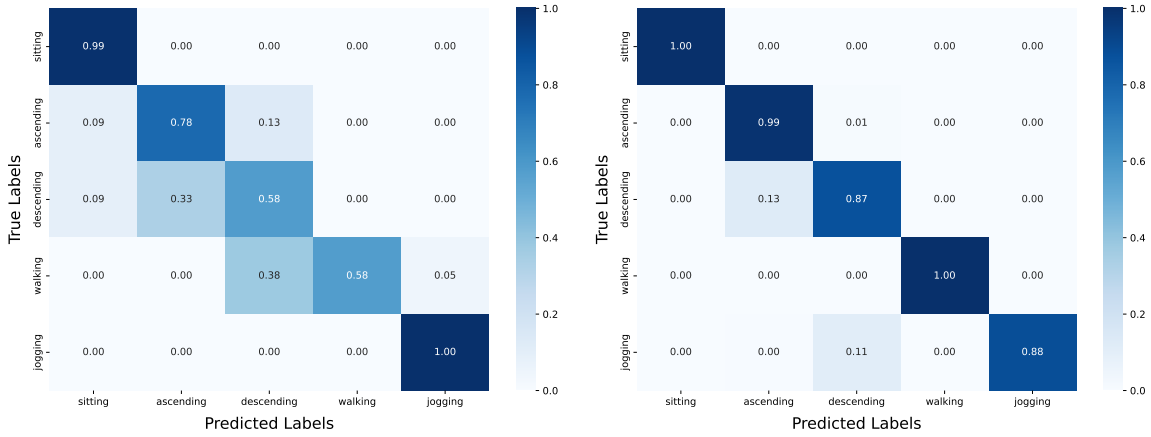
Figure 3.14 presents the LSTM confusion matrices for subject S4 using motion (M) (Figure 3.14a) and motion plus physiological (M+P) (Figure 3.14b) features. In the motion



(a) Transformer Loss Curve for Multi-Subject setup with motion (M) data.

(b) Transformer Loss Curve for Multi-Subject setup with motion and physiological (M+P) data.

Figure 3.13: Loss Curves for Transformer Model in Multi-Subject scenarios: motion (M) vs. motion plus physiological (M+P) data from Cosinuss° Sensor.



(a) Confusion Matrix for LSTM model with motion (M) data.

(b) Confusion Matrix for LSTM model with motion and physiological (M+P) data.

Figure 3.14: Comparison of LSTM Confusion Matrices for subject S4 using motion (M) Data vs. motion plus physiological (M+P) Data from Cosinuss° Sensor.

Only setting, the model struggles with complex activities like ascending, descending, and walking, with significant misclassifications, achieving only 78% accuracy for ascending and 58% for descending and walking. However, sitting and jogging are classified with near-perfect accuracy. When physiological data is combined with motion data (M+P), the

model's performance improves significantly, reaching 99% for ascending and 87% for descending. However, there remains some confusion with jogging, leading to a performance drop. but the inclusion of physiological features leads to better overall classification performance, especially for more complex activities, while maintaining perfect accuracy for sitting.

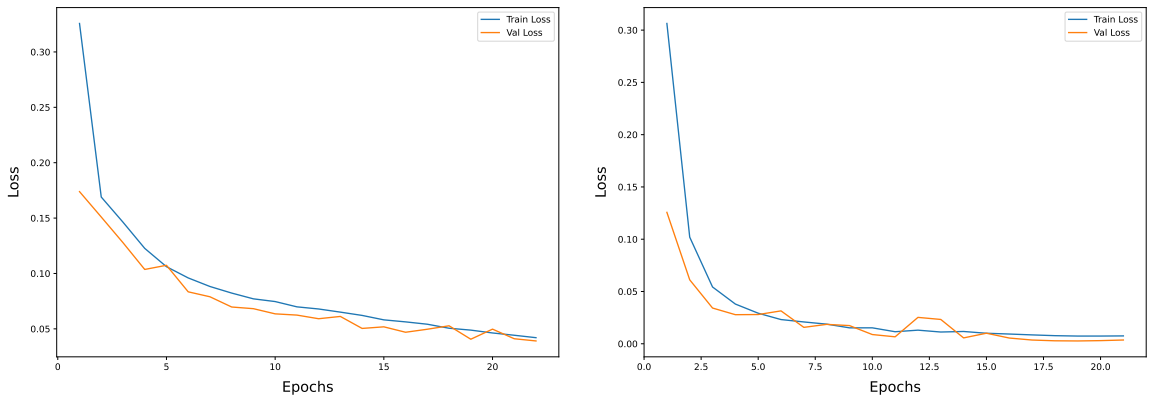
3.10.4 Garmin Results with Deep Learning

Table 3.6: Deep Learning Model Performance on Garmin Dataset.

Subject	LSTM		ConvLSTM		Transformer	
	M	M+P	M	M+P	M	M+P
S1	0.8681	0.9900	0.9960	0.9964	0.9912	0.9936
S2	0.9051	0.9887	0.9525	0.9935	0.9503	0.9960
S3	0.9178	0.9964	0.9169	0.9988	0.9244	0.9912
S4	0.8327	0.9955	0.8327	0.9988	0.9434	0.9934
S5	0.9852	0.9865	0.9623	0.9879	0.9871	0.9917
S6	0.9978	0.9911	0.9970	0.9922	0.9966	0.9974
S7	0.9777	0.9974	0.9882	0.9974	0.9917	0.9978
S8	0.9724	0.9962	0.9560	0.9981	0.9562	0.9972
Multi-Subject Dataset	0.9626	0.9927	0.9731	0.9950	0.9818	0.9918

Table 3.6 shows the performance of deep learning models (LSTM, ConvLSTM, and Transformer) on Garmin data, comparing the use of motion-only (M) features with motion plus physiological (M+P) features across multiple subjects. The inclusion of physiological features with motion consistently enhances classification performance across all models, with ConvLSTM and Transformer achieving near-perfect accuracy in most cases. Transformer models, in particular, perform exceptionally well, often reaching accuracy scores close to 1.0000. LSTM models also show significant improvement when physiological data is added, although their overall performance tends to be slightly lower than ConvLSTM and Transformer. The multi-subject dataset follows a similar pattern, where M+P provides the best results, especially for ConvLSTM and Transformer models. Though with motion-only data, the models deliver strong results, particularly ConvLSTM and Transformer, but the addition of physiological features clearly boosts accuracy across all subjects.

Figure 3.15 illustrates the loss curves for the ConvLSTM model on the multi-subject dataset using motion (M) (Figure 3.15a) and motion plus physiological (M+P) features (Figure 3.15b). In both cases, the training and validation losses decrease steadily over the epochs, demonstrating effective learning. However, when physiological features are added (M+P), the model converges faster and reaches lower overall loss values more quickly. The gap between training and validation loss remains small in both scenarios, indicating good generalization and minimal overfitting. The addition of physiological data

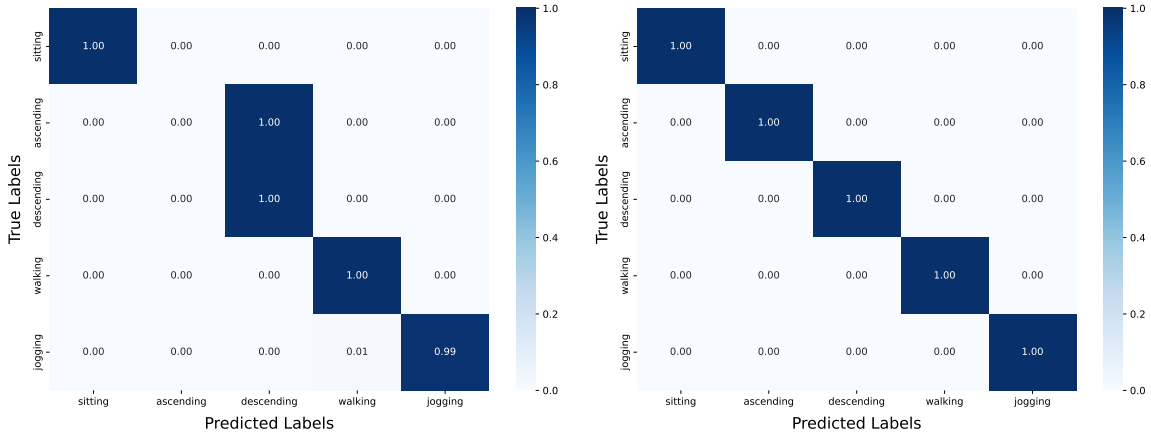


(a) ConvLSTM Loss Curve for Multi-Subject setup with motion (M) data.

(b) ConvLSTM Loss Curve for Multi-Subject setup with motion and physiological (M+P) data.

Figure 3.15: Comparison of Loss Curves for ConvLSTM model in Multi-Subject setup using Garmin data: motion (M) vs. motion plus physiological (M+P) data.

helps the model achieve better performance in fewer epochs, highlighting the benefit of incorporating this information to improve classification accuracy.



(a) Confusion Matrix for ConvLSTM model with motion (M) data.

(b) Confusion Matrix for ConvLSTM model with motion and physiological (M+P) data.

Figure 3.16: Comparison of ConvLSTM Confusion Matrices for subject S4 using Garmin Data: motion (M) vs. motion plus physiological (M+P) data.

Figure 3.16 illustrates the confusion matrix for the ConvLSTM model applied to sub-

ject S4, comparing the model's performance using motion-only (Figure 3.16a) versus motion plus physiological (M+P) features (Figure 3.16b). In the motion-only scenario, the model performs well in classifying activities like sitting and walking, but it faces challenges distinguishing between ascending and descending, with substantial misclassifications of ascending as descending. Jogging, however, is classified with 99% accuracy. When physiological data is incorporated (M+P), the model's performance improves dramatically, achieving perfect classification for all activities, including the previously challenging ascending and descending tasks. These results emphasize the importance of incorporating physiological features, significantly reducing misclassifications and improving the model's ability to distinguish between complex activities like ascending and descending.

3.10.5 Performance Analysis for Individual Subjects

This section provides a detailed analysis of the performance of classical machine learning and deep learning models on an individual subject basis. The weighted F1 score is used as the primary evaluation metric to assess each model's performance on individual subjects, incorporating both motion-only and motion plus physiological features.

3.10.5.1 Performance Evaluation for Individual Subjects on Cosinuss° Dataset

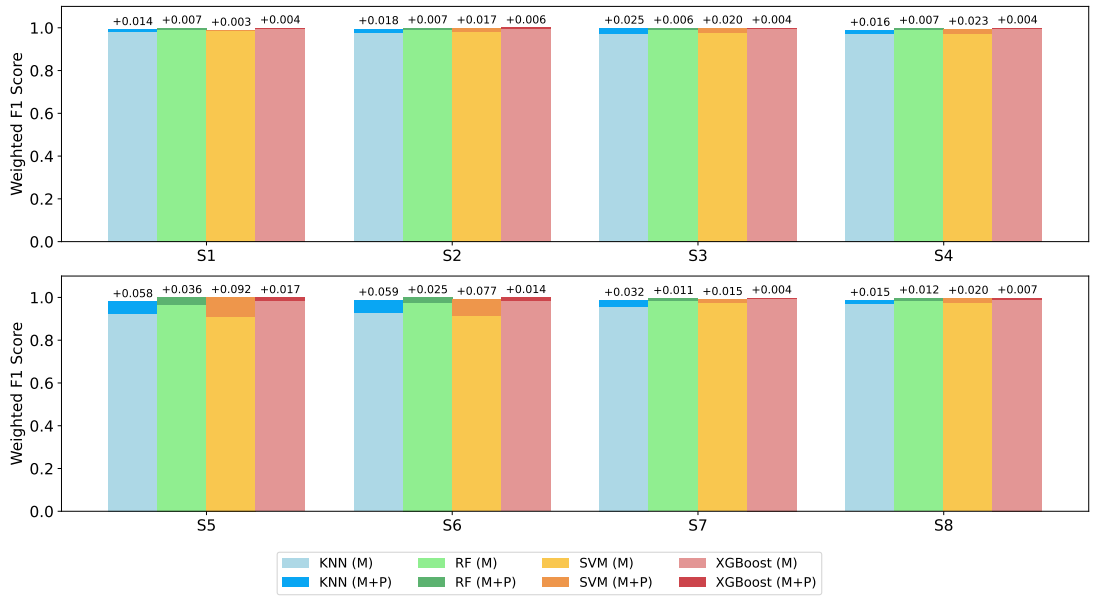


Figure 3.17: Performance of a Deep Learning models on the Cosinuss° dataset for Individual Subjects.

Figure 3.17 presents the weighted F1 scores of machine learning models namely KNN,

Random Forest (RF), SVM, and XGBoost evaluated on individual subjects (S1-S8) using the Cosinuss° dataset. Each subject's performance is displayed in two settings: motion-only (M) data and motion plus physiological (M+P) data. The incremental improvement in the weighted F1 score due to the inclusion of physiological data is shown as a numeric value above the motion bar for each model and subject. The figure demonstrates that for most subjects, combining physiological data with motion data enhances the model performance, as evidenced by the positive increments in F1 scores across the models.

