Marquette University
e-Publications@Marquette

Dissertations (1934 -)

Dissertations, Theses, and Professional Projects

A Package Of Smartphone and Sensor-Based Objective Measurement Tools For Physical and Social Exertional Activities for Patients With Illness-limiting Capacities

Arafat Mahmood
Marquette University

Follow this and additional works at: https://epublications.marquette.edu/dissertations_mu

 Part of the Computer Sciences Commons

Recommended Citation

Mahmood, Arafat, "A Package Of Smartphone and Sensor-Based Objective Measurement Tools For Physical and Social Exertional Activities for Patients With Illness-limiting Capacities" (2023). *Dissertations (1934 -)*. 2569.
https://epublications.marquette.edu/dissertations_mu/2569

A Package Of Smartphone and Sensor-Based Objective Measurement Tools For Physical And Social Exertional Activities For Patients With Illness-limiting Capacities

by

Arafat Mahmood

A Dissertation submitted to the Faculty of the Graduate School,
Marquette University,
in Partial Fulfillment of the Requirements for
the Degree of Doctor of Philosophy

Milwaukee, Wisconsin

May 2023

A Package Of Smartphone and Sensor-Based Objective Measurement Tools For Physical And Social Exertional Activities For Patients With Illness-limiting Capacities

Arafat Mahmood

Department of Computer Science
Marquette University, 2023

Patients with several incompletely diagnosed and understood chronic diseases suffer from symptoms that limit their functional capacity. In particular, patients with chronic fatigue syndrome/myalgic encephalomyelitis (CFS/ME) and long covid syndromes have variable fatigue, malaise, poor and unrefreshing sleep, and delayed post-exertional exacerbations of these symptoms. There are no specific tests for these patients to diagnose their diseases properly. These patients must be aware of their daily activities and energy expenditure. Even a little physical effort or socially extroverted behavior can make them tired and incapable of continuing their daily routine. A comprehensive summary of the measured activities at any particular time of the day will eventually help the patients take precautions and prevent any unwanted physical or social burnout.

To address the current problems faced by these patients, we used a pair of smart insoles and a wrist-worn sensor-integrated device to detect and measure eleven daily living activities accurately. We developed a smartphone application and utilized the smartphone's microphone to analyze the recorded sound buffer and detect voice activities and high-level noise exposures. We also explored the smartphone's sensors to compare with the smart insoles' results.

This work presents a practical package for patients with limited functional capacities due to illness. We describe the design and development of a smartphone application that collects the data from the sensor device and microphone, sends them to a server for machine learning calculation, and provides a comprehensive summary by objectively measuring the activities. The application works as the core component of the package that connects all the other components via BlueTooth and uses the network protocol to send raw data to the server.

ACKNOWLEDGEMENTS

Arafat Mahmood, B.S.

I want to express my deep gratitude to my adviser, Dr. Sheikh Iqbal Ahamed, for his continuous support and feedback throughout my Ph.D. career. He always encouraged me throughout my research journey and directed me with his valuable advice and expertise.

I would also like to thank Dr. Praveen Madiraju and Rumi Ahmed Khan, M.D., for serving as members of my dissertation committee and for their time, comments, and feedback. I especially thank Dr. Richard R. Love for providing the directions in the initial stage of the research.

I also thank my current and previous colleagues from the Ubicomp Lab for their cooperation over the years. I am thankful to the wonderful community in Milwaukee for always helping me and being by my side. Last few years, I never felt the void of staying far from family because of these extraordinary people.

Finally, and most importantly, I would like to thank my parents, Abdul Halim and Rowshan Akter, who always dreamed big of me. I can never express how lucky I am to be their son. I can never thank them enough for their love and sacrifice for me my whole life. I am thankful to my sisters, Farjana Yesmin and Rejuana Yesmin, my brother Masud Rana, and my wife, Tanjina Zaman Shuchi, for their continuous support and inspiration. I am lucky to have them in my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1
Introduction

1.1 Problem Statement

Fatigue refers to a person's reduced capability or willingness to perform. Almost 38% US workers have reported that they have experienced some sign of fatigue [1]. Chronic fatigue syndrome is a long-term disabling disease that causes many debilitating symptoms [2]. There are four million individuals in the United States diagnosed with chronic fatigue syndrome / myalgic encephalomyelitis (CFS/ME), and 80% of them are women [3]. Long Covid, a similar emerging syndrome, may affect 4 to 13 million out of 43 million new cases [4]. Multiple sclerosis (MS), the most widespread disabling neurological disease among young adults, is estimated to affect almost 1 million people in the United States [5]. The treatment of fatigue is challenging and complex as there is no specific diagnostic test for patients suffering from fatigue. Moreover, patients with CFS/ME often face suspicion from healthcare professionals, and they oppose the conceptualization of their illness as psychologically treatable [6].

The major clinical manifestations of CFS/ME patients are severe chronic fatigue with muscle weakness, which gets worse by exertion, resulting in post-exertional malaise and non-restorative sleep [3]. Physical and social functioning with associated pain is the cardinal characteristics that distinguish CFS/ME patients from healthy control patients [7]. There are few objective and predictive measures for better understanding these symptoms. As daily variability in the symptom intensity is unpredictable, the cumulative activities' associated durations and intensity levels can provide insights into an individual's daily limit of functional activities. Any activity beyond a specific intensity or duration can cause an individual to "crash" the following day and severely limit social and physical activity. Monitoring patient activity in detail and evaluating the responses to patient interventions are essential to help the patients avoid next-day crashes.

The specificity and frequency of chronic fatigue symptoms and syndromes following Covid-19 infection is still unclear regarding specifics and their frequencies. However, it seems clear that at least 10% and as many as 30% of patients, regardless of the seriousness of their initial illnesses, develop persistent symptoms lasting six months or more [4]. Fatigue, pain, poor-non restorative sleep, and reduced physical activity are the most common symptoms. Over-exertion clearly delays the recovery process. Patients suffering from CFS are more likely to be depressed, unemployed, and distressed psychologically compared to those who experience fatigue but do not meet the criteria of being a CFS patient [8]. Excessive fatigue makes the symptoms worse. So patients need to be able to objectively measure their daily exertion to know their limits for physical and social activity.

1.2 Research Aims

We aim to build a software and hardware package measuring the duration of an individual's daily physical and social exertion and total energy expenditure. We need to collect physical and social activity data to measure duration and energy expenditure. Questionnaire-based subjective approaches to measuring energy expenditure are unreliable because of possible participant bias in reporting exertion. A more objective measurement of physical and social/cognitive activities is essential for patients. We extended and modified the system described in our previous work [9] for leg and hand activity detection with reasonable accuracy. The system also detects socially extroverted behavior and high noise level exposure. After conducting a comprehensive summary, we summarised the following steps to build a package that detects and measures the common leg, arm, hand, and low-level social exertional activities:

1) Detect leg activities with a pair of smart insoles by analyzing pressure sensor and accelerometer data.

2) Recognize hand activities using a wrist motion sensor device by analyzing accelerometer, gyroscope, and magnetometer data.

3) Observe social interaction from voice activity and surrounding noise captured using a smartphone microphone.

4) Develop a system with a smartphone, a pair of smart insoles, and a wrist motion sensor device to objectively measure the detected activities and analyze energy expenditure.

1.3 Dissertation Focus

This dissertation discusses how we developed a physical activity detection tool using a pair of smart insoles, a wrist-attached sensor-integrated device, and a smartphone application. We describe our analysis to detect voice activity and exposure to high noise levels. We show our result from analyzing and detecting eight emotions from human speech and seventeen environmental sounds to examine the effect of emotion and environment on human fatigue. Tackling different challenges in activity detection and comparison of other methods are described briefly. We describe the development of a package to detect and measure physical and social exertional activities, with the goal of providing users with a comprehensive report of their daily exertion.

1.4 Dissertation Organization

Chapter 2 of this dissertation provides a detailed description and comparison of recent wearable sensor-based activity detection research. This chapter also compares different sensors, sensor devices, and their position on the body. We describe two activity measurement techniques in this chapter. Chapter 3 describes the methods to process sensor data, different data processing techniques, popular machine learning algorithms, and their performance metrics. In chapter 4, we present the leg activity detection methodology and the results. Chapter 5 describes the hand activity detection process. Chapter 6 shows our analysis of the microphone sound buffer to detect voice and noise. We also analyzed the human emotion dataset and environment classification dataset. We deliver our result on the analysis in chapter 7. In chapter 8, we discuss the architec-

ture of our system, the challenges we faced while developing the system, and how we solved the challenges using technology. This section demonstrates the system, the flow to activity detection, and a comprehensive sample summary.

1.5 Publications

- **Arafat Mahmood**, Parama Sridevi, Sheikh Iqbal Ahamed: Demo Paper: A Package Of Objective Measurement Tools For Physical And Social Exertional Activities For Patients With Illness-limiting Capacities in ACM/IEEE International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE' 22), November 17–19, 2022, Washington, DC, USA

- Md Hasanul Aziz, Md Kamrul Hasan, **Arafat Mahmood**, Richard Love, Sheikh Iqbal Ahamed: Automated cardiac pulse cycle analysis from photoplethysmogram (PPG) signals generated from fingertip videos captured using a smartphone to measure blood hemoglobin levels (IEEE Journal of Biomedical and Health Informatics, 2021)

- **Arafat Mahmood**, Parama Sridevi, Sheikh Iqbal Ahamed: Sensor-Based Detection and Objective Measurement of Physical and Social Exertional Activities: State of the Art Review (JMIR Preprints. 10/24/2022:43793) (submitted)

- **Arafat Mahmood**, Parama Sridevi, Sheikh Iqbal Ahamed: Development of a Smartphone and Sensor-Based Detection and Objective Measurement Tools for Physical and Social Exertional Activities(IEEE Systems Journal) (submitted)

- **Arafat Mahmood**, Parama Sridevi, Sheikh Iqbal Ahamed: Design and Implementation of a Sensor-Based Remote Exertion Monitoring System For Elderly Patients (IEEE Access) (submitted)

- **Arafat Mahmood**, Parama Sridevi, Sheikh Iqbal Ahamed: Smartphone and Sensor-based Objective Measurement Tools for Physical and Social Exertional Activities

for Patients with Illness-limiting Capacities (Klingler College of Arts & Science Poster Session, 2022)

Chapter 2
Literature Review

2.1 Overview of Activity Detection Methods

Two main categories are found in the literature to approach physical activity detection: sensor-based and vision-based activity detection [10, 11]. Vision-based techniques rely on sensing videos provided by CCTV and cameras. These techniques use a camera that is attached to the environment. The camera records surrounding videos and applies computer vision techniques to the captured images and videos. The accuracy of the detection process depends on how well it works with noises, such as low image quality and light issues. The major drawback of this approach is that users have severe privacy concerns about maintaining a regular life in front of cameras [12].

Sensor-based techniques can be divided into three sub-categories as shown in Figure 2.1. These include i) Wearable Devices, ii) Object-tagged, and iii) Dense sensing. Wearable device approaches have been widespread in recent years. This approach asks users to wear or attach a sensor-based device to their bodies. Sensors are attached to different locations on the body of a user, and return raw data for further analysis of the activity. Accelerometer, gyroscope, and magnetometer are common sensors attached to the dominant wrist, waist, dominant hip, or chest for data collection [13]. The main problem of this approach is that users may forget to wear the sensor or device [14].

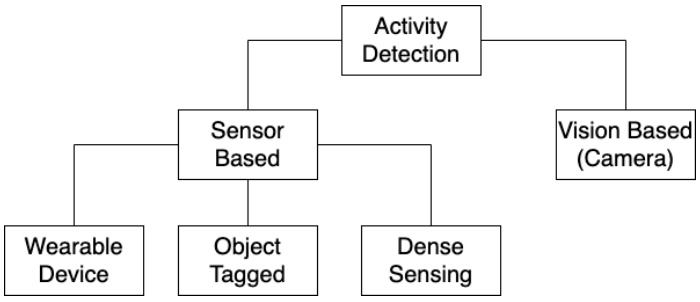


Figure 2.1: Activity Detection Methods classification

Object-tagged approaches solve this issue by attaching the sensors to objects of daily use instead of body locations. One of the most common object-tagged approaches is using smartphone sensors for activity recognition techniques. Due to the recent advancement in sensor technology, smartphones are equipped with a wide range of sensors. People usually carry their smartphones all the time. Data collection process and detection are much more manageable with smartphones using the integrated sensors [15, 16]. Smart Insoles, equipped with an accelerometer, gyroscope, and pressure sensors, are used in different health care research and sports industry research to detect sedentary activities, as well as different leg activities, and gait analysis in [17, 18, 19]. Object-tagged approaches have also been used to track drinking behavior in [20] using accelerometer and health care activity monitoring in [21] using RFID. The problem with the object-tagged approach is that the user needs to carry or be around that object all the time, which may not be feasible in some scenarios. Also, due to the setup cost and challenges, object-tagged sensors are less used than wearable sensors [11].

The third sensor-based technique is called dense-sensing, a device-free approach to solve the problems in wearable sensor and object-tagged approaches. In a device-free approach, users are not required to carry a device or sensor. This technique deploys different sensors such as sound, motion, radars, temperature, pressure, and proximity sensors in the environment. The sensors capture the relative information when the user performs any activity near the sensors. Several researchers have used dense sensing methods to detect physical activities in a smart home environment [22, 23]. Deployment of dense-sense-based sensors in the environment is comparatively more complex than the other approaches. Moreover, sensor data from the dense-sensing approach are affected by environmental noise which requires mitigation to increase the algorithm accuracy [13]. For this reason, dense-based sensors are yet to be ready to use in real-life scenarios.

2.2 Physical Activity Detection Methods

Physical Activity Detection Methods Detection of common activities related to the leg, such as standing, running, and walking, are detected using a tri-axial accelerometer. Subjects wear a device integrated with an accelerometer on their waist [24, 25], hip [26], or pocket [27, 15]. Smartphones are widely used as data collection devices [28, 29, 30, 16]. Use of smart-insoles [31, 18, 19] and smartwatches [32, 33, 34] are also becoming popular as they feature a multi-sensor data collection environment. The hand activities are independent of typical body activities like walking, standing, and running. People can do something with their hands while continuing their regular body activities. A system should know the hand activities, too, to get proper activity recognition. The smartphone or other wearable sensors placed on the waist, hip, or pocket cannot detect the movement of hand or wrist activities. As a result, wrist-worn devices are used widely to detect various hand activities and gestures such as eating, drinking, writing, and typing [35, 36]. Hand-washing activity was detected using the wrist-worn device in [37]. Ramos-Garcia et al. [38] used wrist-worn accelerometers and gyroscopes to detect eating activity. As accelerometers are used to get an object's linear movement, they effectively detect human linear movements, but fail to capture circular hand motions. A combination of accelerometer and gyroscopes works are used to capture both the linear and rotational movements [39]. After data collection with appropriate sensor setup, machine learning algorithms are deployed to extract necessary features and develop a prediction model from the sensor data.

2.2.1 Smartphone based solution

There is a growing number of smartphone applications available for human activity detection. These applications take advantage of the in-built sensors to collect activity and context information and can detect activity in real-time. Smartphone sensors can be configured at required sampling rate. The captured sensor data can be stored in the local storage and pre-processed for further calculation and detection. Voicu et

al. [15] detected users' physical activity using the sensors of smartphones. They collected data using an accelerometer, gyroscope, and gravity sensor, which are integrated into most modern smartphones. They trained the classifier with a large dataset after extracting necessary features from the data. They used two different datasets: internal and external. The authors collected and created the internal dataset for their research and collected the external dataset from the internet. With the internal dataset, they got good accuracy for walking (93%), sitting (94%), and standing (93%), but low accuracy for climbing upstairs (75%) and downstairs (71%). There was relatively low accuracy for the external dataset because of noise in the data and the differences in the environment while collecting the data.