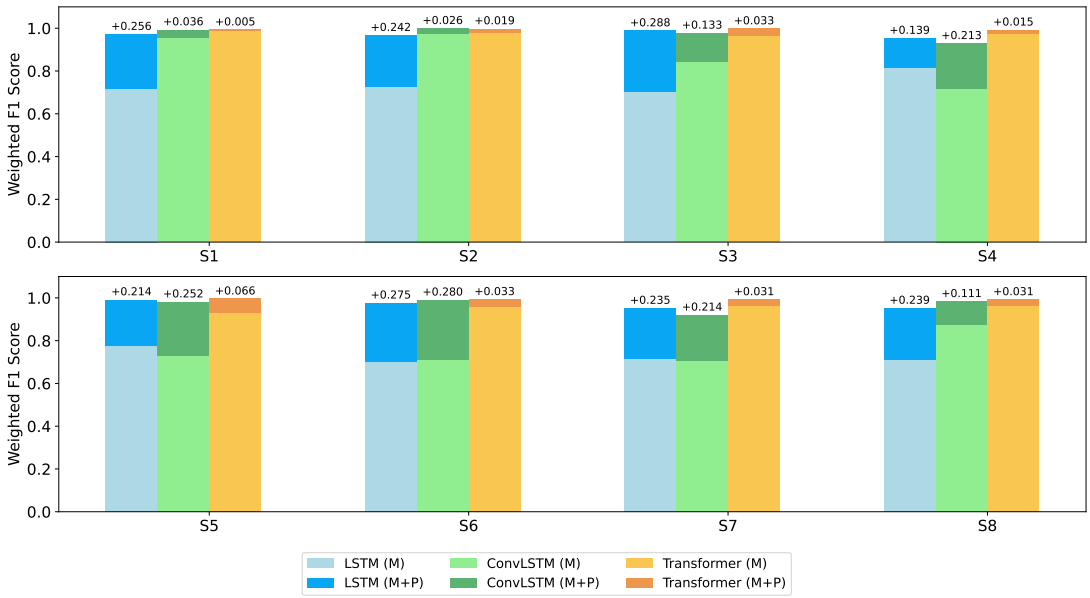


Figure 3.18: Performance of Machine Learning models on the Garmin Dataset for Individual Subjects.

Figure 3.18 illustrates the comparison of weighted F1 scores for three deep learning models: LSTM, ConvLSTM, and Transformer, evaluated across all subjects (S1-S8) using the Cosinuss° dataset. Results for each subject are presented for two conditions: one that utilizes only motion-only (M) data and another that includes both motion plus physiological (M+P) data. For the majority of subjects, a significant enhancement is observed, particularly with the LSTM and ConvLSTM models, where the increase from integrating physiological data is notably greater. This underscores the advantage of merging physiological data with motion data to boost the classification accuracy of the models.

3.10.5.2 Performance Evaluation for Individual Subjects on Garmin Dataset

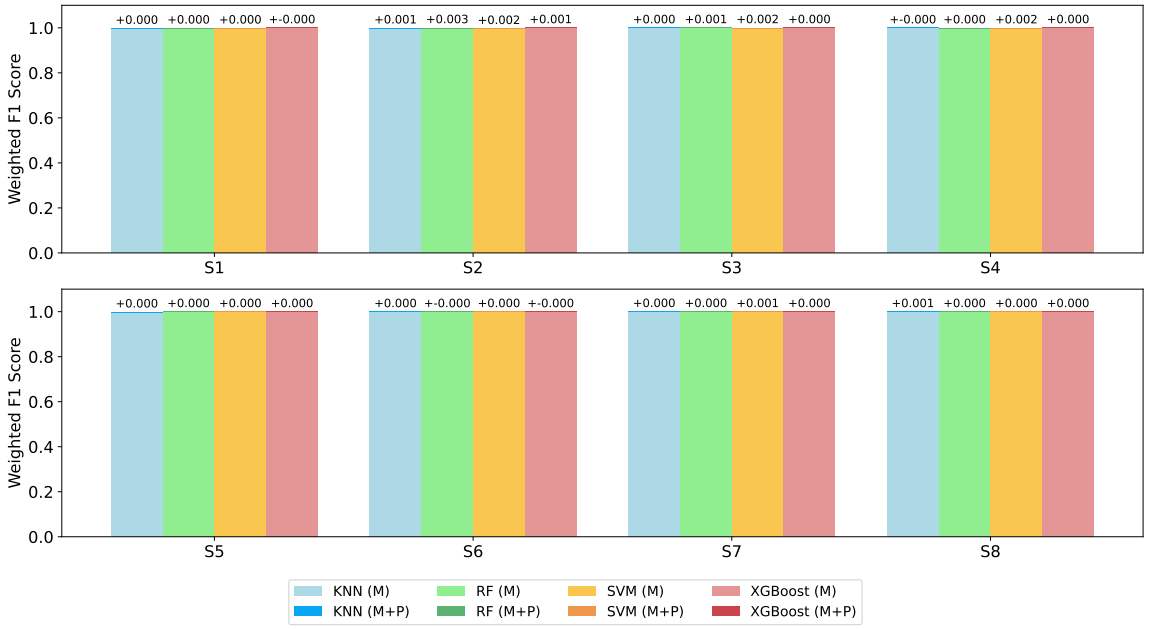


Figure 3.19: Performance of Machine Learning models on the Garmin Dataset for Individual Subjects.

Figure 3.19 shows the performance of the machine learning models KNN, Random Forest (RF), SVM, and XGBoost on individual subjects (S1-S8) using the Garmin dataset. Interestingly, the inclusion of physiological data (M+P) alongside motion data does not result in any significant improvement across most subjects, as evidenced by the marginal or non-existent weighted F1 score changes. In fact, for the majority of the subjects, the performance remains almost identical between the two feature sets (M and M+P), with increments close to zero. This observation suggests that the physiological data collected from the Garmin device has limited impact on enhancing the classification accuracy for these models, in contrast to the improvements seen with the Cosinuss° dataset.

Figure 3.20 illustrates the weighted F1 scores for the deep learning models, LSTM, ConvLSTM, and Transformer, on individual subjects (S1-S8) using the Garmin dataset. The results are shown for both motion (M) data only and motion plus physiological (M+P) data. Similar to the machine learning models, the improvement in weighted F1 score with the inclusion of physiological data is relatively minor for most subjects. Notable improvements are observed for certain models on subjects S1, S2, S3 and S4, where the LSTM and ConvLSTM models exhibit some performance gains, particularly when physiological data is added, as indicated by the positive increments above the bars. However, for the majority of the subjects, the difference between M and M+P configurations remains small, suggest-

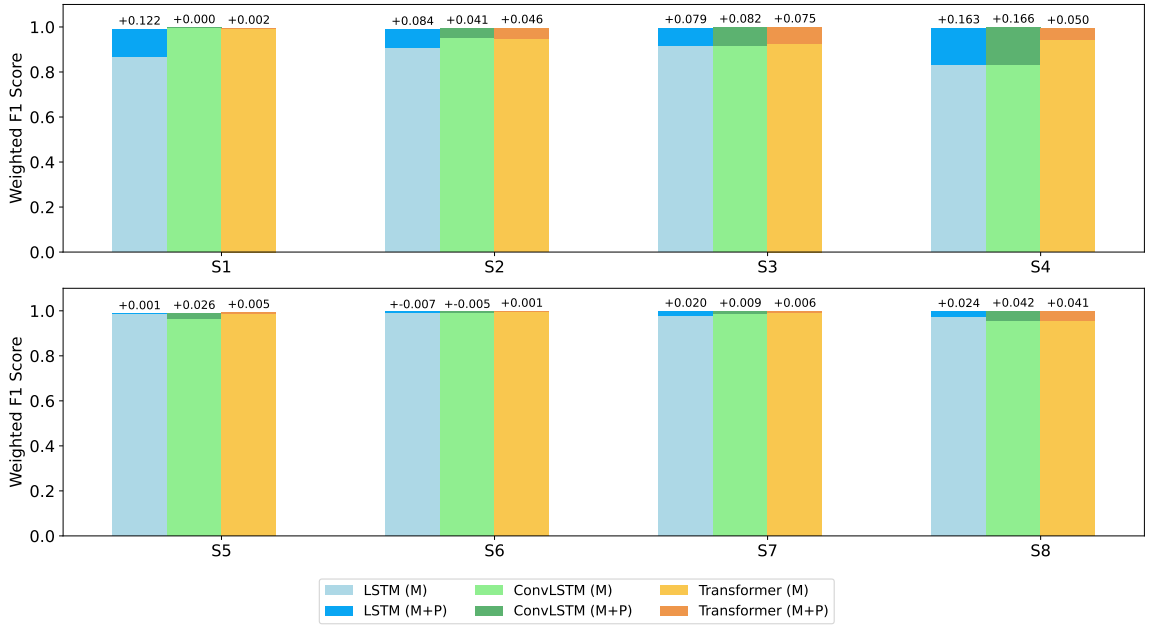


Figure 3.20: Performance of Deep Learning models on the Garmin Dataset for Individual Subjects.

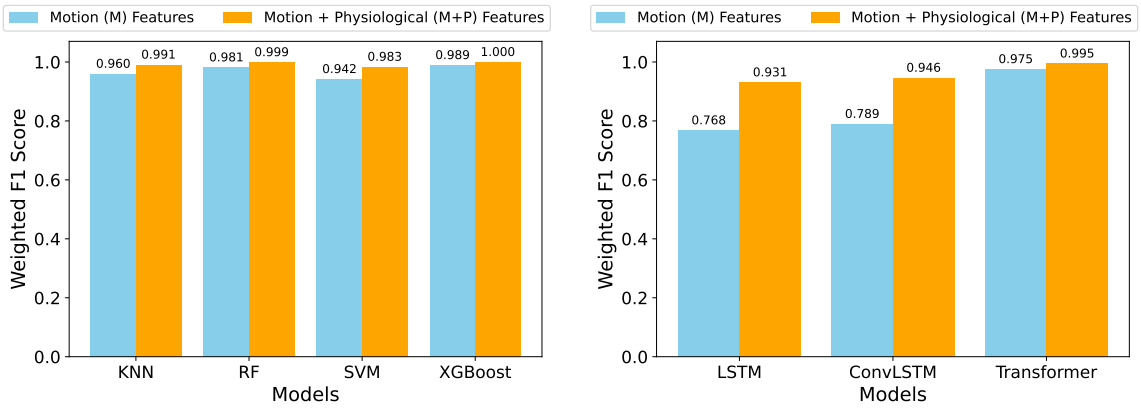
ing that physiological data from the Garmin sensor has limited influence on improving the classification performance of these deep learning models.

3.10.6 Performance Analysis for Multi-Subject

This section examines the combined performance of classical machine learning and deep learning models across multiple subjects, using the Cosinuss° and Garmin datasets. Weighted F1 scores are used to assess each model's generalization effectiveness in multi-subject scenarios, incorporating both motion-only and motion plus physiological features.

3.10.6.1 Performance Evaluation for Multi-Subject on Cosinuss° Dataset

Figure 3.21 compares the performance of Machine Learning (ML) and Deep Learning (DL) models on the multi-subject Cosinuss° dataset, using weighted F1 scores as the metric. In Figure 3.21a, the ML models, KNN, Random Forest, SVM, and XGBoost, show marked improvements when combining motion and physiological (M+P) data. XGBoost reaches the highest score, achieving a perfect F1 score of 1.000 with M+P features, while Random Forest and KNN also perform well at 0.999 and 0.991, respectively. SVM demonstrates a substantial gain, improving from 0.942 (M) to 0.983 (M+P). Similarly, in Figure 3.21b, DL models see notable gains with M+P data: LSTM improves from 0.768 to 0.931, ConvL-



(a) Performance of Machine Learning models on Cosinuss° dataset (Multi-Subject).

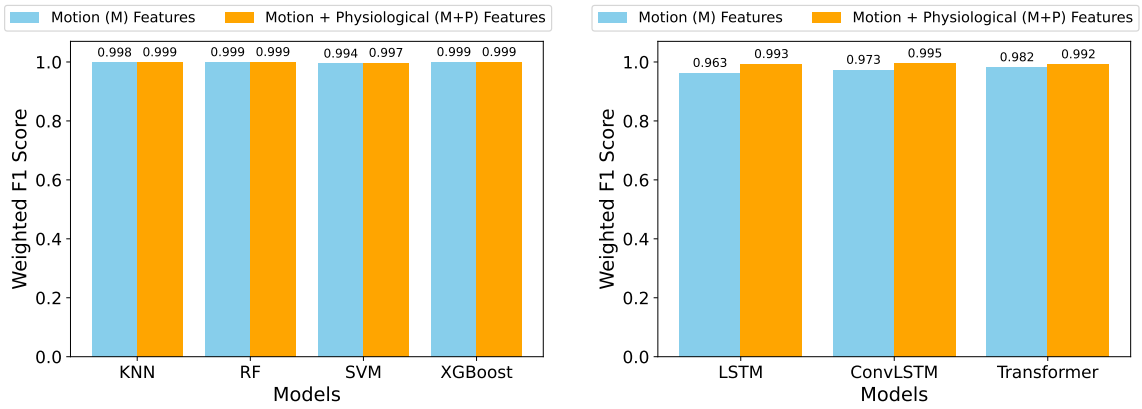
(b) Performance of Deep Learning models on Cosinuss° dataset (Multi-Subject).

Figure 3.21: Comparison of the performance of Machine Learning and Deep Learning models on the Cosinuss° dataset, examining both motion (M) and motion plus physiological (M+P) data across multiple subjects.

STM from 0.789 to 0.946, and Transformer reaches the highest F1 score of 0.995 with M+P data. The Transformer model demonstrates the strongest overall performance among DL models, particularly when utilizing both motion and physiological data. This comparison underscores the positive impact of combining physiological data with motion data across both ML and DL models.

3.10.6.2 Performance Evaluation for Multi-Subject on Garmin Dataset

Figure 3.22 presents the performance comparison of Machine Learning (ML) and Deep Learning (DL) models on the multi-subject Garmin dataset, measured by weighted F1 scores. In Figure 3.22a, the ML models achieve nearly identical results with both motion (M) and motion plus physiological (M+P) data. Both Random Forest and XGBoost obtain F1 scores of 0.999 in both configurations, while KNN scores 0.998 (M) and 0.999 (M+P). SVM shows a slight improvement, from 0.994 with motion data to 0.997 when physiological data is added. In Figure 3.22b, the DL models also demonstrate small gains with the inclusion of physiological data. LSTM improves from 0.963 (M) to 0.993 (M+P), ConvLSTM increases from 0.973 to 0.995, and the Transformer model moves from 0.982 to 0.992. These results highlight the minimal impact of adding physiological data on Garmin's multi-subject setup, as both ML and DL models maintain high performance with motion data alone, with only slight improvements from the added physiological features.

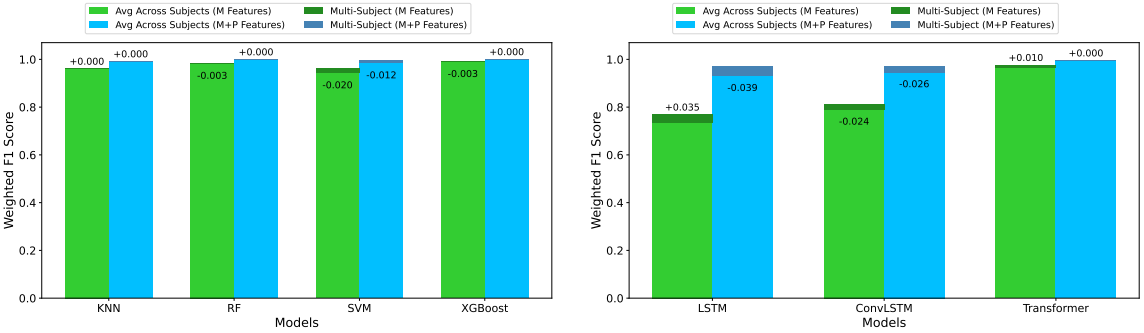


(a) Performance of Machine Learning models on Multi-Subject dataset.

(b) Performance of Deep Learning models on Multi-Subject dataset.

Figure 3.22: Comparison of the performance of Machine Learning and Deep Learning models on the Garmin dataset across multiple subjects, using motion (M) data versus motion plus physiological (M+P) data.

3.10.7 Single-Subject vs. Multi-Subject Model Performance



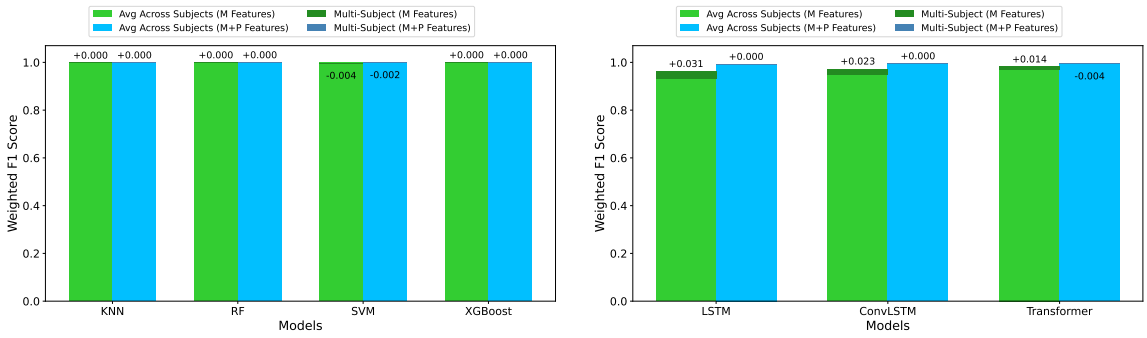
(a) Machine Learning model performance: Avg. Single-Subject vs Multi-Subject.

(b) Deep Learning model performance: Avg. Single-Subject vs Multi-Subject.

Figure 3.23: Comparison of the performance of Machine Learning and Deep Learning models on the Cosinuss° Dataset, using motion (M) features and motion plus physiological (M+P) features.

Figure 3.23 presents a comparison of weighted F1 scores for Machine Learning (ML) and Deep Learning (DL) models on the Cosinuss° dataset, showing average F1 scores for single-subject cases and the combined multi-subject dataset. In the ML models (Figure 3.23a), the addition of physiological data to motion data (M+P) has minimal impact on

KNN performance across both M and M+P settings. For Random Forest (RF) and XGBoost, the average performance in single-subject cases surpasses that of the multi-subject dataset when using motion data (M) alone. However, for SVM, the multi-subject performance is consistently lower than the single-subject average in both the M and M+P settings. Among DL models (Figure 3.23b), LSTM achieves the highest gains from M data, with its score rising from 0.7331 (single-subject average) to 0.7678 in the multi-subject setting, though it slightly drops to 0.9306 with M+P data in the multi-subject case. Conversely, ConvLSTM sees a decline in performance in both single and multi-subject setups with M and M+P features, indicating sensitivity to subject variability. The Transformer model demonstrates outstanding stability, reaching an F1 score of 0.9945 with M+P data in the multi-subject setup, underscoring its strong generalization capability across subjects.



(a) Machine Learning model performance: Avg. Single-Subject vs Multi-Subject.

(b) Deep Learning model Performance: Avg. Single-Subject vs Multi-Subject.

Figure 3.24: Comparison of the performance of Machine Learning and Deep Learning models on the Garmin Dataset, using motion (M) and motion plus physiological (M+P) features.

Figure 3.24 presents the weighted F1 scores for ML and DL models on the Garmin dataset, comparing single-subject averages with the combined multi-subject dataset across motion (M) and motion plus physiological (M+P) feature sets. Among the ML models (Figure 3.24a), KNN, Random Forest, and XGBoost display stable performance across single and multi-subject cases, with little to no effect from adding M+P features, indicating resilience to subject variability. SVM, however, exhibits a minor decrease in F1 scores in both single-subject and multi-subject scenarios with both M and M+P features. For DL models (Figure 3.24b), LSTM shows a modest improvement (+0.031) with multi-subject data in the M-only setting, reflecting its adaptability to broader subject diversity. ConvLSTM also benefits slightly from the multi-subject dataset in the M setting, although it remains consistent in the M+P scenario across both single and multi-subject comparisons. The Transformer model maintains robust performance with minimal variations, showing a slight gain in single-subject settings compared to multi-subject in the M scenario, and a minor drop (-0.004) in the M+P setting when transitioning from single-subject to multi-subject,

underscoring its strong generalization capability across subjects. This analysis highlights that while ML models and the Transformer model exhibit resilience across subjects, DL models, particularly LSTM, gain marginally from multi-subject data.

3.10.8 Impact of Physiological Features on Cosinuss° and Garmin Dataset

In Table 3.7, the ConvLSTM model, using motion data only, achieves an overall accuracy of 0.77 in terms of weighted F1 score. The model performs well for activities like sitting and jogging, with F1 scores of 0.99 and 0.92, respectively, but struggles with more complex activities, such as ascending stairs and descending stairs, achieving F1 scores of only 0.43 and 0.35, respectively. This indicates that motion data alone is insufficient for accurately distinguishing activities with nuanced or subtle movement patterns. The macro average F1 score of 0.68 reflects uneven performance across activities, particularly for challenging tasks like stair navigation.

Table 3.7: Classification report for ConvLSTM on Cosinuss° dataset (Multi-Subject) using motion (M) data only.

Activity	Precision	Recall	F1 Score	Support
sitting	0.99	1.00	0.99	4818
ascending stairs	0.60	0.34	0.43	2053
descending stairs	0.39	0.32	0.35	1882
walking	0.62	0.85	0.72	5038
jogging	1.00	0.85	0.92	4267
accuracy			0.78	18058
macro avg	0.72	0.67	0.68	18058
weighted avg	0.78	0.78	0.77	18058

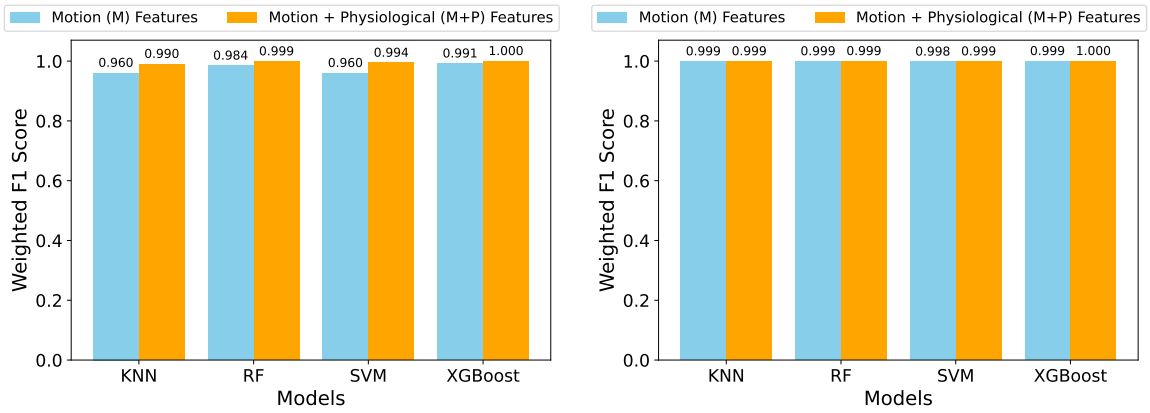
Table 3.8: Classification report for ConvLSTM on Cosinuss° dataset (Multi-Subject) using motion and physiological (M+P) data.

Activity	Precision	Recall	F1 Score	Support
sitting	0.97	1.00	0.98	4818
ascending stairs	0.85	0.87	0.86	2053
descending stairs	0.88	0.84	0.86	1882
walking	0.95	0.98	0.96	5038
jogging	1.00	0.93	0.96	4267
accuracy			0.95	18058
macro avg	0.93	0.92	0.93	18058
weighted avg	0.95	0.95	0.95	18058

In contrast, Table 3.8, which includes both motion and physiological data, shows a sub-

stantial improvement in model performance. The addition of physiological data raises the overall weighted F1 score to 0.95 and enhances F1 scores for ascending and descending stairs to 0.86 for both activities. This improvement highlights the importance of physiological data in aiding activity classification, particularly for complex activities. The macro average F1 score also rises to 0.93, indicating more balanced and consistent performance across all activities. The same situation is observed in the Garmin dataset as well. These results demonstrate that incorporating physiological data significantly boosts the model's ability to classify activities accurately, especially those involving subtle movements.

3.10.9 Comparison of Model Performance Across ML and DL for Garmin and Cosinuss°

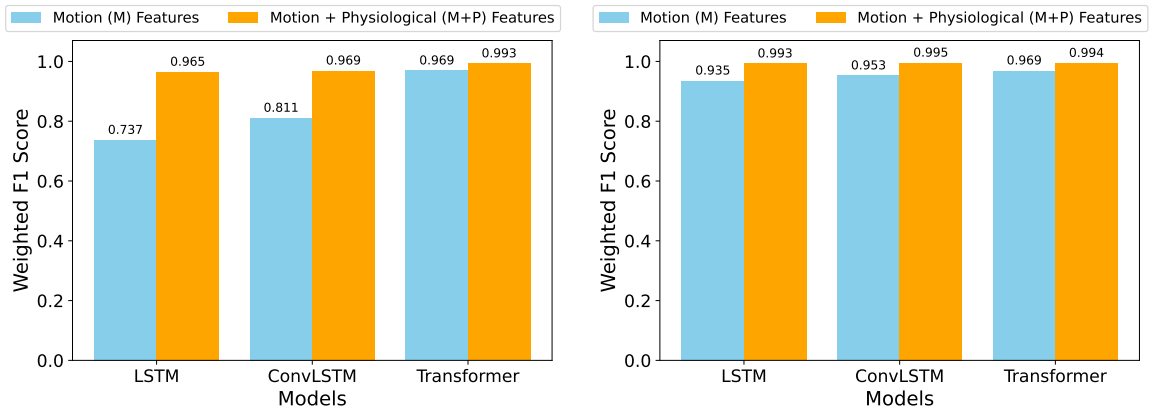


(a) Average ML Performance on Cosinuss° Dataset.

(b) Average ML Performance on Garmin Dataset.

Figure 3.25: Average ML Performance on Cosinuss° and Garmin Datasets (M vs. M+P).

Figure 3.25 presents the average ML performance on the Cosinuss° and Garmin datasets, with the weighted F1 scores averaged across individual subjects and the multi-subject dataset. Figure 3.25a shows the performance on the Cosinuss° dataset, while Figure 3.25b illustrates the results on the Garmin dataset. Across all models (KNN, Random Forest, SVM, and XGBoost), the Garmin dataset achieves nearly perfect F1 scores, with each model scoring close to 1.000, indicating consistent high performance. In contrast, the Cosinuss° dataset displays slightly lower F1 scores, especially for KNN and SVM when using only motion (M) features (0.960). However, incorporating motion plus physiological (M+P) features significantly enhances the Cosinuss° dataset's performance, with Random Forest and XGBoost approaching perfect scores (0.999 and 1.000, respectively). This result highlights the importance of integrating motion and physiological data, especially for the Cosinuss° dataset, in achieving the best classification accuracy.



(a) Average DL Performance on Cosinuss° Dataset.

(b) Average DL Performance on Garmin Dataset.

Figure 3.26: Average DL Performance on Cosinuss° and Garmin Datasets (M vs. M+P).

Figure 3.26 illustrates the average performance of DL models on the Cosinuss° and Garmin datasets, comparing motion (M) features alone with motion plus physiological (M+P) features. In the Cosinuss° dataset (Figure 3.26a), incorporating M+P features significantly enhances model performance, with the LSTM model improving from 0.737 to 0.965, ConvLSTM from 0.811 to 0.969, and Transformer achieving the highest F1 score of 0.993. This highlights the added value of physiological data for activity recognition in Cosinuss° data. In contrast, the Garmin dataset (Figure 3.26b) shows consistently high performance across all models, with minor improvements from M+P features. Here, LSTM, ConvLSTM, and Transformer achieve F1 scores of 0.993, 0.995, and 0.994, respectively, demonstrating the robustness of the Garmin dataset.

3.10.10 Effect of Multimodal Data on Model Performance

Figure 3.27 illustrates the performance variability of machine learning models (KNN, Random Forest, SVM, and XGBoost) on the Cosinuss° dataset, comparing motion-only (M) features and combined motion plus physiological (M+P) features. The inclusion of physiological data leads to consistent improvements in weighted F1 scores, particularly for KNN and SVM, where it notably reduces model variability. While Random Forest and XGBoost already achieve high F1 scores with motion-only data, XGBoost demonstrates near-perfect and stable performance across both feature sets. These results highlight the benefits of multimodal data in improving model stability and accuracy, particularly for models that are more susceptible to variation, such as KNN and SVM.