The common problem with machine learning approaches is that they use too many systems resources [15]. Running such costly algorithms on a mobile device is not feasible as it drains the battery. Morillo et al. [40] provided an innovative approach to detect physical activity by extracting discrete variables from smartphone accelerometer data. They addressed major drawbacks related to smartphone energy consumption and computational cost and proposed a way to overcome these issues. The use of the Ameva algorithm for discretization helps reduce the recognition process computational cost. They achieved an average accuracy of 98% for recognizing eight types of activities. Filios et al. [30] introduce a hierarchical detection algorithm with a two-layer system for daily activity recognition. They used a smartphone accelerometer for motion detection and a microphone for environment detection in the lower layer of the two-layer system. In the second layer, they created the activity detection model by taking the previous layer's motion detection model and environment detection model as input. Four machine learning algorithms were used to classify the data: Logistic Model Tree (LMT), J48, IBk, and FT. They got the best accuracy of 92.13% from the J48 decision tree algorithm for the motion detection model, 90.77% accuracy from the IBk classifier for the environment detection model, and 94.57% accuracy from the FT classifier for the activity detection

model. The hierarchical system can be extended to more levels to detect collaborative activities.

2.2.2 Wearable sensor-based solution

Wearable sensor approaches use data from sensors that are worn on different body parts resulting in efficient detection of human activity. Wearable sensors are small, so they can easily be attached to a human body. With high sensitivity and strong anti-interference ability, they are more suitable for practical situations [25]. Smart device manufacturers like Apple, FitBit, and Nike+ have released some wearable sensor-based products that are successful in activity recognition. However, the user must explicitly select a new activity session for reliable functionality [36]. Pansiot et al. [41] integrated an ear-worn activity recognition (e-AR) sensor with ambient blob-based vision sensors. They preprocessed the sensor data for dimensionality reduction and then applied Gaussian Mixture Model Bayes Classifier to the data. The fusion of the ambient sensor with the e-AR sensor significantly improved the accuracy in detecting activities compared to using only the e-AR sensor. For example, the walking detection had only 79% accuracy when they used the e-AR sensor alone, but it improved to 100% when they applied the sensor fusion. One of the significant challenges of using sensor-based techniques is that so many factors directly affect a person's behavior in performing different activities [42]. Also, the algorithms to detect the activities must be robust against noise to filter out noisy sensor data. Yazdansepas et al. [42] addressed these challenges and comprehensively studied different machine learning techniques for activity detection. To collect the data, they used a single hip worn ActiGraph GT3X+ attached to participants' non-dominant hip and asked them to complete a series of predefined activities. The authors extracted 176 features from the time and frequency domains. They used three feature selection methods to compare six machine learning algorithms: Random Forest, K-Nearest Neighbor, Decision Tree, Multi-Layer Perceptron, Support Vector Machine, and Naive Bayes. Random Forest was the best classifier with an average accuracy

of 83.7%.

Due to the noisy nature of accelerometer data, machine learning algorithms have some performance issues. Moreover, the accelerometers cannot differentiate between activities of similar patterns alone, such as climbing stairs up and down [43]. To solve this, Nweke et al. [43] used a sensor-fusion of accelerometer and gyroscope and compared the performance of seven different classification algorithms using a single sensor and the sensor-fusion placed at a different body part. They found a positive impact of sensor fusion in recognizing human activities. Zhu et al. [44] proposed a human activity recognition system using the fusion data from two wearable sensors. They used neural networks and hidden markov models to reduce the computation load. The system classified human activities into three types: zero displacement activities, transitional activities, and strong displacement activities. Kawser et al. [18] integrated pressure sensors in shoes and used smartphone accelerometers and gyroscope data to build a system prototype for novel activity detection. Placing pressure sensors in shoes removes the requirement to carry or wear any device. Their system worked in two phases: the Learning Phase and the Recognition Phase. The data was collected and analyzed in the learning phase to develop the algorithm. In the recognition phase, the algorithm was applied to real-time incoming sensor data to detect activities. The authors applied a decision tree algorithm to develop four different classifiers based on the value from the accelerometer, gyroscope, left shoe, and right shoe. The final activity was decided using a majority voting algorithm from the output of different classifiers.

2.3 Choice of Sensors and Body Positions

Table 2.1 shows different sensors used in the literature for human activity recognition. Accelerometers are the most common sensors used in human activity recognition systems. They are common in all sensor-based activity detection systems. Gyroscopes are often combined with accelerometers to detect rotational movements. Accelerometer and gyroscopes are often combined to make a 6-axis (or 9-axis with magnetome-

Reference	Sensors	Device	Position	Detected Activities
[24]	Accelerometer, Gyroscope	Smartphone	Waist	Sitting, Lying, Standing, Stair up, Stair down, Walk
[25]	Accelerometer, Gyroscope	Smartphone	Waist	Same as above, sit to stand, sit to lie
[27]	Accelerometer	3-axis accelerometer	Pocket	Walking, Running, Jumping
[15]	Accelerometer, Gyroscope, Gravity Sensor	Smartphone	Pocket	Walking, Running, Sitting, Standing, Upstairs, Downstairs
[18]	Accelerometer, Gyroscope, Pressure Sensor	Smartphone, Insole	Pocket, Shoe	Running, Walking, Standing, Sitting
[17]	Accelerometer, Inertial Measurement Unit, Force Sensing Resistor	Insole	Shoe	Fall Detection
[45]	Accelerometer	Sport Watch	Wrist	Walking, Running, Sitting, Standing, Lying
[46]	Accelerometer	Sport Watch	Wrist	Walking, Sitting, Standing, Lying, Stairs
[35]	Accelerometer, Gyroscope	Watch	Wrist	Eating period from hand gesture
[47]	Accelerometer, Gyroscope	Two smartphones	Pocket, Wrist	Lying, Sitting, Standing, Walking, Stairs, Writing, Typing, Eating etc.
[48]	9-axis IMU	Wristband	Wrist, Arm	Smoking Gesture
[49]	Accelerometer	Smartwatch	Wrist	25 hand activities and gestures
[50]	IMU with Accelerometer, Gyroscope	Ring,Bracelet	Wrist, Fingers	Activity of daily livings

Table 2.1. Position of different sensor devices used in literature

ter) Inertial Measuring Unit (IMU) and placed on one or more body positions of the users [50, 17].

In different studies, the accelerometer and other sensors were set at a predefined sampling rate of 15Hz-60Hz [24, 35, 45]. Approximately 20 Hz is a standard sampling rate for normal human activities, but it can be optimized for better performance [51]. However, according to [52], 99% of the power for walking activity contains below 15 Hz. So, the sampling rate must be at least 20 Hz to avoid under-sampling. The most feasible position to keep the smartphone is the pocket, as pockets are the most common place people carry their smartphones in. For some studies, researchers attached the smartphone to the waist or kept it in hand, but this is not always feasible as a real-life solution. Instead of a smartphone, a smartwatch or wristband can be more practical where the requirements are to get information from hand movement, as they can be easily attached to the wrist. The authors in [38] used a Hidden Markov Model and recognized eating activities. They also divided the activity into sub-activities like resting, eating, drinking, and others, with an accuracy of 84.3%. In [47], Shoaib et al. used an accelerometer, gyroscope, and linear acceleration sensor at two different positions (wrist and pocket) and concluded that the combination of these two positions shows better results than using only the wrist position. In [48], the authors set up an arm trajectory-based method using a combination of an accelerometer, gyroscope, and magnetometer at the wrist position to extract the subject's hand-to-mouth gestures to detect smoking sessions with high accuracy of 95.7%.

In [50], Moschetti et al. placed wearable inertial sensors on the dominant hand's fingers, the back of the hand, and the wrist. They collected data from different sensors for nine different gestures, such as eating with the hand, drinking from a glass, answering the telephone, and brushing with a toothbrush. They used six different combinations of sensors and applied two different machine-learning algorithms for the classification. The authors in [49] noted some distinctions between hand actions and hand activities.

They defined activities as a series of hand actions lasting seconds or minutes. They decided to detect hand activities instead of hand actions for better classification. They modified the kernel of a smartwatch [49] and configured the frequency of the accelerometer. The authors classified 25 hand activities with 95.2% accuracy.

Maurer et al. [53] designed a system to recognize and monitor activities using an eWatch and multiple sensors attached to different body positions. They studied the effectiveness of different activity classifiers by varying the wearing positions of the sensors. The sensors were placed on the shirt, pocket, bag, necklace, and wrist. The authors calculated the classification accuracy for every activity in these six positions. They got the highest accuracy when data from all sensors were used to extract the features. However, there is a trade-off related to performance. The number of sensors increases the data processing time and system resource use. Hanai et al. [54] claimed that increasing the number of sensors reduces the system's efficiency. So, they proposed a new feature selection technique for 3D acceleration signals. They proposed 1D Haar-like and 1D Haar-like biaxial filtering for efficient feature selection and activity recognition. They achieved the highest accuracy of 93.91% and reduced the calculation cost to 21.22%.

2.4 Social Activity Detection Methods

Social behaviors and interactions play essential roles in a person's fatigue. The detection of a person being in a conversation and measuring the involvement level can provide valuable insights into that person's social activity. There is evidence that extroverted behaviors positively affect an individual's current mode and lower fatigue but leads to higher fatigue in a 3-hour delay [55]. Human behaviors and interaction with other persons are vital in the health sector. A person's surrounding audio data provides valuable information about the context. To detect the type of interaction, we can analyze surrounding audio, phone calls, text messages, emails, and other communication methods [56]. Microphones integrated into smartphones are a great way to analyze the surrounding of a user and detect the type of social situation the user is in. Several re-

searchers have detected the type of context by analyzing surrounding sound. Lu et al. [57] developed the first voice-based stress detection system by using microphones in the smartphone. Abdullah et al. [58] used smartphone sensors to collect behavioral data like speech, SMS, call logs, and contextual data like location. They applied machine learning to the data that distinguished between the stable and unstable states in Bipolar Disorder patient's daily life to reduce the risk of relapse.

Smartwatches and smart wristbands can also collect audio data from surrounding. Smartwatches were used to diagnose Parkinson's disease [59], Dysarthria [60], and extract clinical features from speech [61]. A smartwatch-based conversation and noise classification system was developed in [62], which continuously monitored and measured conversational events. Additionally, there has been research on detecting and measuring conversational speech and segmenting each speaker in [62, 56].

2.5 Methods of Activity Measurement

Measurements of physical activity are done using two methods: Subjective and Objective. Subjective methods are based on user perception (recalling or remembering), so there is a chance of having reporting bias in the measurement. Subjective methods include self-report questionnaires and maintaining an activity diary/log. According to [63], participants in subject studies sometimes over-report the quantity of their activity. Objective methods depend on data from devices or observations, which makes the method more accurate than subjective methods. Subjective methods are mainly questionnaires or based on participants' recall ability. On the other hand, objective measures make the observation and quantification using devices that return reliable data. Using different sensors and getting a defined measurement is an example of an objective measure. Heart rate monitors and pedometers are also used for objective activity measurement. Figure 2.2 shows different activity measurement methods.

Objective methods are now regarded as the better method for measuring physical activity level and intensity [63]. Accelerometers provide reliable data that can be used

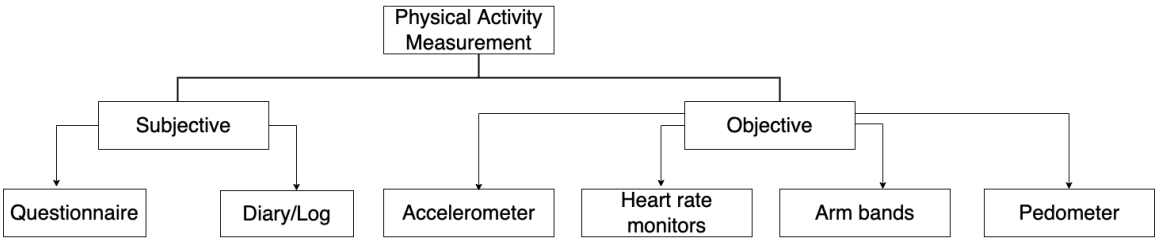


Figure 2.2: Different type of activity measurement methods

to quantify performed activities. Objective methods were developed to reduce the issue caused by subjective methods, and they can avoid reporting bias from the participant while presenting the relationship between activity level and health. There have been discrepancies when comparing the result of subjective and objective methods [64]. The self-reported subjective measure reports a higher amount of physical activity than what was reported by the accelerometers in [65, 66]. Based on these facts, it is evident that to get an accurate measurement of physical activity, an objective method is necessary.

Table 2.2 shows previous literature on managing fatigue and estimating energy expenditure. The recent systems are sensor-based and thus objective and more accurate than the subjective methods to assess fatigue and calculate energy expenditure. The objective systems use wearable sensors, IMU, and accelerometers to detect the severity of CFS/ME patients [67] or to assess their symptoms of fatigue [68]. Using multiple wearable sensors in different body positions may lead to accurate results, but it reduces the comfortability of elderly patients. Also, they did not consider social exertion's effect on assessing the patients' fatigue. Subjective studies have used standard questionnaires to get the perceived exertion of the patients, which leads to bias in the reporting. Most objective and sensor-based MET estimations used regression and approximation-based techniques on predefined activities without any activity detection system. In [69], thirteen activities were detected using multiple wearable sensors on the hip, waist, and chest to calculate energy expenditure. But the use of numerous wearable sensors made the system complex and uncomfortable for the patients.

Reference	Method	Tool	Activity Detection	Social Exertion	Remarks
[67]	Objective	IMU	No	No	CFS/ME severity determination
[68]	Subjective, Objective	Questionnaire, Accelerometer	No	No	Assessment of fatigue
[70]	Objective	Multiple IMU	No	No	Reduces comfortability
[71]	Objective	Multiple IMU	No	No	Fall risk assessment
[72]	Subjective	Questionnaire	No	No	Self-reported assessment, bias in reporting
[73]	Objective	Sensor	No	No	Regression-based MET estimation
[74]	Objective	Sensor	No	No	MET calculations from predefined activities
[69]	Objective	Wearable sensor	Yes	No	Multiple body location

Table 2.2. Literature on fatigue management and EE estimation

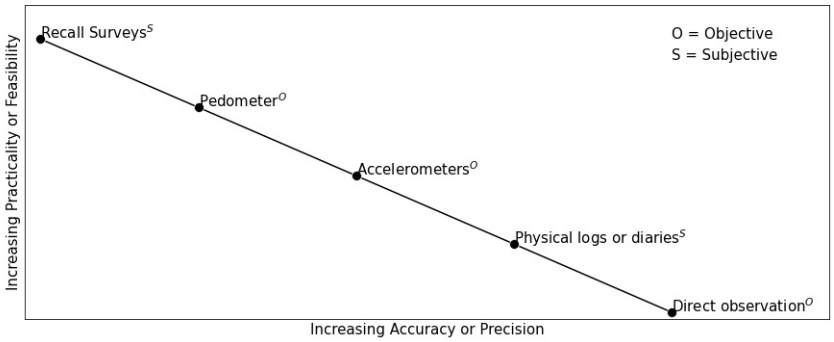


Figure 2.3: Practicality Vs Accuracy in the context of physical activity measurement

There is a trade-off between feasibility and accuracy in the context of the subject and object measurement of physical activity, as depicted in Figure 2.3. Pedometers and Recall surveys are more practical measures of physical activity, but they are less accurate than using an accelerometer or maintaining a physical log or diary [75].

Chapter 3
Processing Sensor Data in Activity Detection Context

Sensors are an essential part of an activity detection system. The system's accuracy depends on how well the data is processed and the extraction of appropriate features. Data from the sensors need to be segmented into windows to match the sampling rate with human activity movement patterns. This chapter describes standard sensors, preprocessing techniques, and the performance metrics used in activity detection systems.

3.1 Sensors in Activity Detection

Accelerometers and Gyroscopes are the two most common sensors used to learn the movement patterns during physical activities. Sensor fusion algorithms are applied to the raw output from sensor data for better performance. This section will briefly describe the essential sensors we need to build an activity detection system.