Figure 3.28 illustrates the impact of combining motion and physiological data (M+P) on the performance of the deep learning models (LSTM, ConvLSTM, and Transformer) using the Cosinuss° dataset. The LSTM model shows the most pronounced improvement, with

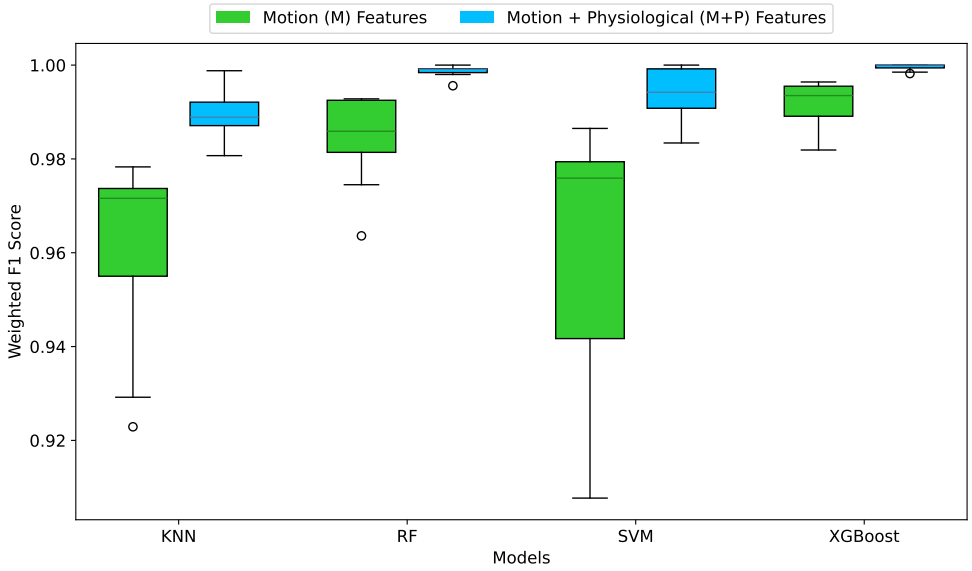


Figure 3.27: Comparison of machine learning models on the Cosinuss° dataset using motion-only (M) and motion plus physiological (M + P) features.

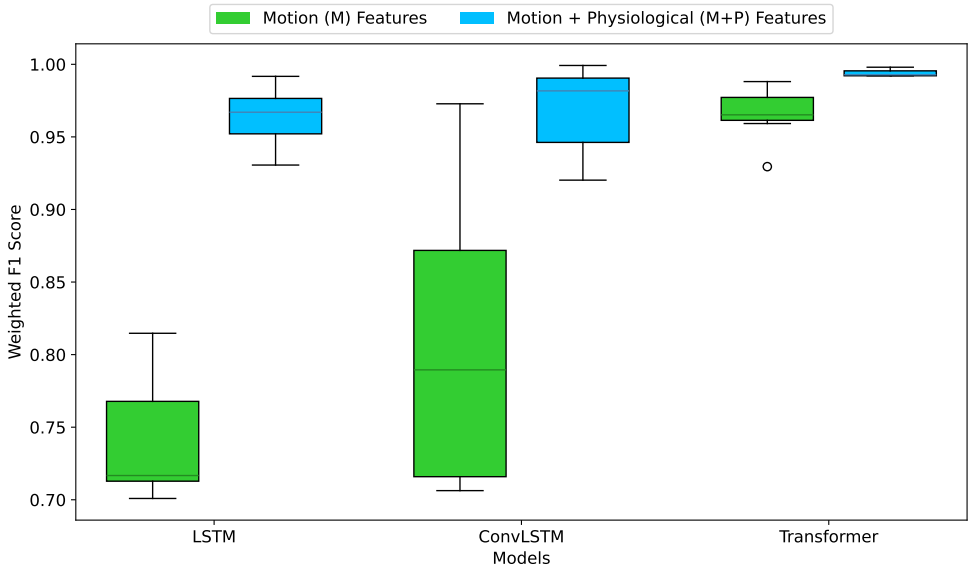


Figure 3.28: Deep learning models on the Cosinuss° dataset using motion-only (M) and motion plus physiological (M + P) features.

its weigted F1 score and variability significantly enhanced by the inclusion of physiological

features. ConvLSTM, which exhibits substantial variability with motion-only data, also achieves better performance and stability with M+P features. The Transformer model, which already performs well with motion features, benefits from further refinement when physiological data is added, reaching near-perfect performance with minimal variance. This demonstrates the importance of multimodal data in improving both performance and consistency in deep learning models.

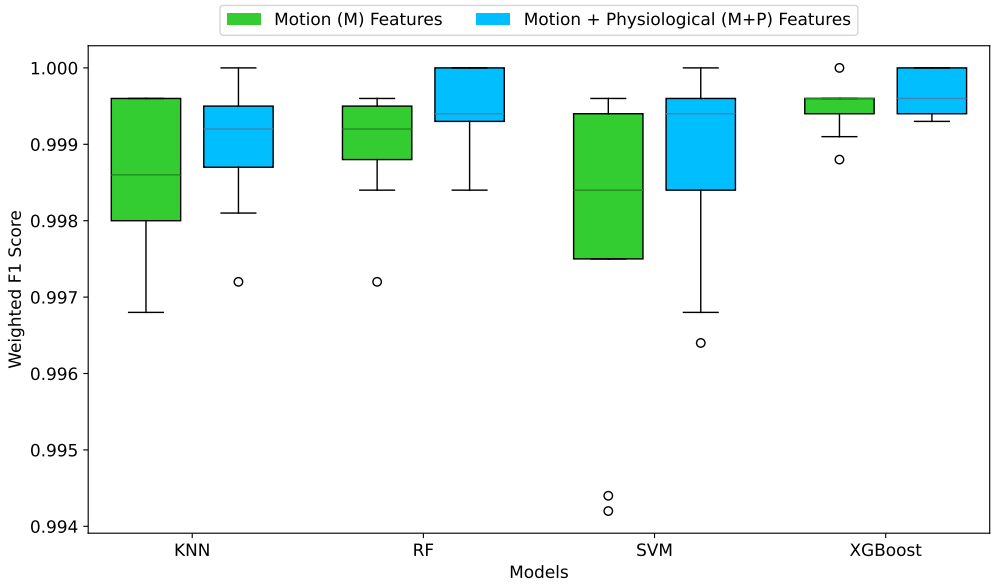


Figure 3.29: Machine learning model performance on the Garmin dataset using motion-only (M) and motion plus physiological (M + P) features.

Figure 3.29 displays the weighted F1 scores for four machine learning models (KNN, Random Forest, SVM, and XGBoost) evaluated on the Garmin dataset, contrasting the performance of motion-only (M) features with the combined motion plus physiological (M+P) features. Overall, all models achieve near-perfect F1 scores, with minor gains observed when physiological features are added. SVM particularly benefits from the addition of M+P features, with a reduction in both variability and outlier presence compared to motion-only data. KNN and XGBoost maintain consistently high performance with minimal variance, irrespective of the feature set. While the overall improvements from M+P features are marginal due to the strong baseline performance, the additional data helps to improve model stability and consistency.

Figure 3.30 shows the performance comparison of three deep learning models (LSTM, ConvLSTM, and Transformer) on the Garmin dataset, highlighting the impact of motion-only (M) versus motion plus physiological (M+P) features on weighted F1 scores. The results clearly show that incorporating physiological data improves performance for all models. LSTM, in particular, benefits from a notable increase in F1 score and reduced

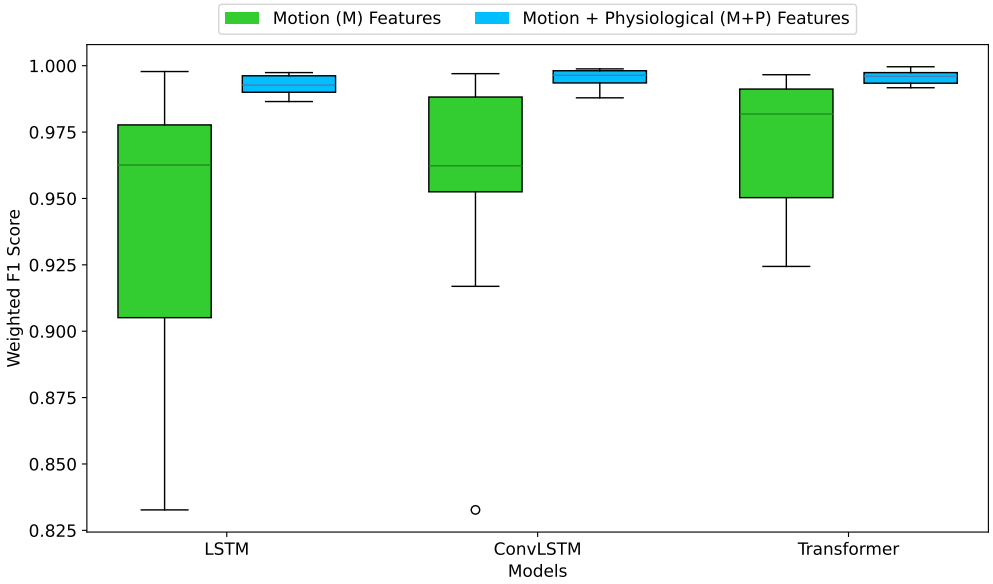
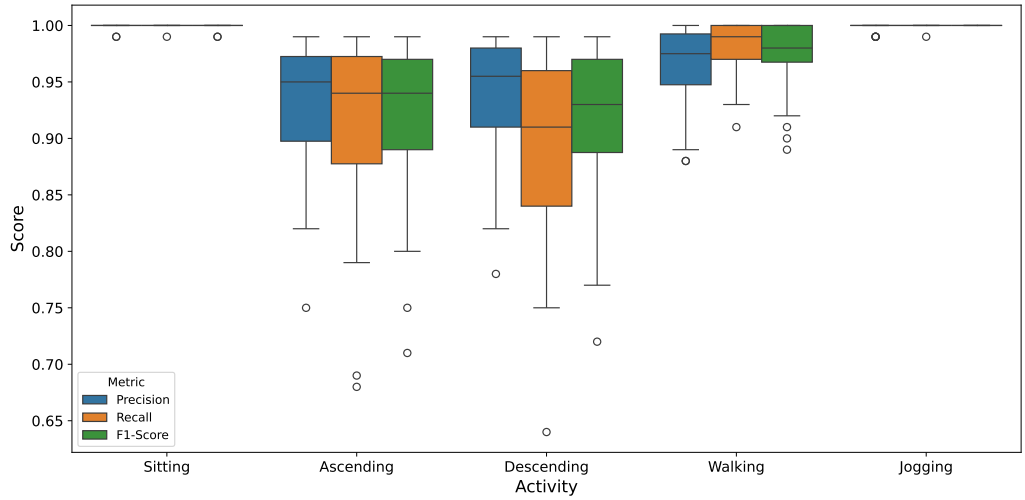


Figure 3.30: Deep learning model performance on the Garmin dataset with motion-only and motion plus physiological features.

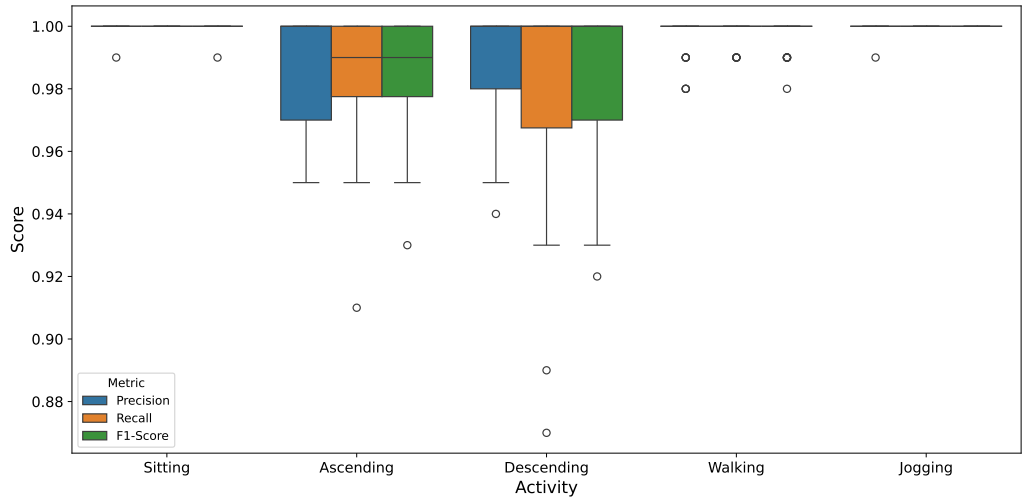
variability with M+P features. ConvLSTM, which displays considerable variance with motion-only data, stabilizes and achieves higher scores when physiological features are added. The Transformer model, already strong with motion features, reaches near-perfect performance with M+P features, showing minimal variability. These results demonstrate the effectiveness of integrating physiological data to enhance deep learning model performance and consistency on the Garmin dataset.

3.10.11 Effect of Multimodal Data on Activity Classification

Figure 3.31 illustrates the performance of the machine learning models on the Cosinuss° datasets using motion-only (M) data and motion plus physiological (M+P) data for different activities. In Figure 3.31a, the box plot shows how the models perform using only motion data. Activities like sitting and jogging exhibit consistently high precision, recall, and F1 scores with minimal variance. However, for more dynamic activities such as ascending and descending, the performance of the models shows higher variability, especially in recall and F1 score, indicating challenges in accurately classifying these activities using motion data alone. Figure 3.31b reveals the impact of adding physiological data to the motion data. With the addition of physiological signals, the performance of the models improves, particularly for walking, ascending and descending, where the precision, recall, and F1 scores are higher and more consistent. This improvement suggests that physiological data provides valuable context for recognizing more complex activities. For simpler



(a) Box Plot for different activities using motion data only.