3.1.1 Accelerometer

A tri-axial accelerometer can sense the movement of the object or body it is attached to and return the linear acceleration along the x, y, and z axes, which results in valuable information about movement patterns. Accelerometers are good at measuring linear acceleration but cannot measure rotational movement. Accelerometers perform very well in recognizing running, walking, standing, and sitting but poorly detect stair-climbing activities.

3.1.2 Gyroscope

A tri-axial gyroscope can measure angular velocity and sense the change in orientation of an object along the x(roll), y(pitch), and z(yaw) axes. Gyroscopes can provide essential insights into the movement of an object that rotates around some particular axes. Though gyroscopes detect immediate orientation change, they are subject to a

drift in orientation values that causes gradual divergence of calculated values from original values over time [76]. Accelerometers can correct the drift of roll and pitch, while a 3-axis magnetometer can compensate for the drift in yaw by reading geomagnetic flux density [76].

3.1.3 Magnetometer

Magnetometers detect and measure the fluctuations in the earth's magnetic fields. The data from magnetometers are fused with other inertial sensors to calculate absolute heading. A 3-axis magnetometer can detect the changes in the magnetic field caused by human movement and can be helpful in activity detection research.

3.1.4 IMU

An individual sensor can only return linear measurements or rotational measurements along some axes, which may need to be sufficient for complex activity detection. A combination of different inertial sensors might help in this scenario. A 6-axis combination of accelerometer and gyroscope and a 9-axis combination of accelerometer, gyroscope, and magnetometer is used as an Inertial Measurement Unit (IMU) to compensate for the issues of different sensors [77]. IMUs can measure acceleration, angular velocity, and magnetic fields from the sensor package. Sensor-fusion algorithms are used to combine data from different sensors of IMU by fusing the data to create a new high-dimensional feature set [78].

3.2 Multi-Sensor Fusion

The fusion of sensors can be done by combining the output of different sensors to extract new features. Multi-sensor fusion can achieve better accuracy and performance in an activity recognition system than a single sensor [53]. The data from a single sensor might be unreliable, so using multiple sensors can benefit human activity recognition research. Data from multiple sensors can be fused in three levels of the recognition system.

3.2.1 Data-level fusion

In the most common sensor fusion technique, raw data from multiple sensors is fused to generate new feature data. Fusing data from various sensors generates huge feature space that may increase the accuracy of the classifier but will reduce the performance in terms of resource usage and time. Data level fusions are done at the early stage of a recognition system.

3.2.2 Feature-level fusion

In feature-level fusion, feature sets from multiple sources are combined into a single feature set by applying appropriate schemes. Different feature transformation and reduction techniques can be used to detect correlated features and generate a compact set of features to improve the performance of the trained classifiers. However, large training sets are usually required to generate the most crucial feature set.

3.2.3 Decision-level fusion

Decision-level fusions are done at the classification stage of a recognition system. It combines the decisions of multiple classifiers from multiple modalities and helps to reach a final decision. Multi-modal sensor data are typically analyzed using multiple classifiers, especially when applying a feature-level fusion on the data is difficult. Each sensor modality first decides on the data, and a decision fusion algorithm combines all the decisions.

3.3 Steps in Activity Detection

The activity detection process consists of four essential components found in the previous research [10] as shown in Figure 3.1. The first task is selecting a proper approach depending on the needs and target users. The appropriate sensors or devices are deployed in environments or tagged with objects for data collection. After collection in the second step, the data is pre-processed to remove noise, and then proper feature extraction techniques are applied. Machine learning approaches are the most common

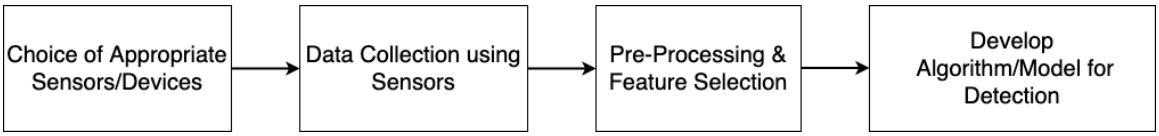


Figure 3.1: Tasks of Activity Recognition

method to recognize activities from processed data. Using machine learning algorithms is the last step in the activity detection process.

3.4 Preprocessing and Feature Extraction

Sensor data typically contains high-frequency noise. They are usually preprocessed before the actual classification. High-frequency noise can be removed using a low-pass filter [79], normal filter [80], and median filter [81]. Features of sensor data can be divided into two domains: time domain and frequency domain. Time domain features include mean, median, standard deviation, variance, the time between peaks [82], RMS [83], range between maximum and minimum values [84], angular velocity, etc. These time domain features are prone to measurement and calibration errors [42]. As noise in accelerometer signals cause the signals to jump from high frequencies to low, analyzing the signals from a frequency domain perspective allows filtering that noise to make cleaner signals. Frequency domain features include coefficients sum calculated using wavelet analysis and Fourier transformations, DC component analysis [82, 85] and dominant frequency analysis [86] using Fourier transformations. A combination of both domains was also used in [87, 42] to achieve maximum performance.

The accelerometer and other sensors are set at a predefined sampling rate in different studies. Approximately 20 Hz is a standard sampling rate for normal human activities, but it can be optimized for better performance [88]. However, according to [52], 99% of the power for walking activity contains below 15 Hz. So the sampling rate must be at least 20 Hz to avoid undersampling.

3.5 Segmentation into Windows

As activities are performed over a long period compared to their sampling rate, it is essential to segment the samples into data windows or smaller data segments for further processing. When people walk, their bodies follow a specific movement pattern, which cannot be expressed in a single row of raw sensor data. So data from multiple rows are combined in a 'window' to represent a small portion of the activity pattern. The segmentation process usually depends on the requirements and end goals of the activity recognition system. Some standard segmentation techniques used in literature are activity-defined window segmentation [89, 27, 51], event-defined window segmentation [90, 91], and sliding windows [92, 82, 93].

Activity-defined windowing procedure partitions the data stream based on activity change. It detects the start and end points of each activity. The changes can be identified through the frequency analysis of the sensor data stream. User feedback is also a great way to detect the start and end points of the activity. The event-defined windowing procedure uses specific events to segment the data. This approach has no defined segment size, as the events are not uniformly distributed. The activity and event-defined windowing techniques segment data using defined conditions but require complex preprocessing steps [94].

The Sliding Window Technique is the most popular data segmentation technique for its simplicity and intuitiveness. The technique segments the data into small fixed-size windows using overlapping and non-overlapping methods. Though the non-overlapping windows method follows a more straightforward approach, a relatively complex overlapping algorithm is used in practice. The overlapping approach overlaps two consecutive segments' data by a defined percentage to avoid potential information loss between adjacent windows. Literature suggests using 25% to 50% overlap in this approach [29]. Besides deciding on the percentage, determining the best window size is also essential. Windows may have information on multiple windows if the window is too large. On the

other hand, if the window is too small, some activities may be split into multiple windows. Wang et al. [95] analyzed different sizes of timed windows based on the F1 Score and concluded that a window size of 2.5s-3.5s can provide the best result for motion recognition.

3.6 Feature Selection

In machine learning studies, some extracted features may be irrelevant and do not contain useful information. Training and computation get slower if there are too many features. So it is essential to reduce and select necessary features using the dimensionality reduction technique before running the actual classification techniques. There are two types of reduction process [96]: feature selection and feature transform.

The feature selection method creates a subset of the original features. The selected features contain the most relevant information contributing to the classifier's performance. There are three different feature selection techniques depending on the feature evaluation process [97]: filter, wrapper, and embedded methods. Filter methods perform the selection task by analyzing the characteristics of the features without communicating with the classifier. Unlike filter methods, wrapper methods use the classifier to evaluate different subsets of features to select the best possible subsets. Embedded methods calculate the optimal weights from a feature subset and choose the best one. Filter methods are the most used in high-dimensional feature space because of their high computational efficiency, but some hybrid strategies are also combined for better selection methods [98].

The feature transformation technique combines original features to make a smaller dimensional space. It works well where multiple features collectively can provide better performance of the classifier. Some widely used feature transformation techniques are Principal Component Analysis (PCA) [99, 29], Independent Component Analysis (ICA) [100], and Local Discriminant Analysis (LDA) [101, 102].

Group	Algorithm	Reference
Generative Model	Bayesian Network	[40, 99, 93]
	Hidden Markov Model	[85, 44, 38, 37]
	Naive Bayes	[53, 40, 42, 35]
	Gaussian Mixture Model	[19, 41, 18]
Discriminative Model	Random Forest	[42, 50, 48]
	J48	[99, 30]
	C 4.5	[53, 85, 50, 45]
	Logistic regression	[104, 56, 58, 15]
	Support vector machine (SVM)	[50, 17, 16, 42, 40]
	K-nearest neighbors (KNN)	[93, 53, 40, 16, 46]
	Convolutional neural network	[25, 36]
	Multilayer perceptron	[42, 45, 46]

Table 3.1. Popular classification techniques used in literature.

3.7 Classification

Sensor-based data classification techniques can be grouped into generative and discriminative models. The dataset is fed into different ML algorithms to analyze the sensor data and then recognize a pattern from the data. Table 5.1 shows a list of popular ML techniques used in literature.

Nam et al. [93] compared different ML techniques on their dataset and found K-nearest neighbor (96.2%) and Decision Trees (94.7%) techniques to be highly accurate with good computation complexity. They also got 95% accuracy with SVM and 96.3% accuracy using MLP, but they took high computational time, suggesting that these algorithms are unsuitable for real-time recognition. In recent years, decision trees (98.4%) and deep learning-based algorithms(95.2%) have been popular among researchers [103].

3.8 Performance Metrics

Accuracy, precision, recall, F-measure, and specificity are the most used performance metrics of machine learning classifiers in activity recognition systems [50, 78]. These measures depend on the count of:

- True Positives(TP): the number of data correctly classified as positive

- True Negatives(TN): the number of data correctly classified as negative
- False Positives(FP): the number of data incorrectly classified as positive
- False Negatives(FN): the number of data incorrectly classified as negative

3.8.1 Accuracy

Accuracy is used to provide a summary of the performance of the classifiers for all classes. Accuracy can be calculated as the total number of true positives and negatives ratio to all prediction counts.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.1}$$

3.8.2 Precision

Precision quantifies the number of correct positive predictions and measures how well a classifier predicts a specific class. The precision of a class can be calculated with the following formula:

$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

3.8.3 Recall

The recall measures the classifier's ability to identify true positives correctly. The difference between precision and recall is that precision measures the rate of detecting true positives out of all positive predictions. In contrast, recall measures the rate of detecting true positives out of all possible positive predictions. Recall can be measured with the following formula:

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

3.8.4 F-measure

F-measure combines both precision and recall into a single measure to explain both properties by taking the harmonic mean of them using the following formula:

$$F-measure = 2.\frac{Precision.Recall}{Precision + Recall} \tag{3.4}$$

3.8.5 Specificity

Specificity measures the ratio of correct negatives out of all possible negative predictions.

$$Specificity = \frac{TN}{TN + FP} \tag{3.5}$$

3.8.6 RMSE

Root mean squared error (RMSE) is used widely to evaluate the quality of predictions by machine learning systems. RMSE calculates the euclidean distances of the predicted values and the original values. RMSE can be calculated from the following formula:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(prediction - original)^2}{n}} \tag{3.6}$$

Chapter 4
Detection of Leg Activities from sensor data

Machine learning algorithms play a vital role in detecting human physical activities. Researchers have collected data from different activities to develop machine-learning models. Recent research on activity detection are mostly sensor-based. With the advancement in sensor technology, manufacturers have integrated modern sensors into smartphones, insoles, smartwatches, and many other objects of daily use. The necessity of the smartphone has made it one of the most used devices to collect physical activity data used to train machine learning models [28, 40, 16, 15]. Researchers have used insoles to detect the pattern of different foot activity in gait analysis and fall detection research [18, 19]. Accelerometers and gyroscopes are the most crucial sensors in any activity detection research that respectively returns the change in speed and orientation of an object along the x,y, and z-axis. The data can be further analyzed to extract essential features in activity detection.

4.1 Insole as a solution

Using different body sensors on different locations on the body might increase the accuracy of activity detection, but can create discomfort for patients. Patients may forget to wear the sensor device, leading to incorrect energy expenditure measurement. Patients dependent on such measurement systems may need a different estimate of their daily exertion and mark their daily limit for activities. Researchers tried to address these issues [18, 17] and used sensor-integrated smart insoles to put inside the shoe to remove the complexities of multi-sensor systems. An insole setup as a data collection tool ensures that patients are less likely to forget to wear the sensor devices and therefore be able to avoid the discomfort of wearing multiple sensors on the body.

4.2 Methodology

Sensor-based approaches detect activities by utilizing machine learning algorithms on the data returned by sensors. Machine learning algorithms use raw data from sensors as features. Sensor data are segmented using a fixed window size, then basic statistical features like mean, median, and mode are extracted from each window. Segmentation is a fundamental and standard technique in activity detection. A sensor with a sampling frequency of 20Hz will have 200 rows for an activity(let X) performed for 10 seconds. So for activity X, these 200 rows are segmented and treated as a single row in a formatted dataset. The segmented dataset is used to train and validate the machine learning algorithms. We developed our leg activity detection algorithm by following the above techniques.

4.2.1 Data Collection

We used a pair of smart insoles to collect the data (Figure 4.1). The insoles had four pressure sensors and an accelerometer. We put smart insoles into the shoes instead of regular insoles. The accelerometer returns raw data along the three axes, and each pressure sensor returns 1 or 0 depending on the pressure on that sensor. Data from 6 different activities were collected using the foot insoles. The target activities are Walking, Running, Sitting, Standing, Upstairs, and Downstairs. We collected the data at a sampling rate of 20 Hz from both insoles. Participants performed each activity for 30 seconds in five different sessions.

We developed an android application to perform the data collection. The application connects to the insoles via BlueTooth. After establishing the device connection, the application asks the user to perform certain activities for calibration purposes. The insoles are ready to send raw sensor data to the smartphone once the application establishes a connection. For the data collection, the application asks the user to perform each target activity, and a timer of 30 seconds starts when the user presses the start button for each activity. After completing each 30-second data collection activity, the

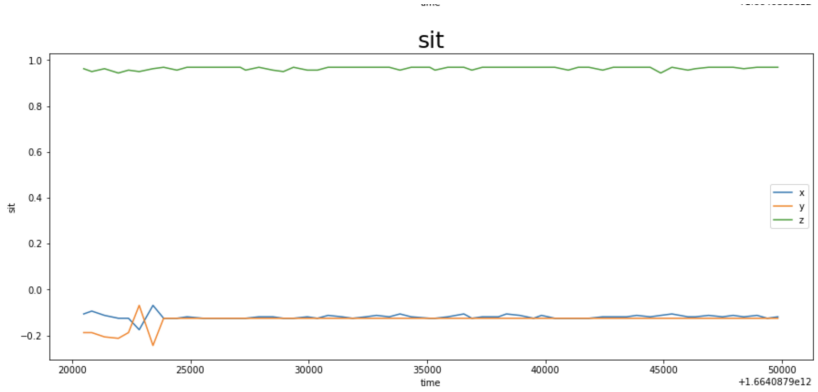
application stores the data for each activity in a comma-separated value format with timestamps and labels in the smartphone storage. After the user completes the data collection process, we move all the data files from the smartphone storage for analysis.

4.2.2 Data Representation

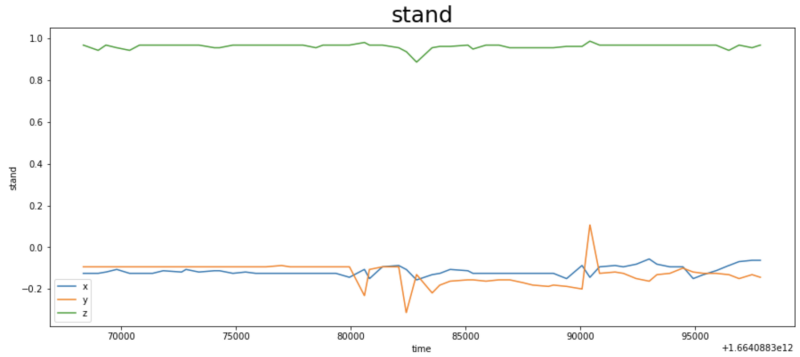
Figure 4.2 shows the accelerometer values along the x,y, and z axis for different leg activities. There is no movement along all three axes in a "sitting" activity (Figure 4.2(a)) and a little movement in the event of "standing" (Figure 4.2(c)). There is some expected movement for "running" (Figure 4.2(c)) and "walking" (Figure 4.2(d)) because of the high frequency of movement of the leg while performing the activities. One important difference we found is that the z-axis values were higher in "running" than in "walking".

Figure 4.3 shows our study's frequency distribution of different leg activities. We see that the counts of "sitting" and "standing" were relatively lower than the others. As "sit-

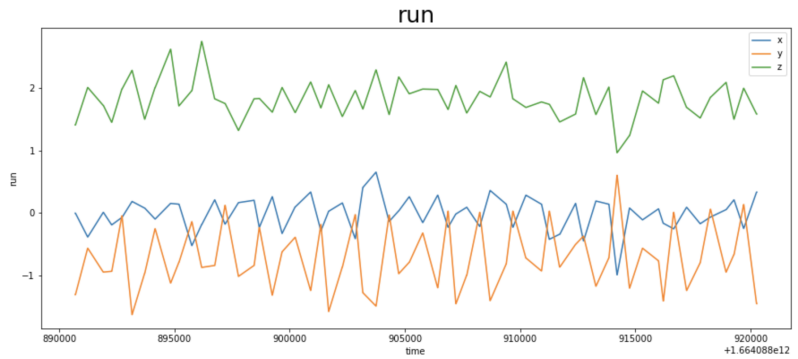
Figure 4.1: Insole for leg activity detection



(a) Sit



(b) Stand



(c) Run

(d) Walk

Figure 4.2: Data representation of different types of leg activity

ting" and "standing" are low-movement activities, there was less movement along the accelerometer's axis, resulting in a low count.

4.2.3 Data preprocessing

As activities are performed over a long period compared to the frequency (20Hz in our case), we segmented the data using the sliding window technique. We segmented every 2 seconds with a 50% overlap between adjacent windows. Each window had 40 rows of data with a sampling rate of 20Hz. We combined all the data in a single CSV file to start the preprocessing. Each row contained the pressure sensor values, accelerometer x,y,z values, timestamps, and labels. From each window(for every 2 seconds), we extracted the mean, standard deviation, and average absolute difference of the sensor data in both the time and frequency domains. We set the mode value of each window's activity label as a label for each row in the extracted feature set. We partitioned the feature data using a 70-30 train-test split method. So we trained the classifiers with 70% of the data and validated the performance using 30% of the data.

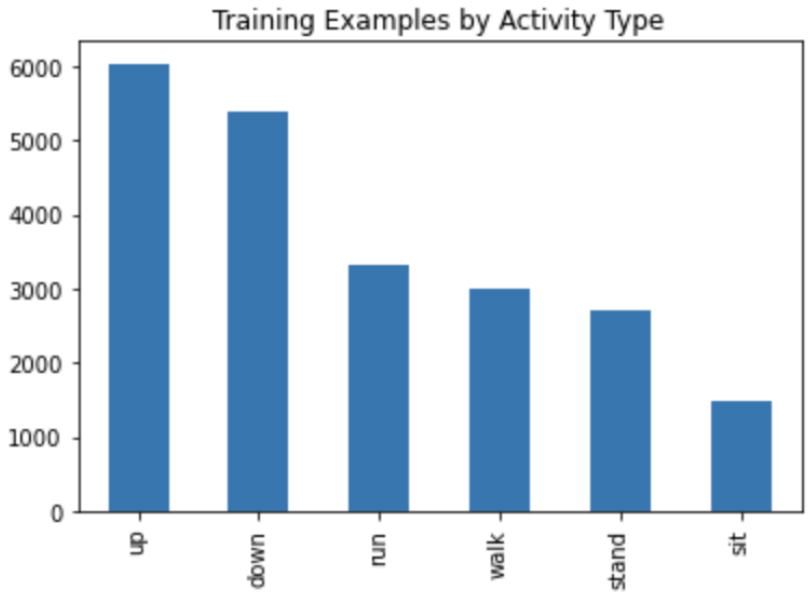


Figure 4.3: Leg Activity Frequency Distribution

Classifier	Sit	Stand	Walk	Run	Upstairs	Downstairs	Average
SVM	96%	92%	91%	96%	76%	78%	84.8%
Decision Tree	95%	100%	98%	98%	75%	69%	84.1%

Table 4.1. Activity-wise accuracies for leg activity detection

4.2.4 Oversampling - SMOTE

The leg activity data set has a higher count for stair activities but a lower count for sedentary activities such as sitting and standing. The imbalance in frequency distribution can bias the machine learning algorithms and might lead to minority class ignorance. Randomly resampling the dataset can balance the dataset. There are two primary approaches to random resampling: over-sample and under-sample. Oversampling can be implemented by creating duplicate data from the minority class. The undersampling approach deletes data from the majority classes. We used SMOTE (Synthetic minority oversampling techniques) to oversample the lower count activities and balance the dataset to tackle the imbalance in the count of some activities. SMOTE randomly selects a row of the minority class and computes its K nearest neighbors to generate synthetic samples of the minority class. We applied this method only to the training dataset to fit the models. The resampling was not applied to the test dataset we used to evaluate the performance of the classifiers.

4.3 Result

We trained two classifiers: Support Vector Machine(SVM) and Decision Tree(DT). With 10-fold cross-validation, we achieved an average accuracy of 86%(+/- 5%) using SVM and 84%(+/- 3%) using DT. The accuracies are acceptable but need to be higher. So we looked at the activity-wise accuracy for one fold from both classifiers. We show the comparison in Table 4.1.

As we see lower accuracies for upstairs and downstairs, we looked at the confusion matrix for SVM, which is shown in Figure 4.4. The confusion matrix shows that

Classifier	Sit	Stand	Walk	Run	Stairs	Average
SVM	92%	88%	97%	98%	**97%**	**96%**
Decision Tree	84%	92%	90%	96%	**98%**	**94.8%**

Table 4.2. Activity-wise accuracies for leg activity detection with stair data combined

the upstairs and downstairs activities are often confused. The upstairs data was sometimes confused with walking too. As people go upstairs or downstairs with different patterns(one step, two steps, or more at a time), it is difficult to differentiate these two activities. Also, our insole had an accelerometer only. A gyroscope and magnetometer may give us better insights into these two activities.

4.3.1 Tackling stair activities

Because of the confusion and low accuracy with upstairs and downstairs, we trained the models without these activities. We achieved an improved average accuracy of 95%(+/- 3%) using SVM and 96%(+/- 2%) using DT.

As we saw that our classifiers were confused about upstairs and downstairs data, we combined both upstairs and downstairs data and created a new class named 'stairs'. The objective of this approach is to detect the activities that involve stairs, irrespective of the direction. We present the result in Table 4.2. The overall accuracy has increased with both classifiers. We achieved an average accuracy of 96% (+/- 2%) using SVM and 94.8% (+/- 2%) using DT.

4.4 Comparison with Smartphone data

The insoles we used to collect leg activity data consisted of one accelerometer and four pressure sensors. Smartphone operating systems provide a sensor framework to access and acquire raw sensor data. These frameworks provide methods to check the availability of a sensor, register the sensor to return data, acquire the data and then unregister the sensor. To compare the result, we used our system application to collect data from the accelerometer, gyroscope, gravity sensor, and magnetometer for six activities.

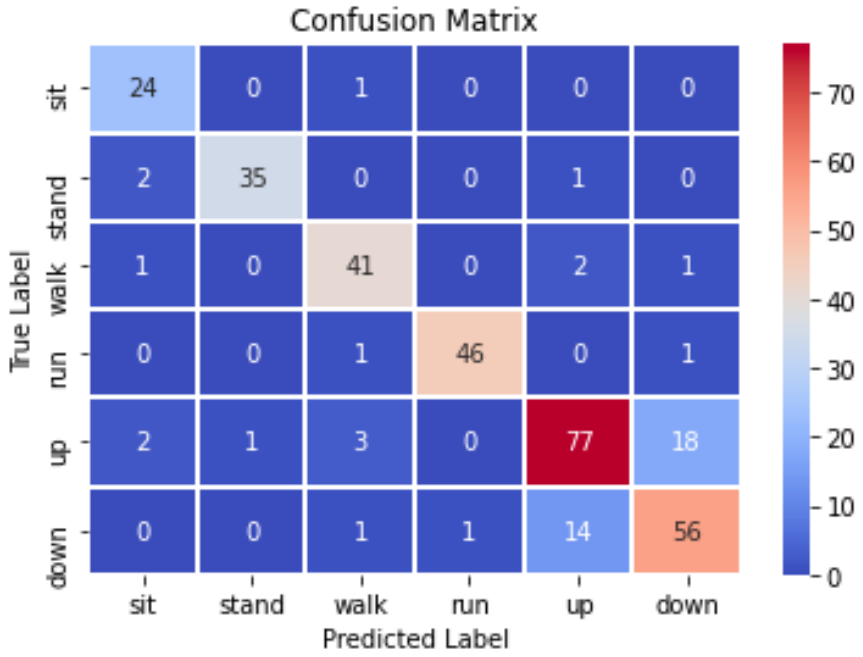


Figure 4.4: Confusion matrix for SVM in detecting leg activity

4.4.1 Smartphone Sensor

Modern smartphones have built-in sensors to measure motion, orientation, temperature, proximity, and environmental properties. There are three categories of smartphone sensors.

Motion Sensors

Motion sensors such as accelerometers, gravity sensors, gyroscopes, and rotational vector sensors measure the acceleration and rotation of the smartphone device along three axes. We depended on the data from the accelerometer and gyroscope to detect leg activities. However, we also collected data from gravity sensors.

Environmental Sensors

Different environmental parameters such as temperature, pressure, light, and humidity can be measured via a smartphone device using integrated thermometers, barometers, and photometers. Illumination (ambient light level) can be measured using a

hardware light sensor to control screen brightness automatically. Dew point and relative humidity can be measured using humidity sensors. For our research, we did not use any environmental sensors to detect physical activities. However, we will take advantage of built-in environmental sensors to detect the surrounding context in the future.

Position Sensors

Position sensors can measure the physical position and orientation of a device. An orientation sensor can measure the degrees of rotation of a device around the x,y, and z axes. Orientation sensors are not physical sensors. Instead, they are software-based sensors that calculate their output from one or more of the hardware-based sensors.

4.4.2 Data collection using a smartphone

We developed a separate data collection module in the application that was used to acquire sensor data from six activities: sitting, standing, walking, running, upstairs, and downstairs. These are the same activities that we analyzed using the smart insoles. We collected data from each activity for five sessions that were 10 seconds long. The smartphone was put in the right pocket while performing the activities. The app asked users to select the name of the activity first, then wait for 5 seconds before registering the sensors to start a 10-second timer. As there may have been some delay for the user to select the activity, place the phone in the right pocket, and then perform the activities, the 5-second delay was needed for accurate data collection.

4.4.3 Sample Rate

Android devices can output sensor data at four different predefined sampling rates. Instead of defining a sampling rate, the android sensor manager takes a *data delay* value while registering the sensors. The data delay specifies the interval at which sensor events are delivered to the application. The four predefined interval values are:

- Normal: 200000 microseconds

- Game: 20000 microseconds
- UI: 60000 microseconds
- Fastest : 0 microsecond

Android also provides the option to define an absolute value in milliseconds as data delay. The specified delay is a suggested value defined according to the system requirements. However, the system usually alters the value and uses a smaller delay. It is advised to specify the largest value possible that meets the system requirement, as larger data delay can improve smartphone system resource usage and results in consuming less power. For our case, we used the SENSOR_DELAY_UI (= 60000 microseconds) as data delay. The equivalent sampling rate was approximately 17Hz, but for some high-frequency events, we got data at a sampling rate of up to 50Hz.

4.4.4 Challenges in data collection using Android smartphone

The smart insole returned all sensor data at the same timestamp, so we did not have to worry about merging or combining the outputs. However, the android sensor manager returns the output from only one sensor simultaneously. Whenever a change is detected in the value of any sensor, the sensor manager returns the sensor output along the three axes, the sensor type, and the timestamp of the change. We saved output from different sensors in different CSV files and later merged them by timestamps. As at a specific timestamp, we had values from only one sensor. We set a tolerance of 500 milliseconds as a merge rule. Suppose the merge algorithm did not find a value from a sensor at a specific timestamp while merging two CSV files from different sensors. In that case, it took the last value from the last 500 milliseconds and used the value to replace the null value.

4.4.5 Data Description

For an activity performed over a 10-second duration, we were supposed to have approximately 160-170 data points per sensor with a sampling rate of 17Hz. For sitting

	Accelerometer	Magnetometer	Gyroscope	Gravity Sensors
Sit	**150**	149	124	123
Other Activities	**498**	149	125	123

Table 4.3. Sensor output for different activities

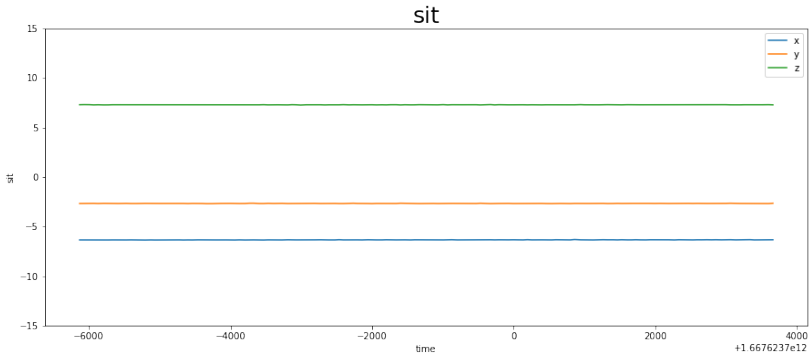
Sit	Stand	Run	Walk	Upstairs	Downstairs
148	488	490	488	489	491

Table 4.4. Average number of sample per activities in 10 second

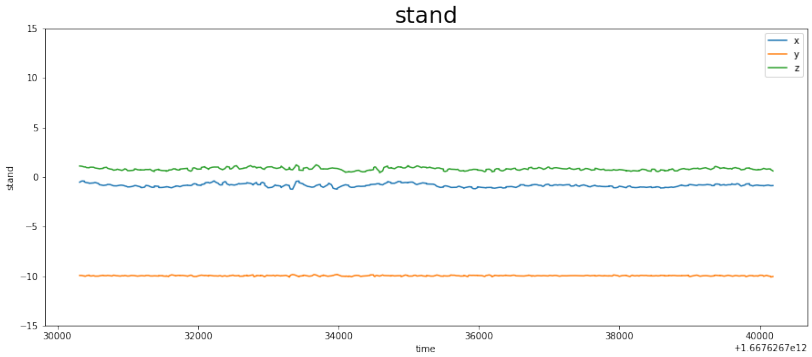
activities, we got less movement in the sensor values. As the android sensor manager returns sensor values only when some changes are detected, the sampling rate of the sitting activity was lower than the other activities. The application detected more change in the accelerometer than other sensors for other activities. Table 4.3 shows the output size from different sensors for activities performed in 10 seconds. Table 4.4 shows the average number of samples per activity after merging the outputs from different sensors with a 500-millisecond tolerance.