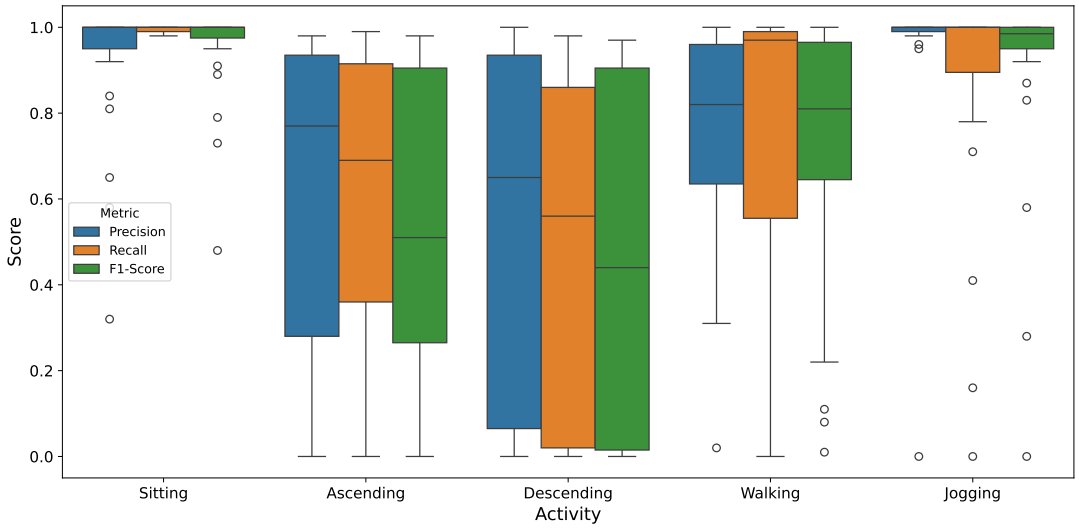


(b) Box Plot for different activities using motion and physiological data.

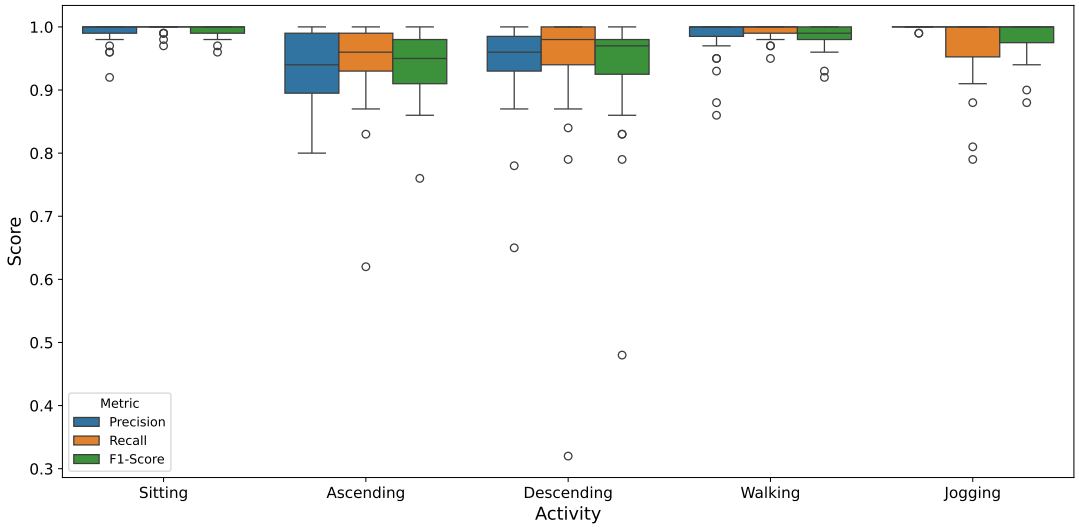
Figure 3.31: Activity variation on Cosinuss° Datasets (M vs. M+P) using Machine Learning Models.

activities like sitting and jogging, the addition of physiological data has a less pronounced effect, as these activities were already classified well using motion data alone.

Figure 3.32 compares the performance of deep learning models on the Cosinuss° datasets using motion-only (M) data and motion plus physiological (M+P) data. In Figure 3.32a, the box plot shows the variance in precision, recall, and F1 score for different activities when using only motion data. While the models perform well on simpler activities such



(a) Box Plot for different activities using motion data only.

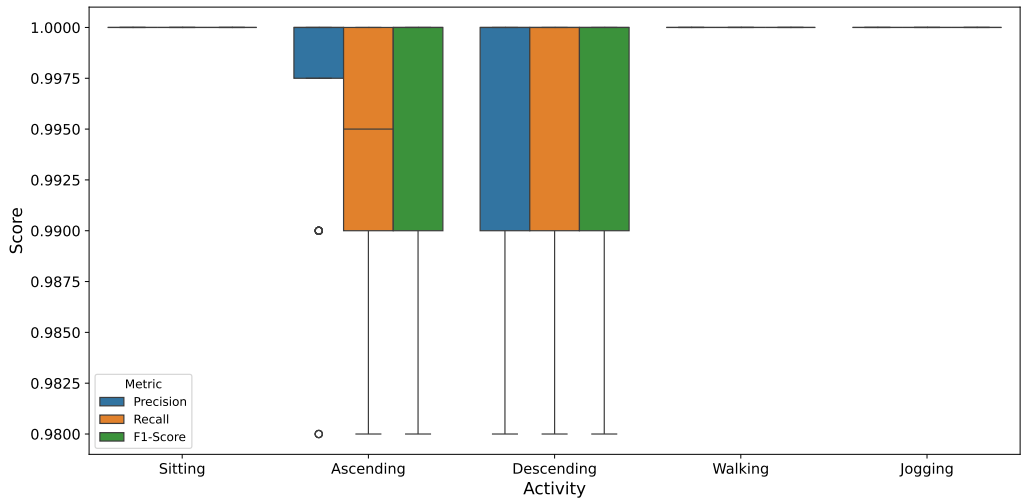


(b) Box Plot for different activities using motion and physiological data.

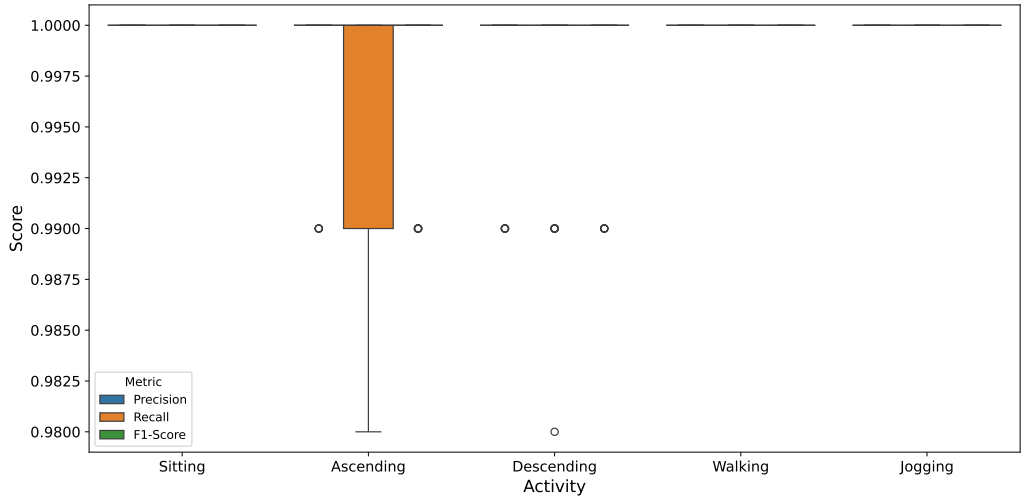
Figure 3.32: Activity variation on Cosinuss° Datasets (M vs. M+P) using Deep Learning Models.

as sitting and jogging, they struggle with more complex movements, particularly ascending and descending stairs, where there is a wider variance in the F1 scores. This suggests that motion data alone may not be sufficient for accurately classifying certain activities. Figure 3.32b demonstrates the improvement in the performance of the models when phys-

iological data, such as heart rate and temperature, are added alongside motion data. Activities like ascending and descending, which previously exhibited high variability, now show more consistent and higher precision, recall, and F1 scores. Incorporating physiological data helps reduce uncertainty and improve classification accuracy, particularly for activities involving subtle or vertical movements.



(a) Box Plot for different activities using motion data only.



(b) Box Plot for different activities using motion and physiological data.

Figure 3.33: Activity variation on Garmin Datasets (M vs. M+P) on Machine Learning Models.

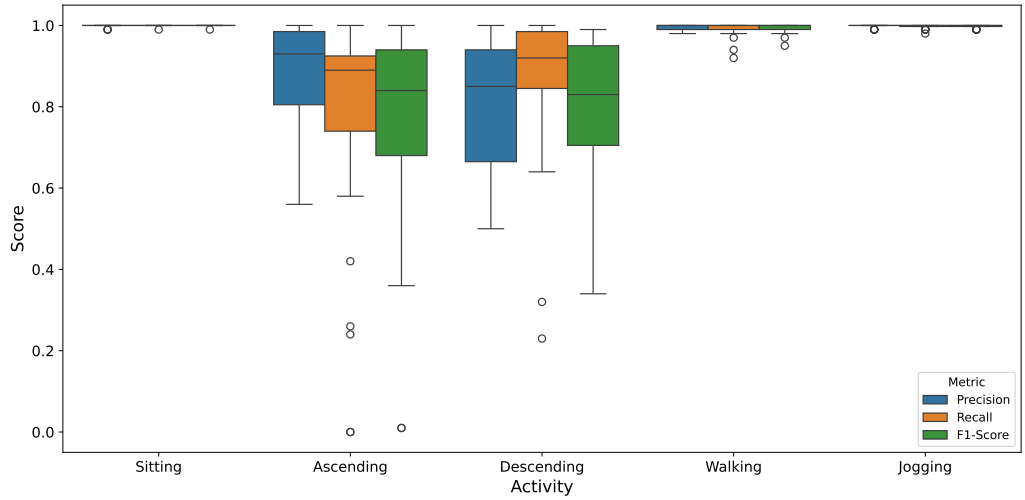
Figure 3.33 presents the performance of machine learning models using the Garmin

datasets using motion-only (M) data and motion plus physiological (M+P) data for various activities. In Figure 3.33a, the box plot shows the performance of the models based solely on motion data. For most activities, such as sitting, walking, and jogging, the model achieves near-perfect precision, recall, and F1 scores with minimal variance. However, activities like ascending and descending show slightly higher variability, indicating that these activities are more challenging to classify using only motion data. Figure 3.33b illustrates the performance of the models when both motion and physiological data are combined. While activities such as sitting, walking, and jogging maintain their near-perfect classification, the addition of physiological data does not show a significant improvement for ascending and descending activities, which exhibit some variability in recall. This indicates that for the Garmin dataset, incorporating physiological data may not enhance the performance of the models for more dynamic activities as significantly as it does for simpler activities, where performance is already nearly perfect.

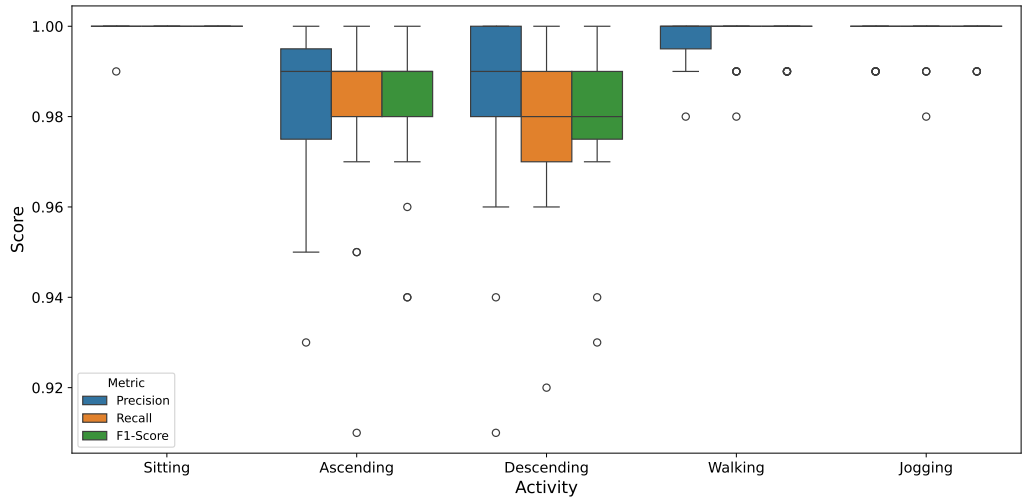
Figure 3.34 presents the performance of deep learning models on Garmin datasets using motion-only (M) data and combined motion plus physiological (M+P) data for various activities. In Figure 3.34a, the box plot shows the variation in precision, recall, and F1 score when using only motion data. For activities such as sitting and jogging, the models perform with high consistency across all metrics, as evidenced by the narrow range in values. However, for more dynamic activities like ascending and descending stairs, the performance exhibits greater variability, especially in recall and F1 score. This indicates that the models struggle more with these movements when relying solely on motion data. Figure 3.34b highlights the impact of adding physiological data alongside motion data. For activities like ascending and descending, there is a noticeable improvement in both precision and F1 score, showing how the inclusion of physiological signals helps reduce the variance and increases the accuracy of the models. Despite this improvement, activities such as sitting and jogging show minimal change, as these simpler activities are already classified with high precision using motion data alone.

3.10.12 Zero Shot Learning

In this section, the effectiveness of zero shot learning is evaluated on the Cosinuss° and Garmin datasets using a Transformer model. Subject S2 was chosen as the training subject for its strong performance on both the Cosinuss° and Garmin datasets, while the trained model was tested on subject S1 for both datasets. This setup allows us to analyze the model's generalization ability across subjects and assess how well it can leverage cross-subject knowledge without additional fine-tuning. Both motion-only and motion plus physiological feature sets are included to assess the impact of physiological signals on model performance, especially in distinguishing various activities with unseen data. Traditional machine learning models could not be tested in this setup because recursive feature elimination (RFE) was used for feature selection, leading to different feature sets for S2 and S1, which prevented consistent cross-subject evaluation.



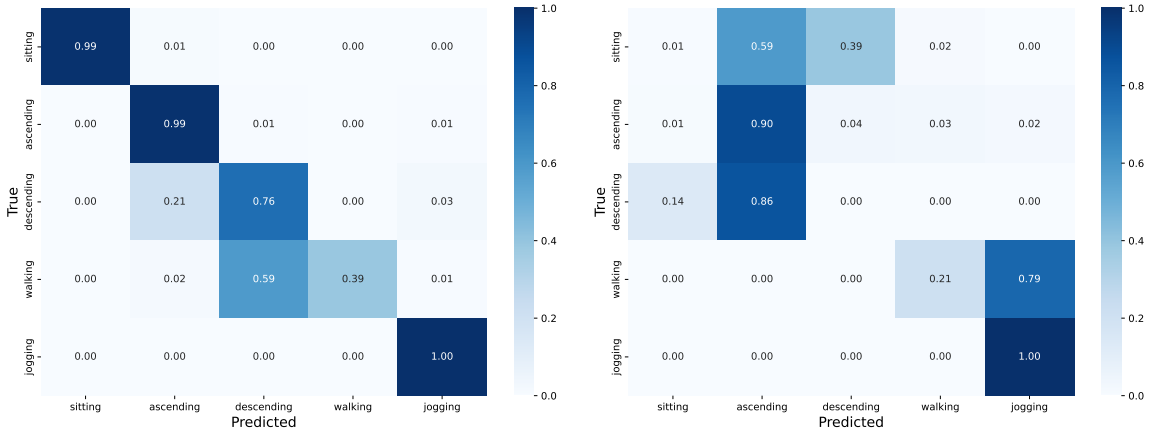
(a) Box Plot for different activities using motion data only.



(b) Box Plot for different activities using motion and physiological data.

Figure 3.34: Activity variation on Garmin Datasets (M vs. M+P) on Deep Learning Models.

The confusion matrices in Figure 3.35 illustrate the performance of zero shot learning on the Cosinuss° dataset using motion-only (M) features (Figure 3.35a) and motion plus physiological (M+P) features (Figure 3.35b). With motion-only features, the model effectively classifies activities, showing high accuracy across most classes. However, when incorporating physiological data, performance declines, especially in distinguishing between activities like ascending and descending. This decrease in accuracy can be attributed to the variability in physiological signals, which differ significantly from user to user, making it



(a) Confusion Matrix with motion (M) Features. (b) Confusion Matrix with motion and physiological (M+P) Features.

Figure 3.35: Zero Shot Learning Analysis on Cosinuss° Dataset: Comparison of Confusion Matrices for motion (M) vs. motion plus physiological (M+P) Features.

challenging for the model to generalize. The individualized nature of health vitals, such as heart rate and SpO_2, adds complexity to the model's task, as these signals are less consistent across different subjects compared to motion data, which is relatively more stable and activity-specific.

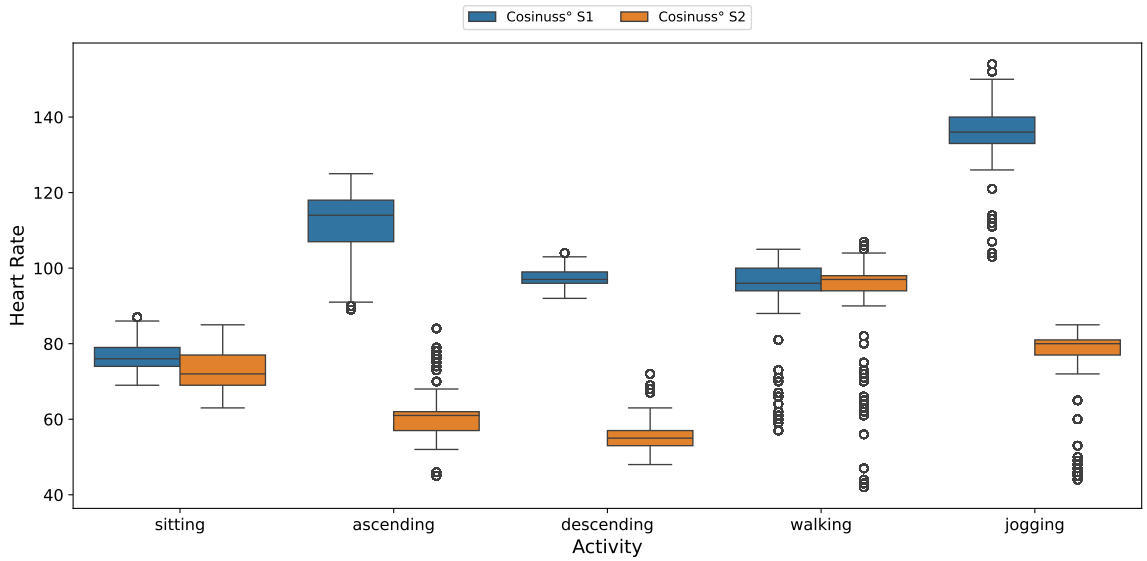
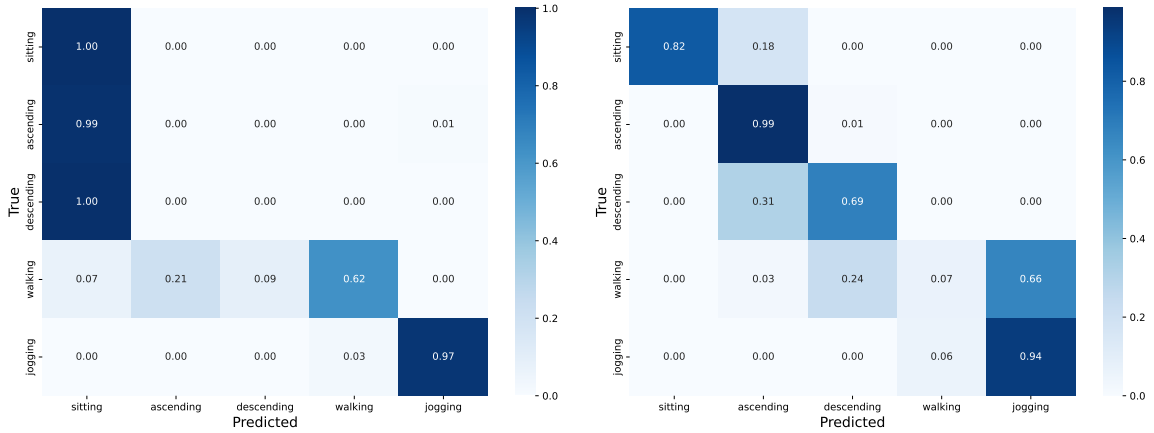


Figure 3.36: Heart Rate Distribution for subject S1 and S2 of Cosinuss° Dataset.

Figure 3.36 shows the heart rate distribution for subjects S1 and S2 across different activities, illustrating how inter-subject variability impacts model generalization in the Cosinuss° dataset. Activities like ascending and jogging show a higher heart rate range for S1 compared to S2, suggesting different exertion levels. Conversely, descending and sitting display lower heart rate variability, with S2 generally having lower heart rates across these activities. This variability complicates zero shot learning, as physiological responses differ significantly between individuals, leading to inconsistent patterns across similar activities. As shown in Figure 3.35, such differences hinder the model's ability to generalize effectively, indicating that physiological data alone may not provide the necessary distinctiveness for accurate cross-subject classification, especially in activities with subtle physiological changes.



(a) Confusion Matrix with motion (M) Features.

(b) Confusion Matrix with motion and physiological (M+P) Features.

Figure 3.37: Zero Shot Learning Analysis on Garmin Dataset: Comparison of Confusion Matrices for motion (M) vs. motion plus physiological (M+P) Features.

Figure 3.37 presents the confusion matrices for zero shot learning on the Garmin dataset using motion-only (M) features (Figure 3.37a) and combined motion plus physiological (M+P) features (Figure 3.37b). With motion-only data, the model achieves high accuracy for sitting and jogging but struggles to distinguish complex activities, such as ascending and descending stairs. When physiological data is included, the model's performance improves for these complex movements. This enhancement is likely due to the additional physiological indicators, such as heart rate and SpO_2, which vary significantly with physical exertion and provide unique signals for activities involving elevation changes. These physiological metrics complement motion data by capturing the body's response to different activity levels, making it easier for the model to differentiate between similar activities that vary in physical intensity.

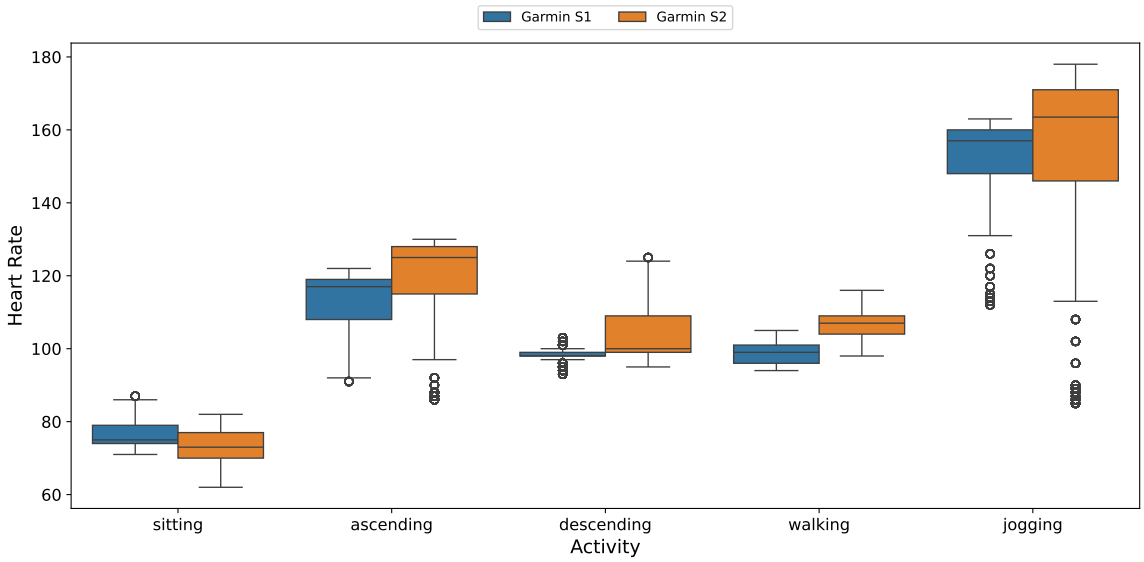


Figure 3.38: Heart Rate Distribution for subject S1 and S2 of Garmin Dataset.

Figure 3.38 shows the heart rate distribution for subjects S1 and S2 across various activities in the Garmin dataset, illustrating how physiological data enhances model performance in distinguishing complex movements. Activities like ascending and jogging exhibit higher heart rates for both subjects, while sitting has the lowest heart rate, reflecting different exertion levels. This consistent trend in heart rate across both users helps the model better differentiate between complex activities, such as ascending, descending and jogging, when physiological data is included, as seen in Figure 3.37. The distinct heart rate patterns for each activity provide complementary information to motion data, enabling the model to capture intensity-specific cues. For example, in descending and walking, the heart rate differences between subjects create a clearer distinction than motion data alone, thereby improving model accuracy in zero shot learning by leveraging both motion and physiological features.

Figure 3.39 compares heart rate distributions for subjects S1 and S2 across different activities using both Cosinuss° and Garmin datasets. Overall, in this experimental setting, Garmin demonstrates more consistent heart rate patterns between subjects, particularly in activities like ascending and jogging, where both S1 and S2 show similar trends and ranges. This stability supports Garmin's effectiveness for complex activity recognition, such as ascending and descending. In contrast, Cosinuss° data exhibits significant inter-subject variability, especially for S2, where heart rate trends diverge from those of S1. This inconsistency may be due to sensor placement issues in the ear for S2, potentially affecting signal accuracy, or individual physiological responses that differ between subjects. Such variability suggests that Cosinuss° measurements may be more sensitive to sensor positioning, impacting reliability in cross-subject scenarios.

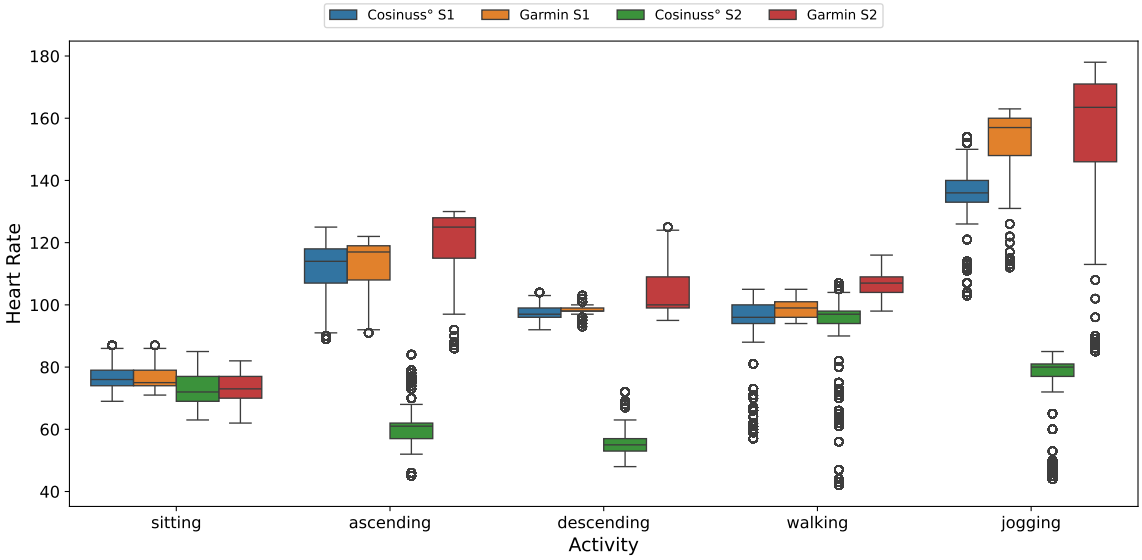


Figure 3.39: Heart Rate Distribution for subject S1 and S2 of Cosinuss° and Garmin Dataset.