4.4.6 Data Visualization

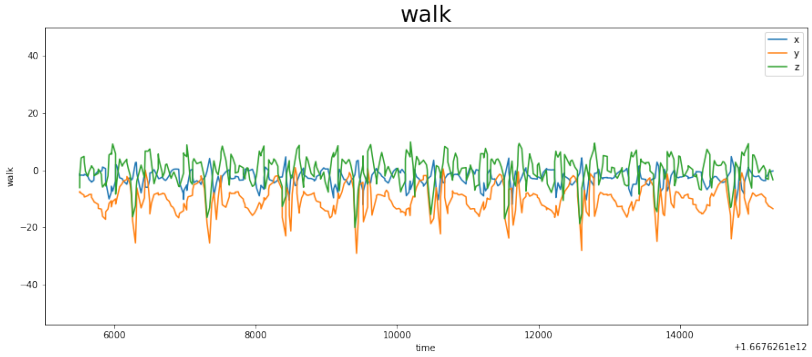
Figure 4.5 and 4.6 visualizes the accelerometer data from different activities collected using a smartphone. The data from sitting (Figure 4.5(a)) and standing (Figure 4.5(b)) show less movement along all axes and match the visualization of data from insoles (Figure 4.2(a) and 4.2(b)). The data from running (Fig 4.5(d)), walking (Fig 4.5(c)), upstairs(Fig 4.6(b)), and downstairs(Fig 4.6(d)) show a periodic pattern for each of the sensors and generate a large amount of information. The shape and pattern of the movements from these activities are significantly different than we saw from insoles. The values from smartphone sensors are relatively higher too. Running and walking motions are similar, but running is executed faster than walking, with a shorter time difference between periods.



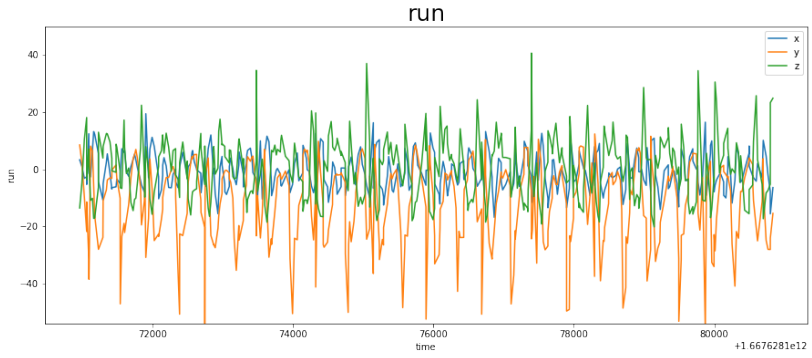
(a) Sit



(b) Stand

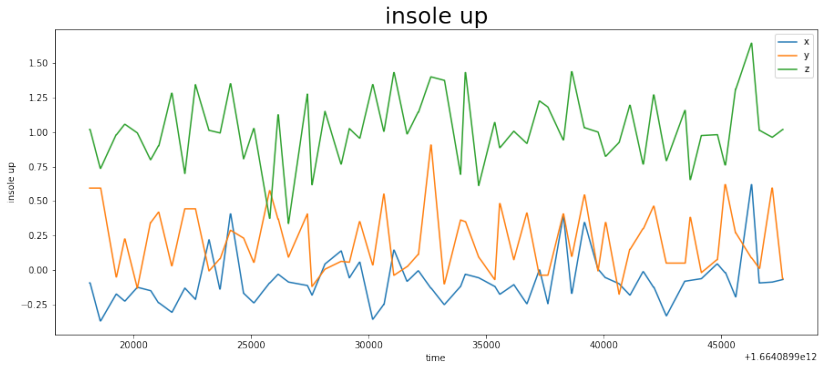


(c) Walk

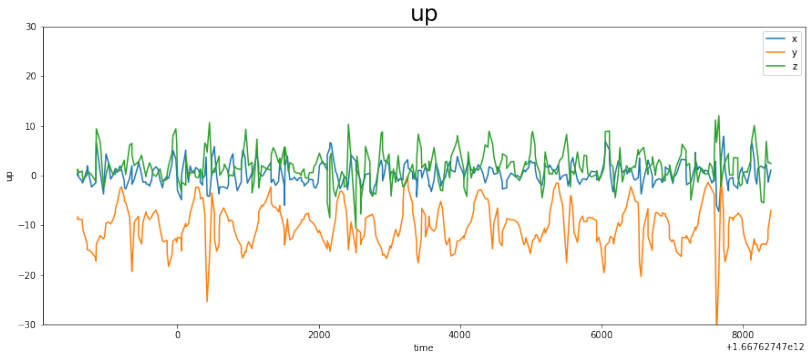


(d) Run

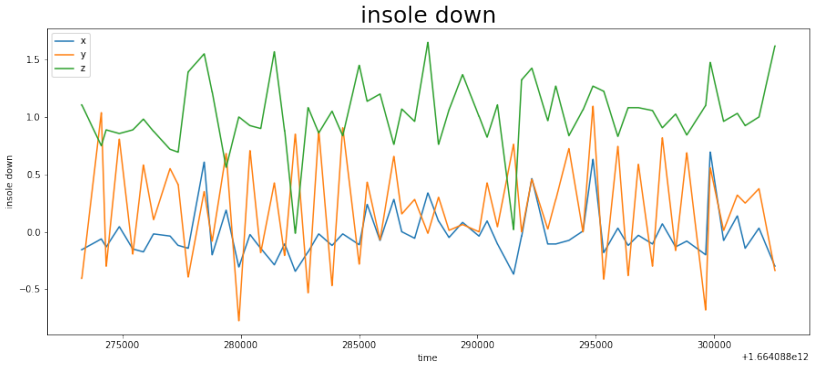
Figure 4.5: Data collection using smartphone: Sit, Stand, Walk and Run



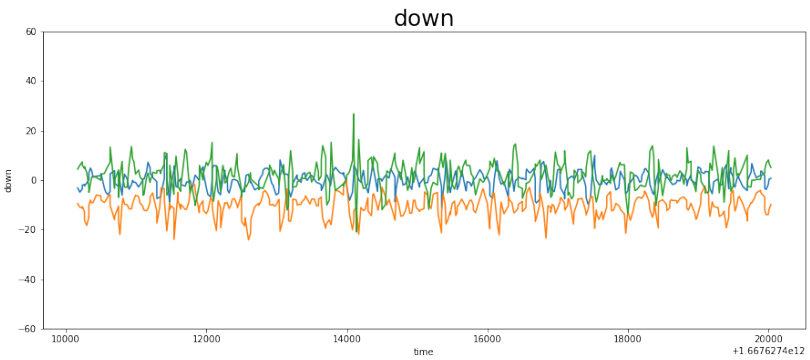
(a) Insole Upstair



(b) Smartphone Upstair



(c) Insole Downstair



(d) Smartphone Downstair

Figure 4.6: Comparison of Upstair and Downstair data from Insole and Smartphone

	Precision	Recall	F-Score
Sit	1.00	1.00	1.00
Stand	1.00	1.00	1.00
Walk	0.94	1.00	0.97
Run	1.00	1.00	1.00
Up	1.00	0.91	0.95
Down	0.89	0.94	0.92

Table 4.5. Leg activity detection with smartphone sensor (Accuracy 96.9%, RMSE 0.11)

4.4.7 Result

The objective of using smartphone sensors to detect leg activities was to compare the result with insoles and see if we improved the detection of upstairs and downstairs data by taking advantage of gyroscopes and magnetometers. In this section, we present our comparative result.

We achieved the best result with the support vector machine algorithm on features from the accelerometer and gyroscope (both time and frequency domains). The 10-fold cross-validation score is 98% (+/- 2%). The activity-wise performance is shown in table 4.5, and the confusion matrix is shown in figure 4.7. If we use the accelerometer feature only, the overall accuracy goes down to 92.3% with a higher RMSE value (0.26) than using the accelerometer and gyroscope together (0.11). Also, the accuracy of predicting upstairs (83%) and downstairs (83%) activity goes down too. Thus, we need gyroscope data combined with accelerometer data for better leg activity prediction. Frequency domain features are also important, as well as time domain features. With only time domain features of the accelerometer and gyroscope values, we got an overall accuracy of 87% with comparatively lower accuracy for the run (84%), upstairs (82%), and downstairs (64%) activity. We collected data from gravity sensors and magnetometers too. However, we did not get any significant improvement in adding these features to the training dataset.

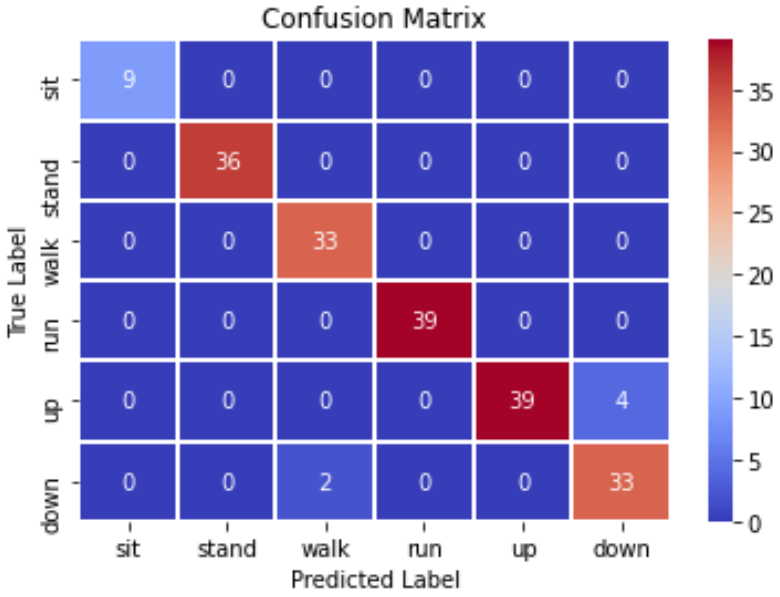


Figure 4.7: Confusion matrix for SVM in detecting leg activity using smartphone sensors

4.4.8 Smartphone Vs Insole

After performing leg activity detection using the smartphone, we have some key points to mention:

- We got a better result from smartphone sensors than using an insole in detecting leg activities. Moreover, smartphone sensors can successfully distinguish upstairs and downstairs activities. Using a smartphone to detect leg activities makes better sense than using an insole.

- The main difference between the smartphone and smart insoles is the gyroscope. We used the gyroscope values as features to train the classifiers using the smartphone. The insoles do not have any gyroscope integrated. We also experimented by building a classifier without taking the features from gyroscopes but ended up with lower performance.

- The position of the sensors on the body can make a difference too. The movement and change in smartphone orientation during different activities provide informa-

tion about the performed activities. We put the smartphone in the pocket during data collection and the smart insoles inside the shoes. Smartphones are one of the most used objects by humans and can be carried anywhere. However, in a home setup, an individual usually only sometimes carries smartphones in their pocket. On the other hand, insoles are a great option as they increase users' comfort and remove the condition of carrying a sensor-integrated device all the time.

- The visualizations from Figure 4.2 and Figure 4.5 suggest that accelerometer data from the insoles may not provide accurate information. The main reason behind inaccurate information is the insoles were manufactured in 2015 and did not contain updated and improvised accelerometers. On the other hand, smartphone sensors are continuously improved. Using an updated insole as a data collection device may provide better accuracy.

Chapter 5
Detection of Hand Activities using MetamotionC

5.1 MetamotionC

To collect accurate information from performed hand activity, we used MetamotionC [105], a wearable device that can be attached to the wrist (Figure 5.1). The device comes in a small rectangular board and includes a sensor fusion of a 3-axis accelerometer, a 3-axis gyroscope, and a 3-axis magnetometer. With the combination of the three types of sensors, the device uses a sensor fusion algorithm to calculate a robust absolute orientation vector in the form of Quaternion or Euler angles. MetamotionC can record and stream raw sensor data with a sampling rate of up to 100Hz. The sensor fusion algorithm fuses the raw data from different sensors to improve the sensor output.

Figure 5.1: Metamotion for hand activity detection

5.2 Quaternions

A quaternion can be expressed with a scalar component and a vector component with the following equations:

$$q = (r, \vec{v}) = (q_r, q_x\hat{i}, q_y\hat{j}, q_z\hat{k}) \tag{5.1}$$

Here q_r is the real components, and the remaining terms are imaginary components. q_x, q_y, q_z are real numbers along three axes and $\hat{i}, \hat{j}, \hat{k}$ are mutually orthogonal imaginary unit vectors.

Euler angles are the most common method to represent the orientation of an object with the help of three successive rotations about the z-axis, then the new x-axis, and finally again about the new z-axis. Euler angles calculation can be affected by *Gymbal Lock* that prevents a rotation from properly executing. Gimbal lock cannot affect quaternion representation, and successive rotation can be applied easily with quaternions. Quaternions can efficiently represent the rotational and orientational information of a rotating object. Quaternion value of a rotation of θ about a unit axes $\vec{n}$ can be represented as following:

$$q = cos\frac{\theta}{2} + \vec{n}sin\frac{\theta}{2}\vec{n} \tag{5.2}$$

To get a new orientation q_2 by rotating the current orientation q_0 by a specified amount q_1, we can multiply q_0 with q_1. The new quaternion q_2 represents a rotating object with a rotation described by q_1. A series of rotations can be expressed efficiently with quaternion multiplication. MetamotionC fuses the accelerometer, gyroscope, and magnetometer data and applies the sensor fusion algorithm to calculate rotation vector and quaternion values. We have used quaternion values (q_r, q_x, q_y, q_z) as features for our hand activity recognition.

5.3 Methodology

Hand activities such as chopping, brushing hair, typing, and eating with a fork differ from leg activities. Smartphones or sensors kept in the pocket or attached to the ankle or hip cannot collect hand gesture data. Devices attached to the wrist provide accurate information about hand movement. Smartwatches like FitBit and Apple Watches are now equipped with accelerometers, gyroscopes, and magnetometers and can be used to detect hand activities. Smart wrist-worn devices were used in [106, 107, 108, 109] for activity recognition and analyzing important human characteristics from the activities like falls and stress detection. Like leg activity detection, hand activity detections are also mostly sensor-based. So the fundamental methodologies of both hand and leg activities are almost the same in the literature. Machine learning techniques can explore sensor data well and extract essential features from the data.

5.3.1 Data Collection

We detected five hand activities: Resting, Chopping, Exercise, Scratching Hair, and Typing. We developed an android application that connects to the metamotion device via BlueTooth. The manufacturer of the device provides an SDK to collect the sensor data. We integrated the SDK into our application and set the sampling frequency at 20Hz for all the sensors. The application asked the user to perform the predefined target activities for 20 seconds. Figure 5.2 shows the frequency distribution of our collected data.