3.10.13 Hyperparameter Optimization Analysis

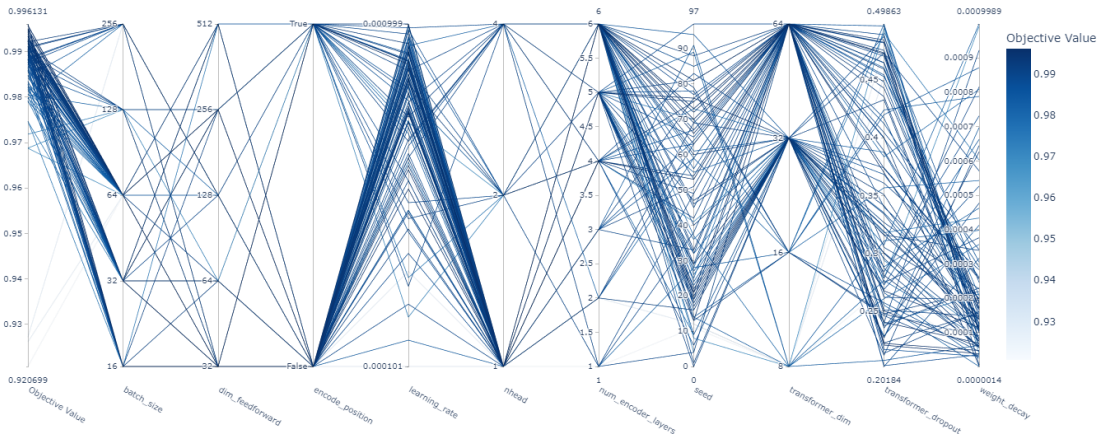


Figure 3.40: Parallel Coordinates Plot for Transformer Hyperparameter Tuning on Cosinuss° Dataset.

Figure 3.40 shows a parallel coordinates plot generated by Optuna framework [1], illustrating the influence of various hyperparameters such as batch size, dropout rate, and

learning rate on model performance. Each line represents a trial, or a unique set of hyperparameters, with the line's position on each axis indicating the chosen hyperparameter value. Line color reflects the objective value (e.g., weighted F1 score), with darker lines denoting better performance. This plot helps uncover patterns and interactions among hyperparameters, highlighting ranges that optimize model performance. For example, we observe that certain values of batch size and dropout rate consistently yield higher objective values, suggesting their critical influence on the model's effectiveness.

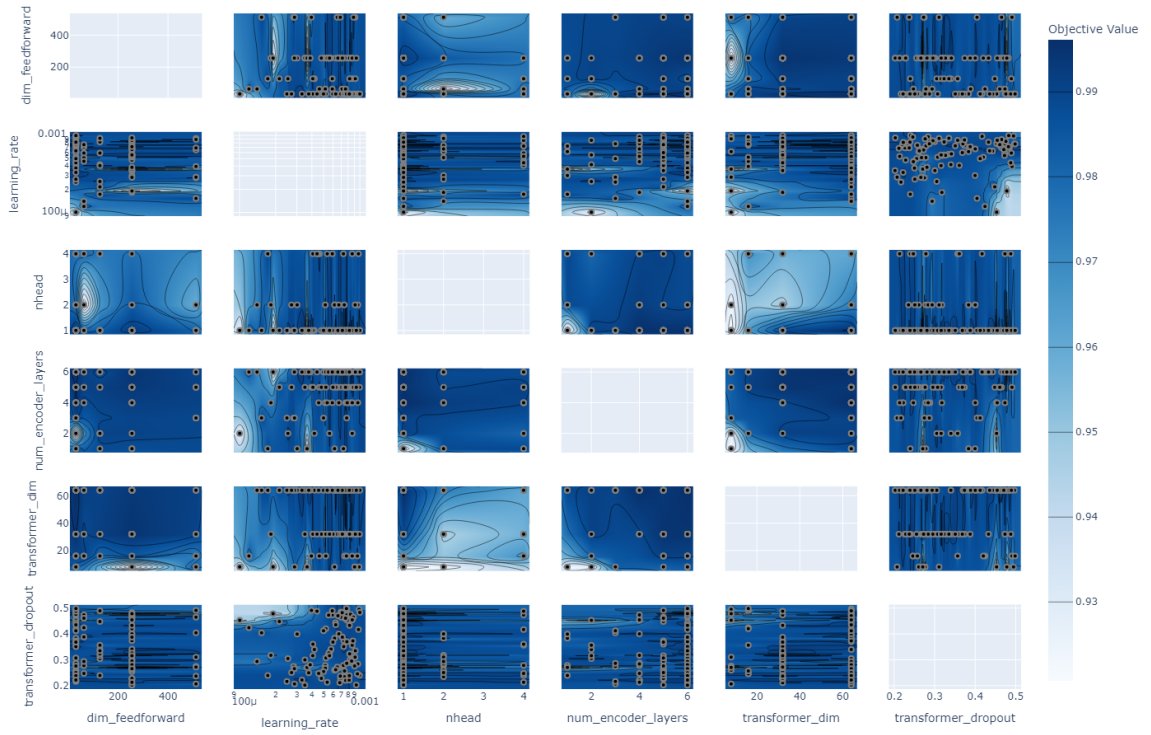


Figure 3.41: Contour Plot of Hyperparameter Interactions for Transformer Tuning on Cosinuss° Dataset.

Figure 3.41 presents a contour plot illustrating the interactions between selected hyperparameters and their influence on model performance. Each subplot shows a pair of hyperparameters, with color gradients representing the objective value (e.g., weighted F1 score), where darker regions indicate better performance. Contour lines highlight areas of optimal combinations, making it easier to identify effective hyperparameter ranges. For instance, specific regions in the learning rate vs. dropout plot show distinct contours, suggesting a strong interaction and a notable impact on performance. In contrast, pairs like encoder layers and dropout exhibit flatter contours, implying less sensitivity within the

tested range. This plot provides valuable insights into which hyperparameter combinations most effectively improve the model, guiding targeted tuning strategies.

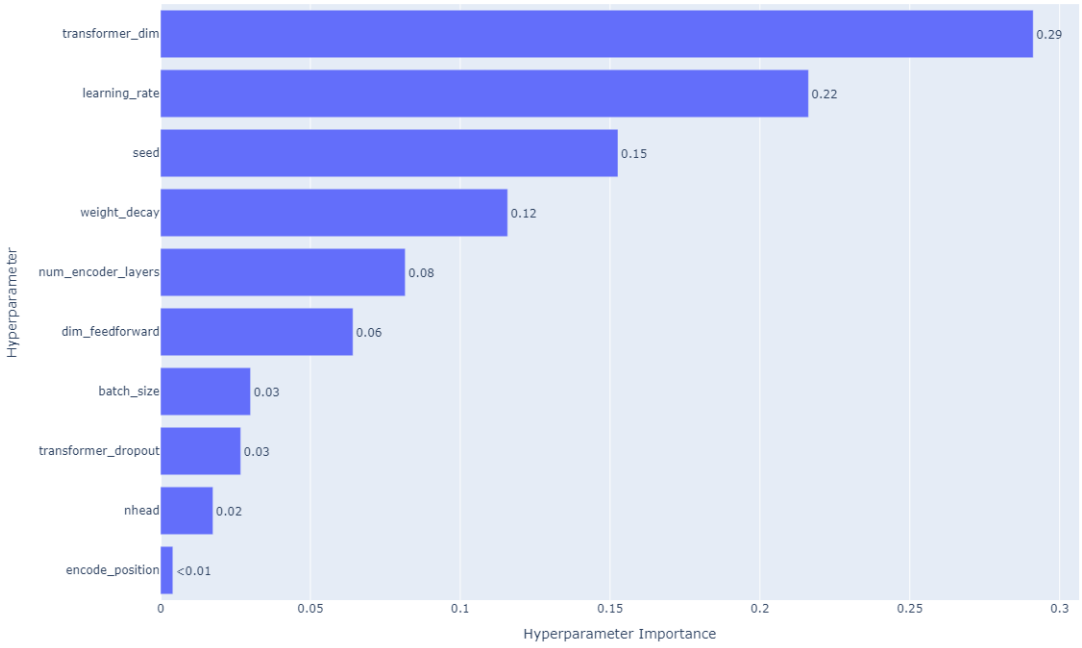


Figure 3.42: Hyperparameter Importance for Transformer Model Tuning on Cosinuss° Dataset.

Figure 3.42 illustrates the relative importance of hyperparameters in optimizing the Transformer model using Cosinuss° data. The transformer dimension has the highest impact (0.29), followed by the learning rate (0.22) and random seed (0.15), indicating these parameters are critical for performance. Weight decay also has moderate influence (0.12), while the number of encoder layers (0.08) and feedforward dimension (0.06) have smaller effects. Other hyperparameters, such as batch size, dropout, and the number of heads, show minimal impact, each contributing 0.03 or less. This plot highlights which hyperparameters are most valuable to prioritize in fine-tuning efforts.

Figure 3.43 shows the progression of objective values during the hyperparameter tuning process for the Transformer model on the Cosinuss° dataset. Each blue dot represents a trial, which is a single run of the model with a specific set of hyperparameters, and its corresponding objective value (e.g., weighted F1 score). The red line tracks the best value achieved up to each trial. The plot reveals that the model's performance rapidly improves within the initial trials, stabilizing around an optimal objective value of approximately 0.99

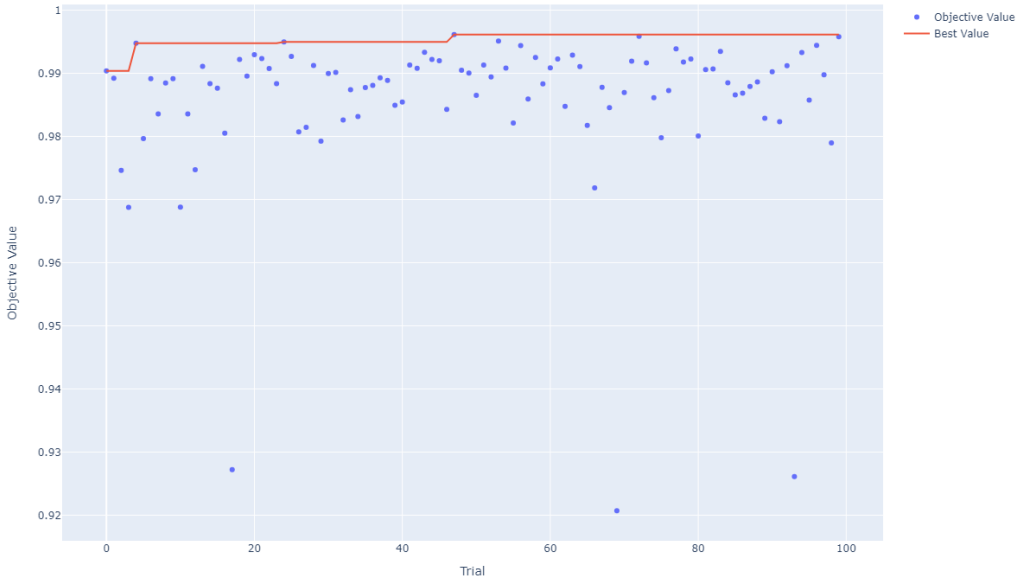


Figure 3.43: Objective Value (weighted F1 score) Progression During Hyperparameter Tuning for Transformer Model on Cosinuss° Dataset.

after about 10 trials. Subsequent trials produce values close to this optimal level, indicating that the tuning process converged efficiently, with only minor performance variations in later trials. This suggests that the search algorithm quickly identified effective hyperparameter configurations, leading to a stable and near-optimal solution early in the process.

4 Conclusions

This study explores the effectiveness of wearable sensor data from Cosinuss° and Garmin devices in Human Activity Recognition (HAR). It focuses on how combining motion-only features with physiological features influences model performance. Machine learning and deep learning models are applied to assess the impact of these data combinations, with particular attention to their performance in recognizing complex activities like stair ascent and descent. Machine learning models (e.g., KNN, SVM, Random Forest, XGBoost), which rely on handcrafted features, performed robustly with motion-only data, demonstrating that carefully engineered features capture essential activity information effectively. In contrast, deep learning models (e.g., LSTM, ConvLSTM, Transformer) trained on non-handcrafted features saw substantial performance gains with the addition of physiological data, with the Transformer model consistently outperforming others across single and multi-subject datasets. The LSTM model, for instance, exhibited a 30.03% improvement with motion and physiological data in Cosinuss°, reaching a performance level comparable to machine learning models. Furthermore, zero shot learning findings reveal that accelerometer data by Cosinuss° provided superior classification with motion-only data compared to Garmin's multi-featured motion data. However, when physiological features were included, the Garmin dataset achieved better zero shot generalization across subjects than Cosinuss°, underscoring the importance of physiological data for cross-subject transferability. Overall, the study highlights that while motion data alone may suffice for simpler tasks, physiological data is essential for accurately recognizing complex movements. The Transformer model's consistent performance further emphasizes its suitability for generalized activity recognition in multimodal wearable applications.

The multimodal characteristics of the dataset highlight distinct differences in the data capture capabilities and limitations of the Cosinuss° and Garmin devices, each presenting unique challenges. Cosinuss°, an in-ear wearable, excels in continuous physiological monitoring, capturing high-quality heart rate, SpO_2, and body temperature data through a photoplethysmography (PPG) sensor. However, its motion data is limited to a single triaxial accelerometer, which may reduce its sensitivity to complex spatial movement patterns. Data collection can be further complicated in free-living conditions, as connectivity issues may interrupt data transmission to the online portal, leading to intermittent data loss. Additionally, some participants reported occasional fit instability during complex activities, which could affect data quality. To address this, Cosinuss° includes a signal quality parameter to help ensure collected data meets predefined standards. Garmin, by contrast, captures a broader range of motion data through an integrated accelerometer, gyroscope, and GPS, which enhances spatial accuracy but may compromise the stability of physi-

ological data during high-motion activities due to potential wrist sensor displacement. Furthermore, Garmin's high accelerometer and gyroscope sampling rates impact battery life, potentially limiting continuous data collection. While this study includes eight participants, increasing the number of participants would strengthen the robustness of HAR patterns and enhance model accuracy, especially across varied real-world contexts.