5.3.2 Data Representation

Figure 5.3 shows the quaternion values along the x,y, and z axis for different activities. There is no movement along all three axes when the hand is at rest (Figure 5.3(a)). There is little movement when typing on a keyboard (Figure 5.3(b). The movement is very little compared to chopping and exercise because when a person types, the wrist does not move much, only the fingers move. The movements along all the axes increase as

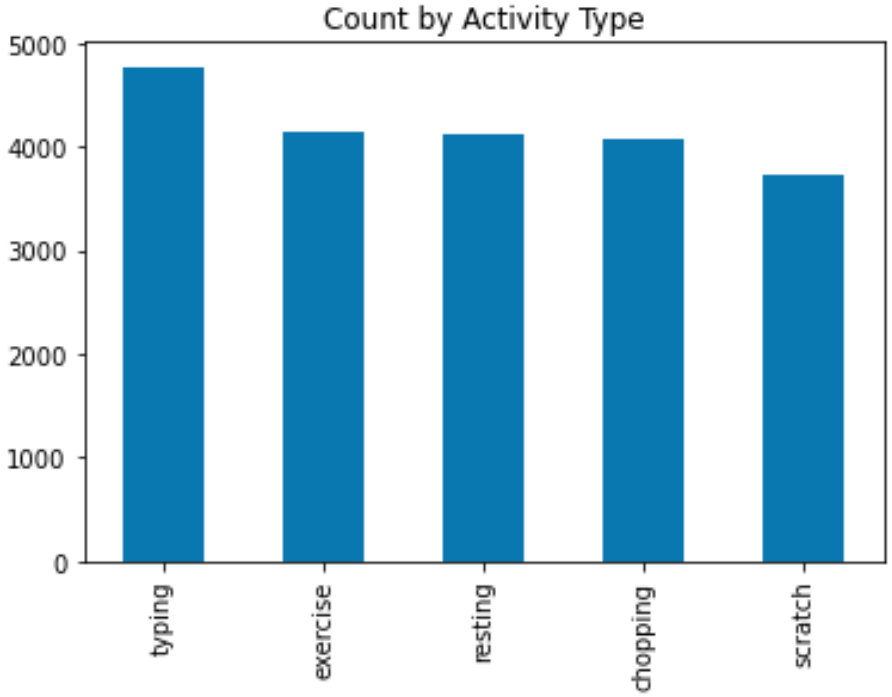
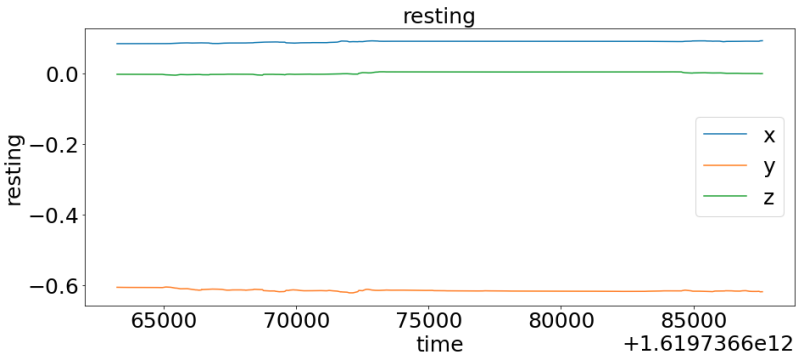


Figure 5.2: Hand Activity Frequency Distribution

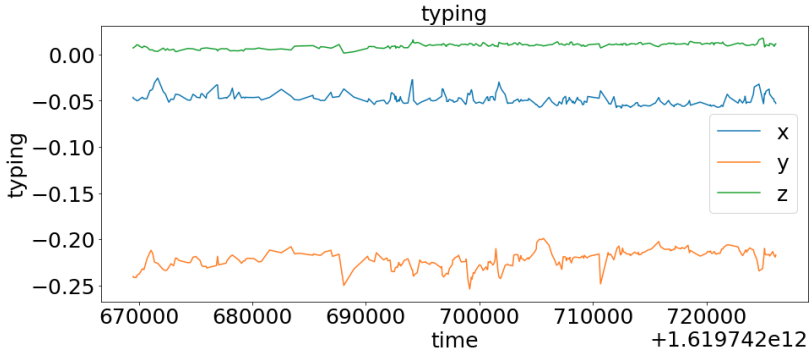
expected in the event of chopping and exercise (Figure 5.3(c) and 5.3(d)), as there is a lot of hand movement involved in these two activities.

5.3.3 Data Preprocessing

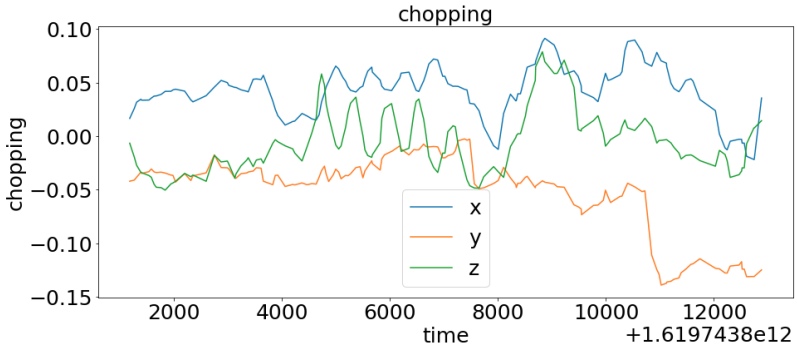
We segmented the sensor data into a 2-second window with a window size of 40 and 50% overlap between adjacent windows. We used the accelerometer x,y,z values, gyroscopes x,y,z values, and the quaternion w,x,y,z values to generate the features. For each window, we calculated the mean and standard deviation of the sensor data in both the time and frequency domains. The mode value of the activity labels of each window was set as the label for the training features. We experimented with other features such as median, max values, min values, and the time between peaks, but it did not change the accuracy of the classifiers.



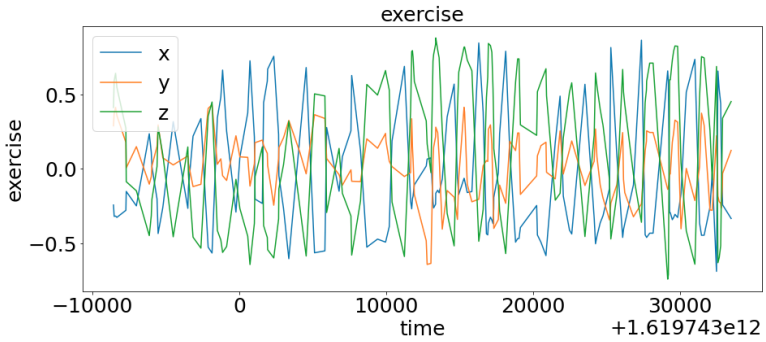
(a) Resting



(b) Typing on keyboard



(c) Chopping with a knife



(d) Exercise

Figure 5.3: Data representation of different types of hand activity

Classifier	Exercise	Resting	Chopping	Typing	Scratching	Average
SVM	100%	91%	96%	87%	80%	91%
Decision Tree	100%	91%	93%	90%	100%	94%

Table 5.1. Activity-wise accuracies for hand activity detection

5.4 Result

After doing a 70-30 split on the feature data, we trained an SVM and a DT classifier with the training data. We achieved an average accuracy of 93% (+/- 2%) for the SVM classifier and 95% (+/- 1%) for the DT classifier using all the features with 10-fold cross-validation. Table 5.1 shows the comparative metrics of both the classifier for different activities. We can see that both the classifiers perform very well in classifying exercise data. DT has good accuracy with "typing" but gets relatively low with chopping and scratching. On the other hand, SVM performs well in detecting "chopping" compared to DT but performs poorly in detecting "scratching".

Chapter 6
Voice Activity Detection and Exposure to High Noise Level

Besides physical activity, social interaction also plays a vital role in a person's fatigue level. Though it is evident from the literature that social behavior positively affects current mood and lower fatigue, there are reports of increased fatigue 3 hours after high social engagement [55]. Exposure to a high level of surrounding noise can lead to severe fatigue. Older adults exposed to a sound level of 85 dB or more are fatigued more often and physically and mentally less active. [110]. We want to detect a person's social engagement and exposure to high noise levels, which may lead to higher fatigue. We utilized a smartphone application to connect the insoles and metamotion. We used the same application and developed a separate module using the smartphone's microphone sensor to detect voice activity and surrounding noise levels.

6.1 Features of sound

This section describes various properties that we can extract from the sound wave. Sounds can be represented in a wave with time to the x-axis and the intensity of the sound to the y-axis. The intensity represents the pressure sound creates while traveling through the air. A louder sound generates louder pressure. Sound waves are split into smaller frames or windows for analysis purposes, and the features are extracted from the individual frames. The length of the frames may vary depending on analysis techniques. Figure 6.1 shows the wave formed by a woman speaking, "Kids are talking by the door".

6.1.1 Short-Time Energy

The energy of a signal can be expressed as the total magnitude of the signal. In the context of the sound wave, it represents the loudness of a signal. The energy associated with speech usually varies with respect to time. So short time analysis is done to estimate the energy in different parts of the speech signal. There are different regions in a

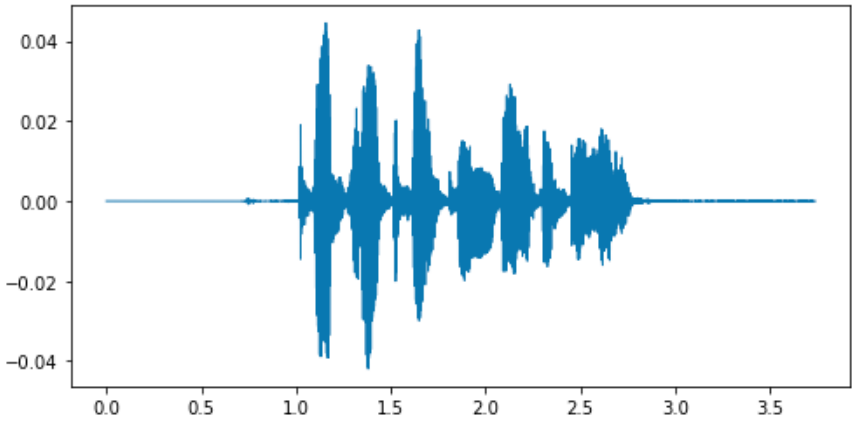
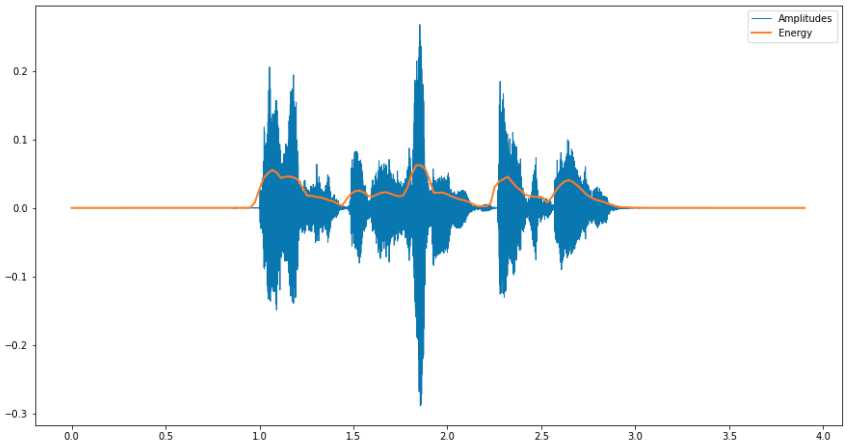


Figure 6.1: Speech wave

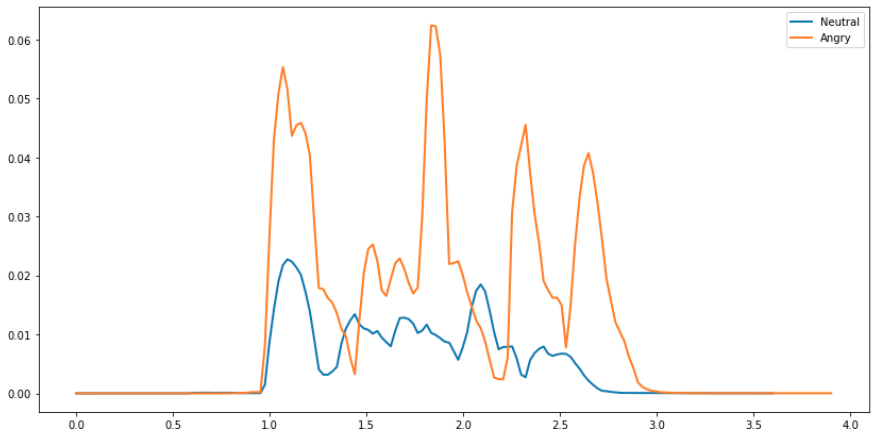
speech signal, such as voiced, unvoiced, silent, and noise. The regions containing voice have higher short-time energy, and unvoiced regions have lower short-time energy. We can visualize speech wave energy from Figure 6.2. In Figure 6.2(a), we can see the corresponding varying short-time energy in the orange line. In figure 6.2(b), we plotted the energy of a person speaking the same sentence with two different emotions: neutral and anger. The energy from angry speech is higher than that from neutral speech. Also, From figure 6.2(b), the lower energy areas are unvoiced, silent, or noisy regions. In figure 6.2(c), we show the comparison of energy between sound sources from human speech and a truck engine. It also visualizes the fact that louder sounds have higher short-time energy.

6.1.2 Zero-Crossing Rate

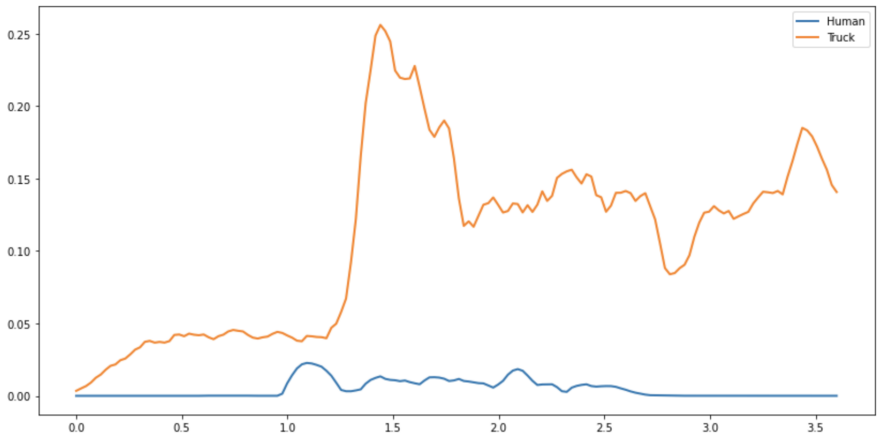
A zero crossing occurs when a sound waveform passes through the axis and changes sign. A zero crossing can be measured by comparing the signs of two successive samples. Zero crossing is said to occur if two successive samples have different algebraic signs. A zero-crossing rate can be measured by the rate at which the amplitude of a signal passes through zero. Figure 6.3 shows the zero-crossing rates of the sound wave from human speech and a truck engine. Human speech has a relatively high zero-crossing



(a) Energy of a speech



(b) Comparison of speech wave energy for two different emotion: Neutral and Angry



(c) Comparison of speech wave energy for two different source: Human Speech and Truck Engine

Figure 6.2: Speech Energy Visualization

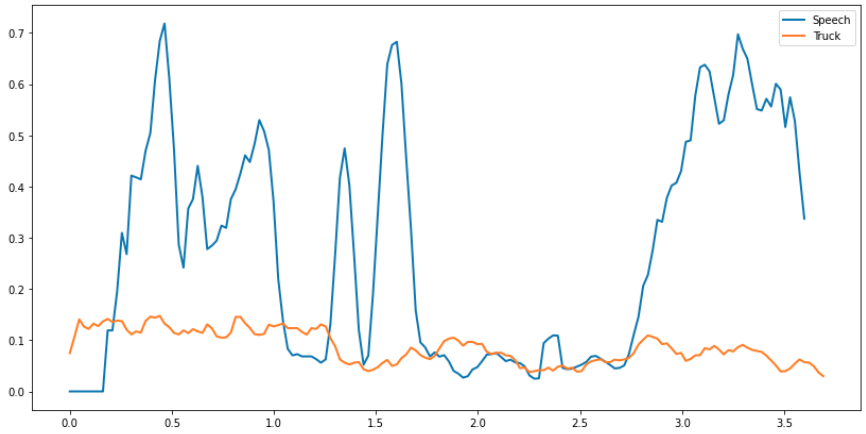


Figure 6.3: Speech wave

value because of the oscillating nature around zero. The unvoiced regions of a speech wave have a higher zero-crossing rate than the voiced region.

6.1.3 Short-Time Fourier Transforms

Fourier Transform on a signal can convert the signal from the time domain to the frequency domain and helps to visualize how time interacts with frequency. We can look at signal features at some particular time if we 'window' the signal into shorter frames. The segmented windows can be transformed into Fourier series to obtain *Short Time Fourier Transform (STFT)*, to visualize different windows or frames of signals. The STFT provides a visual representation named Spectrogram to provide essential insights into the signals. Figure 6.4 shows the Spectrogram of a speech signal. Spectrograms are a way to provide a visual representation of a signal's loudness as it varies over time and frequencies. As humans can hear sound with a small range of frequencies, the y-axis in a spectrogram is converted to a log scale, and the color dimension is converted to decibels.

6.1.4 Mel Frequency Cepstral Coefficients

The Mel Scale is a way to measure the perceived frequency of a sound and relate the value to the actual frequency. Human ears can differentiate small changes in lower

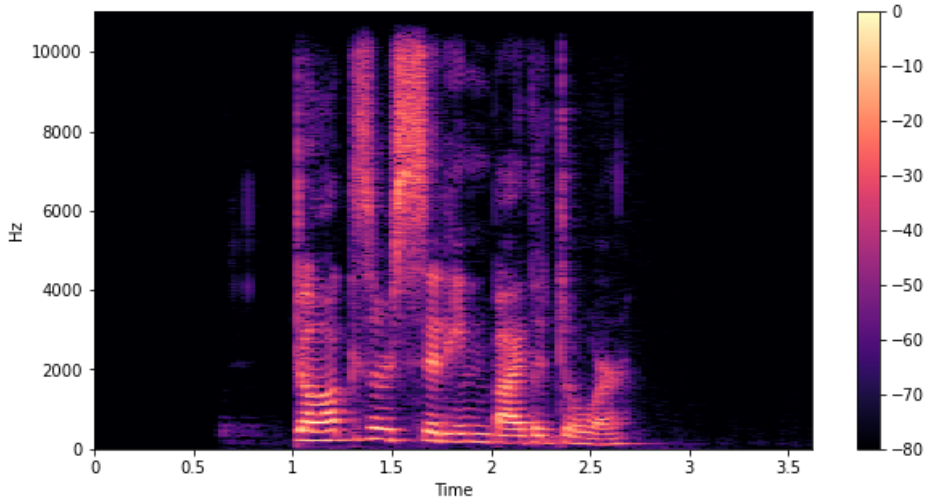


Figure 6.4: STFT Spectrogram

frequencies but fail to do that in higher frequencies. The Mel Scale converts the sound feature to match the frequency range of the human ear. The following formula can be used to convert an actual frequency to the Mel Scale:

$$M(f) = 1125 \ln(1 + \frac{f}{700}) \tag{6.1}$$

Human-generated sound can be determined by the shape of their vocal tracts, such as their tongue and teeth. The envelope of the time series power spectrum of speech wave represents the vocal tract. The envelope can be calculated using the Mel-Frequency Cepstrum Coefficients (MFCC). MFCCs are one of the most popular ways of representing and analyzing sounds. We can extract 26 MFCCs from the audio signal by following the steps shown in Figure 6.5. First, the audio signals are split into frames with specified frame lengths. A short fast fourier transform (STFT) can extract the power spectrum envelopes on the frames. The mel filter banks are triangular waveforms that can be applied to the STFT signals to extract the mel scale-based values using logarithm. A discrete cosine transformation on mel values can output the MFCC values for further analysis.

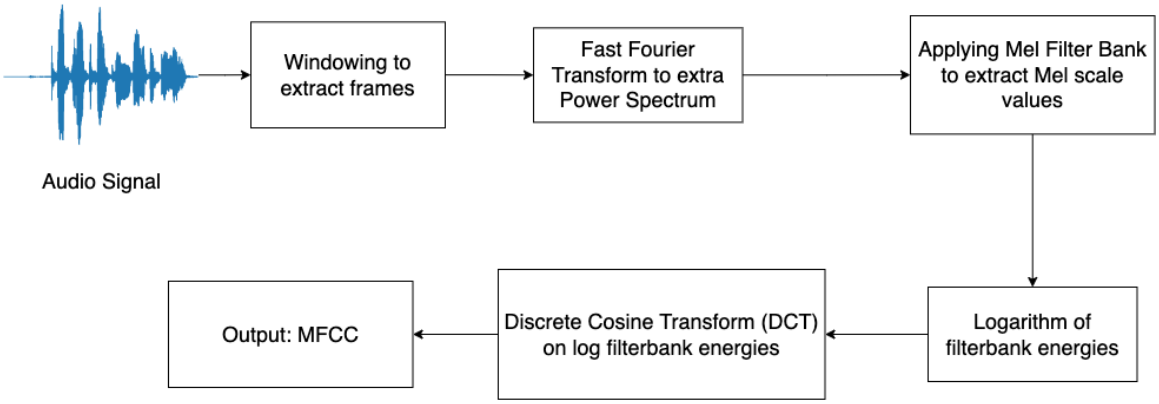


Figure 6.5: Flow chart for obtaining MFCCs

6.2 The Mel Spectrogram

A Mel Spectrogram shows the sound wave frequencies in mel scale values. The mel scale values are beneficial as it is specially developed keeping human frequency range in mind. The mel spectrogram maps the frequency values in hertz to the mel scale, which are essential in applications that work on human hearing perception. Figure 6.6 shows the mel spectrogram of a human speech.

6.3 Voice Activity Detection

Detecting voice or being in the presence of a conversation will be helpful for us to find a user's social and behavioral patterns. Detecting voice in a sound buffer will lead us to analyze the voiced part further to detect stress, emotion, and social exhaustion from it and find a pattern of social fatigue. This section describes our approach to detecting voice in a sound buffer.

6.3.1 Google VAD

As a first approach, we used Google's Voice Activity Detection tool. We integrated a wrapper of Google VAD, built to be used in the Android platform [111]. This algorithm applies Gaussian Mixture Model to the microphone's voice buffer and detects if a voice is heard in real-time. Any sound other than a voice is classified as noise. However, the limitation of this approach is that we found many false positives, where a high-intensity

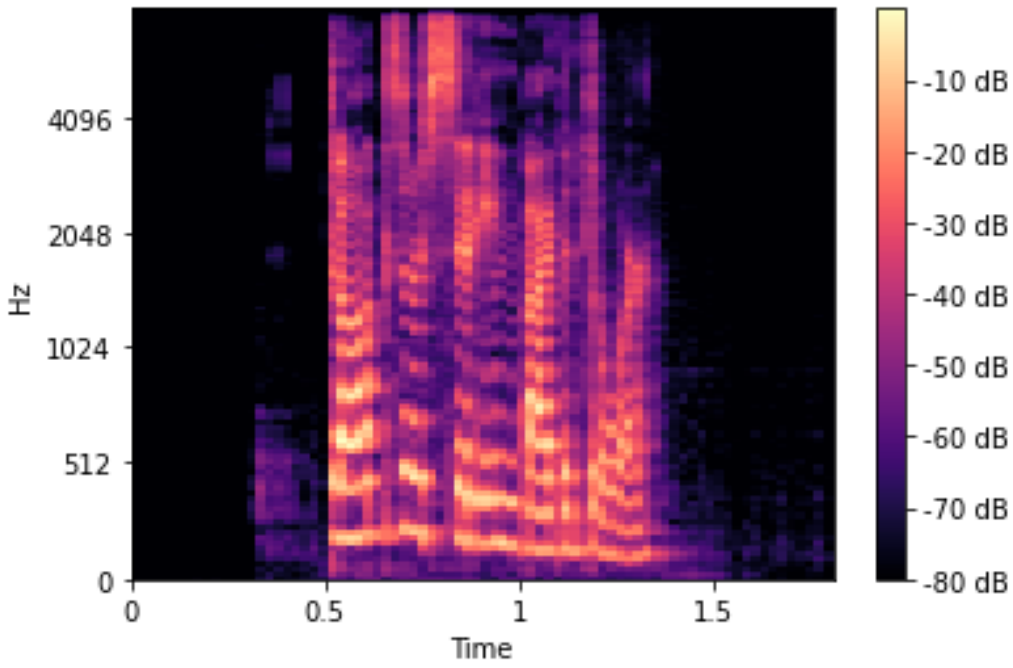
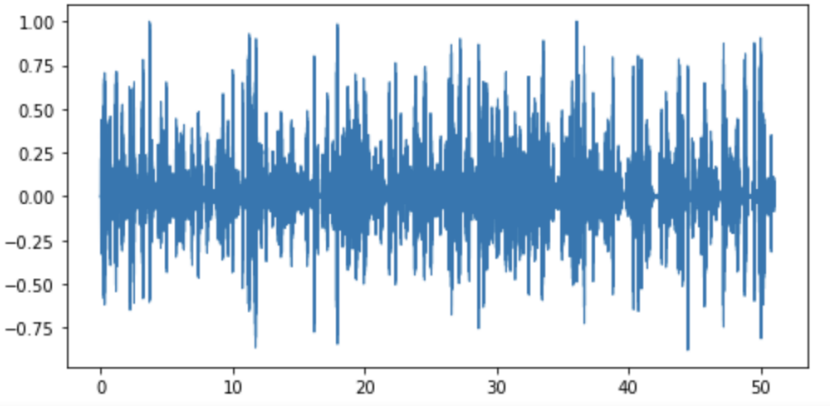


Figure 6.6: Mel Spectrogram

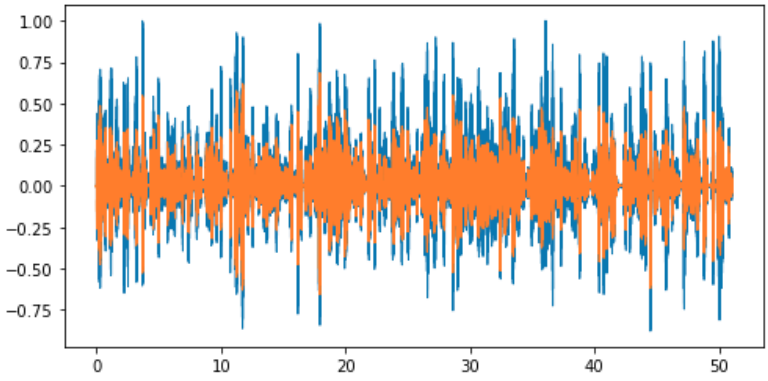
sound but not a voice was classified as a voice.

6.3.2 Experimental Analysis

We wanted to analyze raw sounds to extract features from the sound buffer. We extracted the sound buffer from the recording with the microphone. We used the AudioRecorder class of the Android platform. We initialized an instance of the class with a sampling rate of 48kHz and started the recorder in different environments. The environments included silence, a person speaking, and loud engine noise. Android recorder returns the real-time sound buffer amplitude values in a stream of arrays. We processed the buffer and normalized the values to be in the range of -1.0 to 1.0. We know that human voice frequency can range from 80 Hz to 3000 Hz [112]. So we applied a digital filter to filter out all the other sounds that were unnecessary in our case, as shown in Figure 6.7. Figure 6.7(a) shows the buffer of a person speaking with some background noise. After filtering out the unnecessary noise, we found the denoised sound, shown in orange in Figure 6.7(b).



(a) Normalized sound buffer



(b) Filtered sound buffer

Figure 6.7: Applying a digital filter on normalized sound buffer

As there may be silence between spoken sentences, or background noise that falls in the range of human voice frequency, we segmented the filtered buffer with a window of 2 seconds. For every 2 seconds of data, we analyzed various features. We found exciting values when we calculated the range (difference between maximum and minimum) of each 2-second window. As human voices have a wide range, we found the range of speech buffers to be relatively high (0.7-1.3) compared to the range of silence (0.001-0.1) and engine noise (0.4-0.7).

Calculating the zero-crossing rate and energy of the sound waves can detect the voiced part from a buffer. Zero-crossing value in a wave is defined by the count of a wave that crosses the horizontal axis. Short-time energy is calculated by finding the energy variation in a small speech frame. A voiced part has a low zero-crossing rate and high short-time energy in a sound buffer with both voiced and unvoiced parts. On the other hand, an unvoiced part has a high zero-crossing rate and low short-time energy [113].

6.3.3 Audio Dataset Description

Our purpose is to detect the presence of voice, silence, and any other noise in an audio stream. We collected 30 audio files containing 2 seconds of speech for training from the Noizeus [114] dataset. The clean speech in each file was corrupted using different real-world noises. To train the classifier with additional environment noise, we used the data from [115], which was created for environmental sound classification. The dataset had 50 labeled classes, but for our purpose, we marked them as 'noise' that does not include any human speech. We recorded 30-minute audio of a silent room and split the files into 2-second streams to make the silence data. We denoised the audio dataset for each class using a noise reducer library available in python [116]. Then we analyzed the clean sound buffer to extract mel-frequency cepstral coefficients (MFCCs), mel spectrograms, energy, and zero crossing rates as features.

Figure 6.8 shows the visualization of different audio streams we analyzed. The blue wave represents a high-amplitude noise from a washing machine. The orange waves

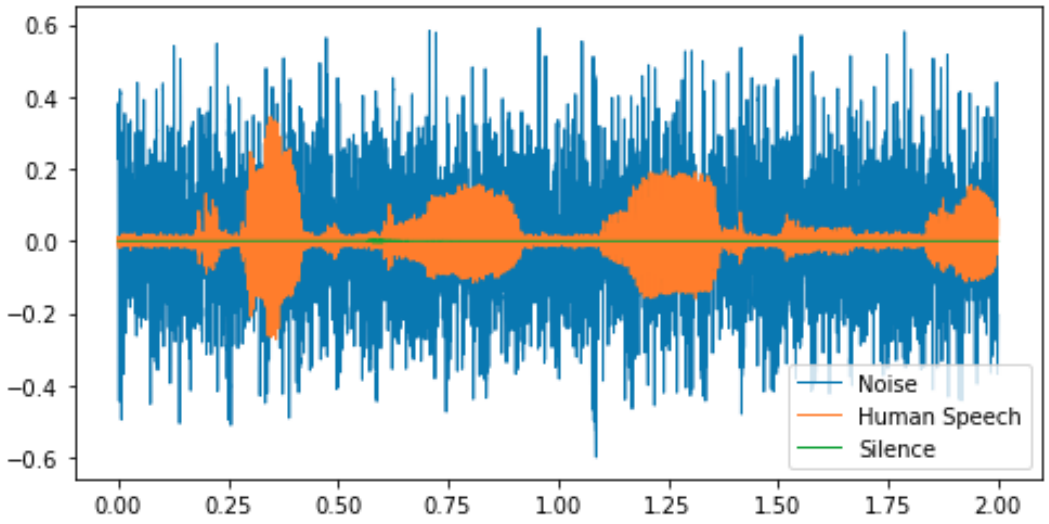


Figure 6.8: Sound wave visualization of noise, human speech, silence

are from human speech, where the low frequency is silence between spoken words. The green straight line represents quiet environments.

6.3.4 Voice and Noise Detection

From the combined audio dataset of speech, environment noise, and silence, we had 19 classes, including sounds from an airplane, rain, brushing teeth, voice, silence, and engine. As we wanted to detect the presence of voice and noise in an audio stream, we kept the voice and silence label unchanged and renamed all other classes “noise”. As we had only 30 human speech samples, we randomly picked the same number of noisy and silent data from the other two types and trained a Support Vector Machine classifier with 70% of the data. We got an average accuracy of 98% (+/- 2%) to distinguish the sound of the voice, silence, and any other noise. Though we trained with a small dataset for speech, the result is promising, and we hope to add more environment noise and speech data to our dataset.

6.4 Measuring Surrounding Noise Level

Sounds can be expressed as the pressure change in the air when it travels. The sound level is measured in decibels(dB) and can be obtained from the following formula:

$$E = 20log_{10}\frac{P}{P_0}$$

Here P is the perceived pressure, and P_0 is a pressure reference value. The measurement of sound level depends on this reference point. 0 dB is the softest sound a person can hear. Normal human voice levels can range from 50-60 dB [112]. In Table 6.1, we list the standard decibel values for different kinds of environments [117, 118]. We used the formula to calculate the decibel value of surrounding noise. As amplitudes are directly related to pressure, we calculated the perceived pressure for every stream of the amplitude array. Using our algorithm, we experimented with a different reference point and concluded that a reference pressure $20\mu Pa$ could approximate the best possible sound level. For comparison and experimenting, we downloaded a SoundMeter application [119] from Google Playstore, which measures the surrounding noise level. We ran our algorithm and the application simultaneously and found near-identical values with the set reference pressure point. The user is notified when the application detects a long-time high noise exposure. We are also saving the duration of different exposure levels to show in the comprehensive report.

Environment	**Sound Level(dB)**
Softest sound that can be heard	0
Average Home, Refrigerator hum	40
Normal conversation, air conditioner	40
Noisy office, City traffic (inside the car)	80-85
Approaching subway train, sporting events (such as hockey playoffs and football games)	100
Loud rock concert, Standing beside or near sirens	120

Table 6.1. Standard dB values for different environments

Chapter 7
Emotion Analysis and Environment Classification

7.1 Emotion Analysis from Speech

Human speech audio can express emotional information through its features. It is possible to detect the intensity of a conversation and the speaker's emotional state by analyzing the speech features. This section describes our approach to analyzing the RAVDESS dataset to build a Convolutional Neural Network and detect emotion from human speech.