Future research should focus on implementing transfer learning and cross-subject training to improve HAR model generalizability, allowing models trained on large datasets to adapt quickly to new users and scenarios. Developing robust cross-subject training could further enhance generalization by allowing models to better account for inter-user variability, especially in physiological responses. Additionally, optimizing window and overlap configurations could be explored to understand their impact on model performance, potentially fine-tuning these parameters to improve accuracy across activities [20]. Further research is needed to validate the reliability of these models in diverse populations, including non-healthy clinical groups, to expand HAR's applicability in healthcare and other domains. Integrating techniques like deep reinforcement learning (DRL) [25] and large language models (LLMs) may also strengthen HAR by enabling adaptive learning, efficient annotation of large datasets, and personalization based on user preferences. By focusing on these advancements, HAR systems can evolve into more accurate, adaptive, and user-centered tools for various fields, thriving in complex real-world environments.

Bibliography

[1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019.

[2] Hande Özgür Alemdar, Halil Ertan, Özlem Durmaz Incel, and Cem Ersoy. Aras human activity datasets in multiple homes with multiple residents. *2013 7th International Conference on Pervasive Computing Technologies for Healthcare and Workshops*, pages 232–235, 2013.

[3] D. Anguita, Alessandro Ghio, L. Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In *International Workshop on Ambient Assisted Living and Home Care*, 2012.

[4] Ankita, Shalli Rani, Himanshi Babbar, S. Coleman, Aman Singh, and Hani Moaiteq Abdullah Aljahdali. An efficient and lightweight deep learning model for human activity recognition using smartphones. *Sensors (Basel, Switzerland)*, 21, 2021.

[5] Vidyananth Balu and Sasikumar P. Wearable multi-sensor data fusion approach for human activity recognition using machine learning algorithms. *SSRN Electronic Journal*, 2022.

[6] Djamila Romaissa Beddiar, Brahim Nini, M. Sabokrou, and Abdenour Hadid. Vision-based human activity recognition: a survey. *Multimedia Tools and Applications*, 79:30509 – 30555, 2020.

[7] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Neural Information Processing Systems*, 2011.

[8] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13:281–305, 2012.

[9] Lukas Boborzi, Julian Decker, Razieh Rezaei, Roman Schniepp, and Max Wuehr. Human activity recognition in a free-living environment using an ear-worn motion sensor. *Sensors (Basel, Switzerland)*, 24, 2024.

[10] Jason Brownlee. Supervised and unsupervised machine learning algorithms, September 2016. Accessed: 2024-09-24.

[11] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *ArXiv*, abs/2005.12872, 2020.

[12] Kaixuan Chen, Dalin Zhang, Lina Yao, Bin Guo, Zhiwen Yu, and Yunhao Liu. Deep learning for sensor-based human activity recognition. *ACM Computing Surveys (CSUR)*, 54:1 – 40, 2020.

[13] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

[14] Alok Kumar Chowdhury, Dian Tjondronegoro, Vinod Chandran, and Stewart G. Trost. Physical activity recognition using posterior-adapted class-based fusion of multiaccelerometer data. *IEEE Journal of Biomedical and Health Informatics*, 22:678–685, 2017.

[15] Seungeun Chung, Chi Yoon Jeong, Jeong-Mook Lim, Jiyoun Lim, Kyoung Ju Noh, Gague Kim, and Hyuntae Jeong. Real-world multimodal lifelog dataset for human behavior study. *ETRI Journal*, 44:426 – 437, 2021.

[16] Corinna Cortes and Vladimir Naumovich Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[17] Cosinuss GmbH. C-med alpha in-ear sensor. https://www.cosinuss.com/en/products/in-ear-sensors/c-med-alpha/. Accessed: 2024-10-14.

[18] Burcu F. Darst, Kristen M C Malecki, and Corinne D. Engelman. Using recursive feature elimination in random forest to account for correlated variables in high dimensional data. *BMC Genetics*, 19, 2018.

[19] Samundra Deep and Xi Zheng. Leveraging cnn and transfer learning for vision-based human activity recognition. *2019 29th International Telecommunication Networks and Applications Conference (ITNAC)*, pages 1–4, 2019.

[20] Akbar Dehghani, Omid Sarbishei, Tristan Glatard, and Emad Shihab. A quantitative comparison of overlapping and non-overlapping sliding windows for human activity recognition using inertial sensors. *Sensors (Basel, Switzerland)*, 19, 2019.

[21] Florenc Demrozi, Cristian Turetta, Philipp H Kindt, Fabio Chiarani, Ruggero Angelo Bacchin, Nicola Valè, Francesco Pascucci, Paola Cesari, Nicola Smania, Stefano Tamburin, et al. A low-cost wireless body area network for human activity recognition in healthy life and medical applications. *IEEE Transactions on Emerging Topics in Computing*, 11(4):839–850, 2023.

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.

[23] Giovanni Diraco, Gabriele Rescio, Pietro Siciliano, and Alessandro Leone. Review on human action recognition in smart living: Sensing technology, multimodality, real-time processing, interoperability, and resource-constrained processing. *Sensors (Basel, Switzerland)*, 23, 2023.

[24] Navid Mohammadi Foumani, Chang Wei Tan, Geoffrey I. Webb, and Mahsa Salehi. Improving position encoding of transformers for multivariate time series classification. *Data Mining and Knowledge Discovery*, pages 1–27, 2023.

[25] Fuqiang Gu, Mu-Huan Chung, Mark H. Chignell, Shahrokh Valaee, Baoding Zhou, and Xue Liu. A survey on deep learning for human activity recognition. *ACM Computing Surveys (CSUR)*, 54:1 – 34, 2021.

[26] Simon Haykin. *Neural Networks and Learning Machines*. Pearson Prentice Hall, third edition edition, 2009.

[27] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997.

[28] Hodinky 365. Garmin venu 2 silver / granite blue band. https://www.hodinky-365.com/garmin-venu-2-silver-granite-blue-band-x1212936. Accessed: 2024-10-14.

[29] Zawar Hussain, Quan Z. Sheng, and W. Zhang. A review and categorization of techniques on device-free human activity recognition. *J. Netw. Comput. Appl.*, 167:102738, 2020.

[30] Chihiro Ito, Masaki Shuzo, and Eisaku Maeda. Cnn for human activity recognition on small datasets of acceleration and gyro sensors using transfer learning. *Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers*, 2019.

[31] Sivakumar Kalimuthu, Thinagaran Perumal, Razali Yaakob, Erzam Marlisah, and Lawal Babangida. Human activity recognition based on smart home environment and their applications, challenges. *2021 International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, pages 815–819, 2021.

[32] Junhyuk Kang, Jieun Shin, Jaewon Shin, Daeho Lee, and Ahyoung Choi. Robust human activity recognition by integrating image and accelerometer sensor data using deep fusion network. *Sensors (Basel, Switzerland)*, 22, 2021.

[33] Pritam Khan, Yogesh Kumar, and Sudhir Kumar. Capslstm-based human activity recognition for smart healthcare with scarce labeled data. *IEEE Transactions on Computational Social Systems*, 11(1):707–716, 2022.

[34] Sheharyar Khan, Sadam Hussain Noorani, Aamir Arsalan, Awais Mahmood, Usman Rauf, and Zohaib Ali. Classification of human physical activities and postures during everyday life. *2023 18th International Conference on Emerging Technologies (ICET)*, pages 98–103, 2023.

[35] Jong-Sung Kim. Dnn-based human activity recognition by learning initial motion data for virtual multi-sports. *2021 23rd International Conference on Advanced Communication Technology (ICACT)*, pages 373–375, 2021.

[36] Jennifer R. Kwapisz, Gary M. Weiss, and Samuel Moore. Activity recognition using cell phone accelerometers. *SIGKDD Explor.*, 12:74–82, 2011.

[37] S. K. Lakshmanaprabu, K. Shankar, M. Ilayaraja, Abdul Wahid Nasir, V. Vijayakumar, and Naveen K. Chilamkurti. Random forest for big data classification in the internet of things using optimal features. *International Journal of Machine Learning and Cybernetics*, 10:2609 – 2618, 2019.

[38] Oscar D Lara, Alfredo J Pérez, Miguel A Labrador, and José D Posada. Centinela: A human activity recognition system based on acceleration and vital sign data. *Pervasive and mobile computing*, 8(5):717–729, 2012.

[39] Song-Mi Lee, Sang Min Yoon, and Heeryon Cho. Human activity recognition from accelerometer data using convolutional neural network. *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 131–134, 2017.

[40] Jonathan Lester, Tanzeem Choudhury, Nicky Kern, Gaetano Borriello, and Blake Hannaford. A hybrid discriminative/generative approach for modeling human activities. *International Joint Conference on Artificial Intelligence*, 2005.

[41] Kah Sin Low and Swee Kheng Eng. Performance evaluation of deep learning techniques for human activity recognition system. *Journal of Physics: Conference Series*, 2641, 2023.

[42] Hugo Luís, Luís Garrote, and Urbano Jose C. Nunes. Human activity recognition for indoor robotics: A deep learning based approach using a human detection stage. *2021 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 43–49, 2021.

[43] Iveta Dirgová Luptáková, Martin Kubovčík, and Jiri Pospíchal. Wearable sensor-based human activity recognition with transformer model. *Sensors*, 22(20):1–16, 2022.

[44] Vittorio Mazzia, Simone Angarano, Francesco Salvetti, Federico Angelini, and Marcello Chiaberge. Action transformer: A self-attention model for short-time human action recognition. *Pattern Recognit.*, 124:108487, 2021.

[45] Sakorn Mekruksavanich and Anuchit Jitpattanakul. A lightweight deep residual network for recognizing activities in daily living using channel state information. *2023 IEEE 14th International Conference on Software Engineering and Service Science (ICSESS)*, pages 171–174, 2023.

[46] Violeta Mirchevska, Mitja Luštrek, and Matjaž Gams. Combining domain knowledge and machine learning for robust fall detection. *Expert Systems*, 31, 2014.

[47] Saeedi Mohsen, Ahmed Elkaseer, and Steffen G. Scholz. Human activity recognition using k-nearest neighbor machine learning algorithm. *Sustainable Design and Manufacturing*, 2021.

[48] Michael A. Nielsen. *Neural Networks and Deep Learning*. Determination Press, 2015.

[49] Bahareh Nikpour, Dimitrios Sinodinos, and Narges Armanfard. Deep reinforcement learning in human activity recognition: A survey and outlook. *IEEE transactions on neural networks and learning systems*, PP, 2024.

[50] Henry Friday Nweke, Teh Ying Wah, Ghulam Mujtaba, and Mohammed Ali Algaradi. Data fusion and multiple classifier systems for human activity detection and health monitoring: Review and open research directions. *Inf. Fusion*, 46:147–170, 2019.

[51] Shristi Pandey and Niraj Kumar. Enhancing human action recognition in high-resolution videos using convlstm and lrcn model. *2023 4th International Conference on Communication, Computing and Industry 6.0 (C216)*, pages 1–5, 2023.

[52] Thomas Phan. Improving activity recognition via automatic decision tree pruning. *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, 2014.

[53] I. Pires, Nuno M. Garcia, Nuno Pombo, and Francisco Flórez-Revuelta. From data acquisition to data fusion: A comprehensive review and a roadmap for the identification of activities of daily living using mobile devices. *Sensors (Basel, Switzerland)*, 16, 2016.

[54] Attila Reiss and Didier Stricker. Introducing a new benchmarked dataset for activity monitoring. *2012 16th International Symposium on Wearable Computers*, pages 108–109, 2012.

[55] Jorge Reyes-Ortiz, Davide Anguita, Alessandro Ghio, Luca Oneto, and Xavier Parra. Human Activity Recognition Using Smartphones. UCI Machine Learning Repository, 2013. DOI: https://doi.org/10.24432/C54S4K.

[56] Daniel Roggen, Alberto Calatroni, Long-Van Nguyen-Dinh, Ricardo Chavarriaga, and Hesam Sagha. OPPORTUNITY Activity Recognition. UCI Machine Learning Repository, 2010. DOI: https://doi.org/10.24432/C5M027.

[57] Pornthep Rojanavasu, Ponnipa Jantawong, Anuchit Jitpattanakul, and Sakorn Mekruksavanich. Improving inertial sensor-based human activity recognition using ensemble deep learning. *2023 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT & NCON)*, pages 488–492, 2023.

[58] Yoli Shavit and Itzik Klein. Boosting inertial-based human activity recognition with transformers. *IEEE Access*, 9:53540–53547, 2021.

[59] Ingo Steinwart and Andreas Christmann. Support vector machines. *Wiley Interdisciplinary Reviews: Computational Statistics*, 1, 2008.

[60] Guilherme Augusto Silva Surek, Laio Oriel Seman, Stéfano Frizzo Stefenon, Viviana Cocco Mariani, and Leandro dos Santos Coelho. Video-based human activity recognition using deep learning approaches. *Sensors (Basel, Switzerland)*, 23, 2023.

[61] Theo Theodoridis. EMG Physical Action Data Set. UCI Machine Learning Repository, 2011. DOI: https://doi.org/10.24432/C53W49.

[62] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Neural Information Processing Systems*, 2017.

[63] Huaijun Wang, Jing Zhao, Junhuai Li, Ling Tian, Pengjia Tu, Ting Cao, Yang An, Kan Wang, and Shancang Li. Wearable sensor-based human activity recognition using hybrid deep learning techniques. *Security and communication Networks*, 2020(1):2132138, 2020.

[64] Pengbo Wang, Yongqiang Zhang, and Wenting Jiang. Application of k-nearest neighbor (knn) algorithm for human action recognition. *2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, 4:492–496, 2021.

[65] Yan Wang, Shuang Cang, and Hongnian Yu. A survey on wearable sensor modality centred human activity recognition in health care. *Expert Syst. Appl.*, 137:167–190, 2019.

[66] Tuba Yilmaz, Robert Foster, and Yang Hao. Detecting vital signs with wearable wireless sensors. *Sensors (Basel, Switzerland)*, 10:10837 – 10862, 2010.

[67] Shibo Zhang, Yaxuan Li, Shen Zhang, Farzad Shahabi, Stephen Xia, Yuanbei Deng, and Nabil Alshurafa. Deep learning in human activity recognition with wearable sensors: A review on advances. *Sensors (Basel, Switzerland)*, 22, 2021.

[68] Fan Zhou, Ruomei Wang, Han Su, and Shenyi Xu. A human activity recognition model based on wearable sensor. *2022 9th International Conference on Digital Home (ICDH)*, pages 169–174, 2022.