7.1.1 RAVDESS Dataset

The Ryerson Audio-Visual Database Of Emotional Speech and Song (RAVDESS) [120] contains speech and song recordings by 24 professional actors(12 male and 12 female) expressing various emotions. The original database has 7356 files, from which we extracted the speech-only files. The extracted part has 1440 files. For each actor, 60 speech recordings expressed eight emotions: neutral, calm, happy, sad, angry, fearful, surprised, and disgusted. Figure 7.1 shows the frequency distribution per emotion in the dataset. The audio files were saved in WAV format with a 16-bit bitrate and a sampling rate of 48 kHz.

7.1.2 Methodology

In the RAVDESS dataset, the audio files are in WAV format. We used the librosa [121] package available in python to read the audio files, generate a waveform, visualize and extract necessary features such as Fast Fourier Transform, Mel Frequency Cepstral Coefficients, and Mel Spectrogram. We trained a Convolutional Neural Network on the clean dataset to detect the eight emotions from speech. Figure 7.2 shows different steps in detecting emotions from the RAVDESS dataset. The mel spectrogram and MFCCs features

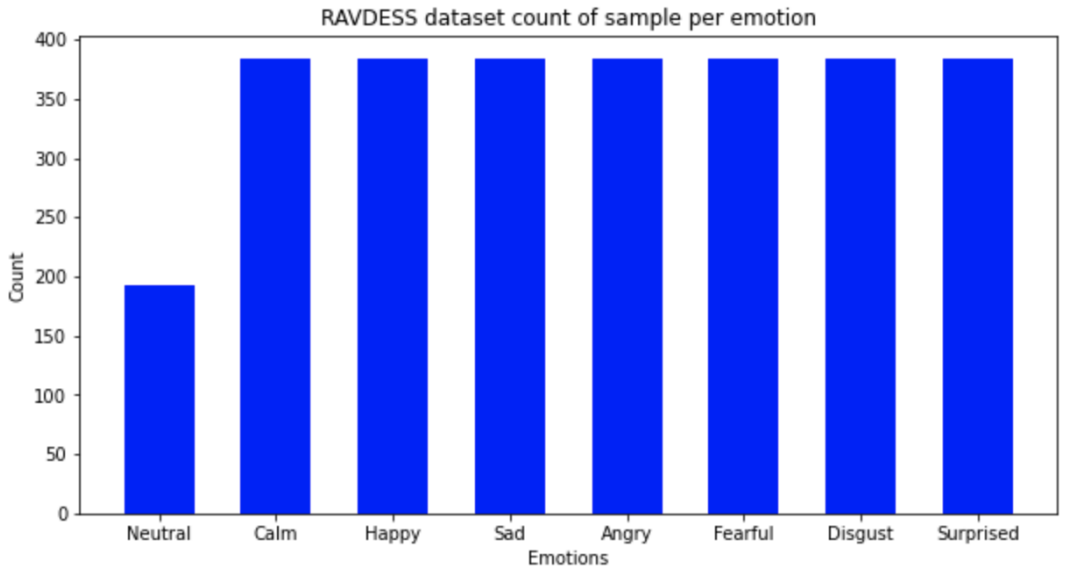


Figure 7.1: Frequency distribution of RAVDESS dataset

can be expressed as a two-dimensional image. So popular image processing techniques using deep learning can be applied to these features.

7.1.3 CNN architecture

The convolutional neural network consists of several building layers such as the convolutional layers, pooling layers, and fully connected layers [122]. A typical architecture consists of several repetitions of convolution layers and pooling layers on a stack, followed by some other fully connected layers. The convolutional layer is the principal component in the architecture that does all necessary feature extraction. The outputs from this layer are passed through some activation functions to decide if these layers' inputs are essential in the prediction process. A pooling layer provides a downsampling operation for dimensionality reduction. The CNN architecture we used on the RAVDESS dataset is shown in figure 7.3. We used ReLu as an activation function and added Batch Normalization to normalize the data in the batches to speed up the training. We added one Batch Normalization at the initial convolutional layer and another in the middle. In the end, there is a dense layer as the output layer with Softmax as an activation function

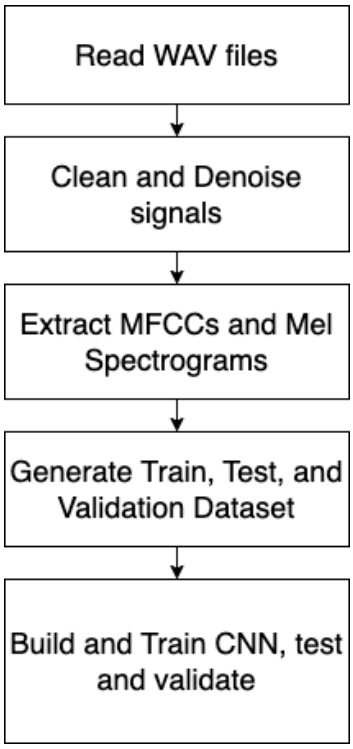


Figure 7.2: Approach to detect emotion from RAVDESS dataset

to provide a probability for each class. A Flatten layer is used as a connection between the convolution layers and the dense layer. The building block is done sequentially as a stack using the *Keras* library available in Python.

7.1.4 Loss Function

The performance of a CNN can be measured using prediction accuracy and loss value. Loss can be expressed as the penalty for a bad prediction. The aggregate value of loss indicates how bad the model is at predicting target values. A model predicting perfect values has a loss value of zero. Otherwise, the loss value is greater than zero. A simple and standard loss function is squared loss that can be calculated by taking the square of the difference between actual value and prediction. The average loss for the whole dataset can be expressed as Mean Square Error (MSE) and calculated from the following equation:

$$MSE = \frac{1}{N}\sum(y - prediction(x))^2 \tag{7.1}$$

Here N is the number of samples in the dataset, x is the feature set, y is the actual label, and $prediction(y)$ is the predicted value.

MSE is not always the best or most practical loss function for all classification problems. As our problem is a multi-class problem, we used a more complex loss function named Categorical Cross-Entropy Loss Function. This function compares each predicted class probability to the actual value and assigns a logarithmic penalty based on the difference between the actual and predicted value. The logarithmic nature assures a more significant penalty for larger differences. The cross-entropy function can be defined as

$$L_{CE} = -\sum_{i=1}^{n} t_i log(p_i) \tag{7.2}$$

where t_i is the truth label and p_i is the probability for the i^{th} class.

7.1.5 Result

We separately extracted the mel scale-based spectrogram and MFCCs from the dataset to build two CNN models and compared the result using these features. After experimenting with different architectures, we built the sequential CNN model following the architecture shown in figure 7.3 with the best accuracy and loss value. To measure the performance, we show the accuracy and loss of the model for each epoch. We tested our dataset with 20% test data. We also describe the confusion matrix for each class using both features separately.

MFCCs

The CNN model executes with 100 epochs with a batch size of 64. After the execution, we achieved a 94.44% accuracy and an average loss value of 0.3342 on the test

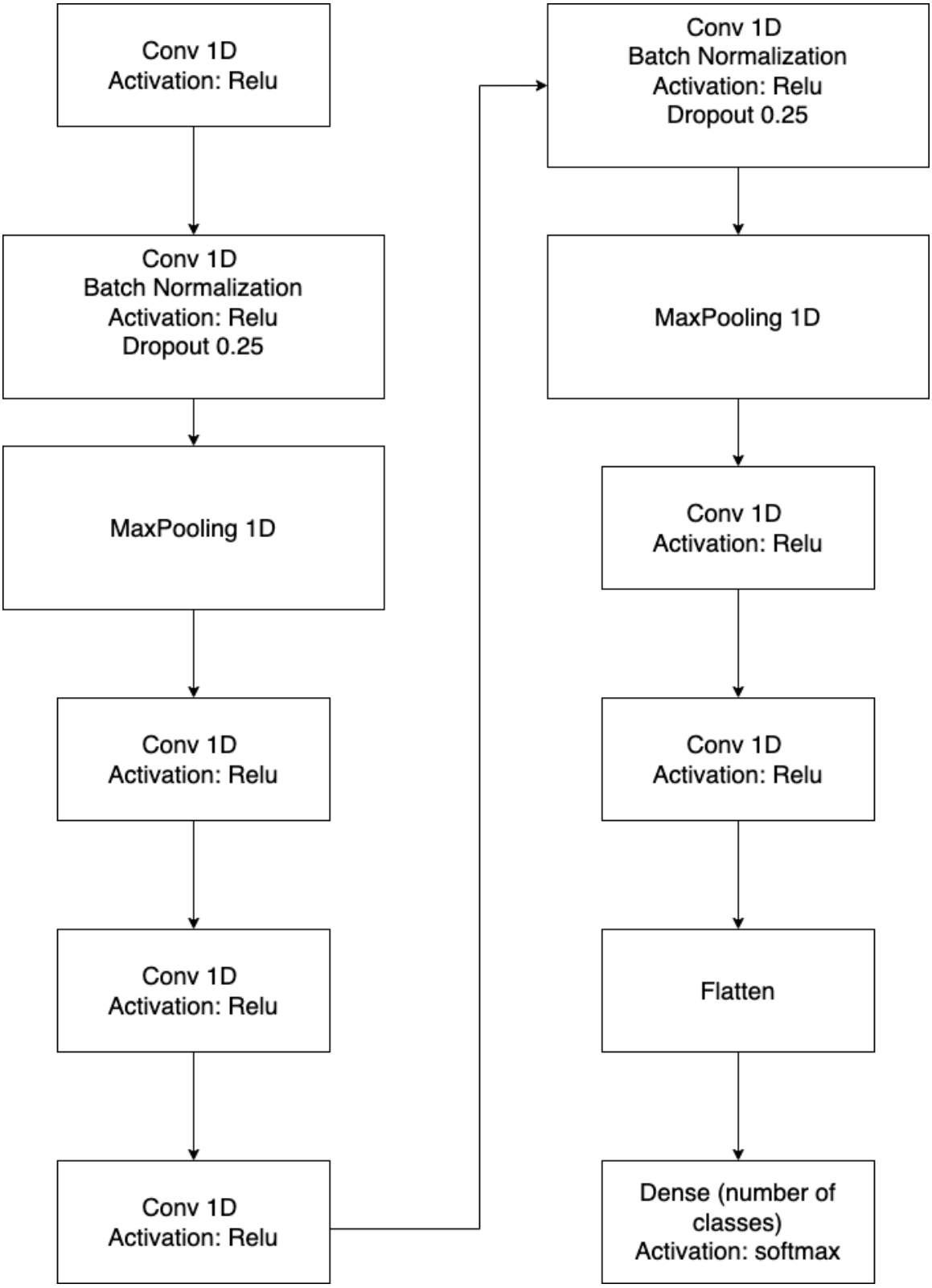


Figure 7.3: CNN architecture for RAVDESS dataset

dataset with the MFCCs feature. Figure 7.4(b) shows the accuracy in each epoch, and figure 7.4(a) shows the loss value change per epoch. The graphs show significant performance improvement until epoch 30, after which the learning rate did not increase. Figure 7.7 shows the confusion matrix of the CNN model prediction with MFCCs features. The overall predictions for all the classes are satisfactory, but we see some confusion around some emotions. "Sad" emotions are confused with "fearful" for a significant time. "Surprised" is sometimes confused with "disgust" and "happy".

MEL Spectrogram

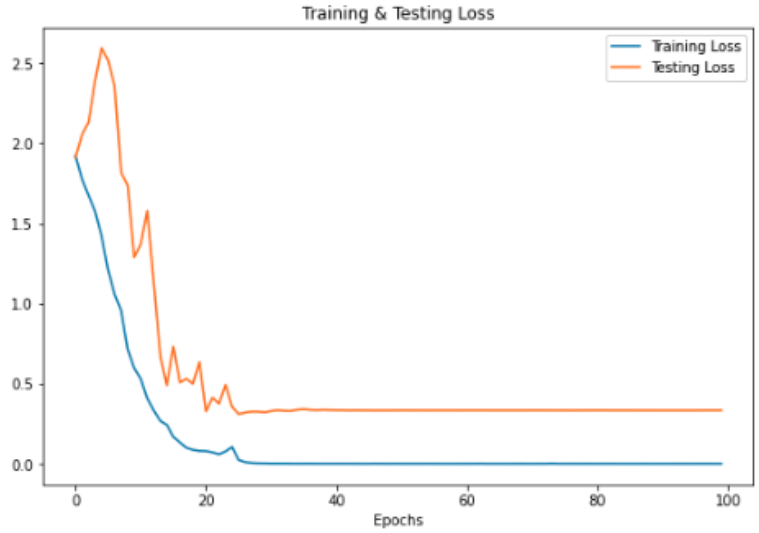
We achieved relatively lower accuracy (64.86%) and higher average loss value (0.9548) with mel spectrogram features. Figure 7.6(b) and 7.6(a) show each epoch's accuracy and loss, respectively. The difference between training and test accuracies suggests that the model may overfit and, thus, is not practical for real-life emotion detection. The MFCCs, on the other hand, show better results and thus can be deployed as a practical model.

7.2 Environment detection

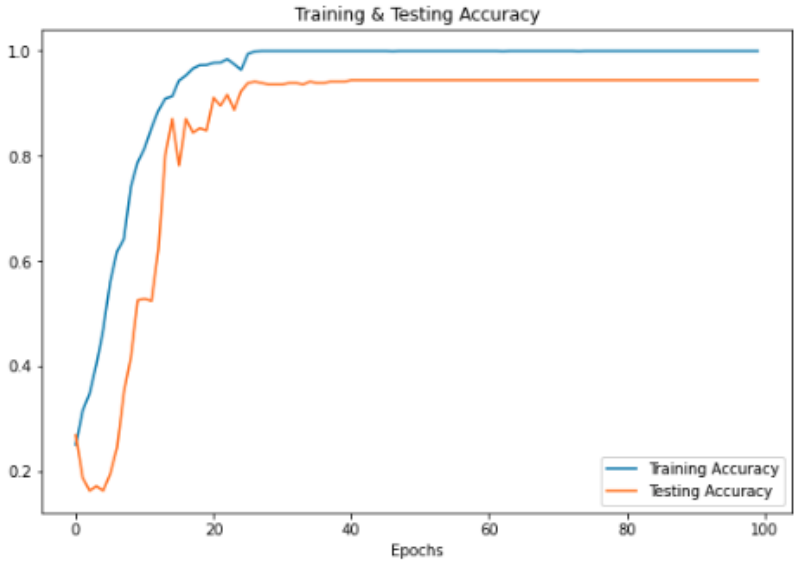
The presence of a person in a high-noise environment not only results in hearing loss but can cause cardiovascular disease, low-quality sleep, stress, and mental health problems. Exposure to sound from railway or aircraft traffic significantly affects blood pressure, hypertension, and stress [123]. So, it is essential to control exposure to a noisy environment. We aim to detect different environmental sounds to understand the effect of the environment on the user's stress and fatigue levels.

7.2.1 Dataset: ESC-50

The ESC-50 (Environmental Sound Classification) [115] is a dataset containing environmental audio recordings from 50 different sources. The dataset contains 5-second recordings from each class and 40 examples per class, aggregating a sample size of 2000. Out of the 50 classes, we selected 17 categories of environmental sounds. The categories are shown in Table 7.1.



(a) Training and Testing Loss



(b) Training and Testing Accuracy

Figure 7.4: Accuracy and Loss using MFCCs from RAVDESS dataset

Airplane	Brushing Teeth	Car Horn
Chainsaw	Chirping Birds	Clock Alarm
Coughing	Dog	Engine
Helicopter	Laughing	Rain
Siren	Thunderstorm	Train
Vacuum Cleaner	Washing Machine	

Table 7.1. Filtered categories from ESC-50.

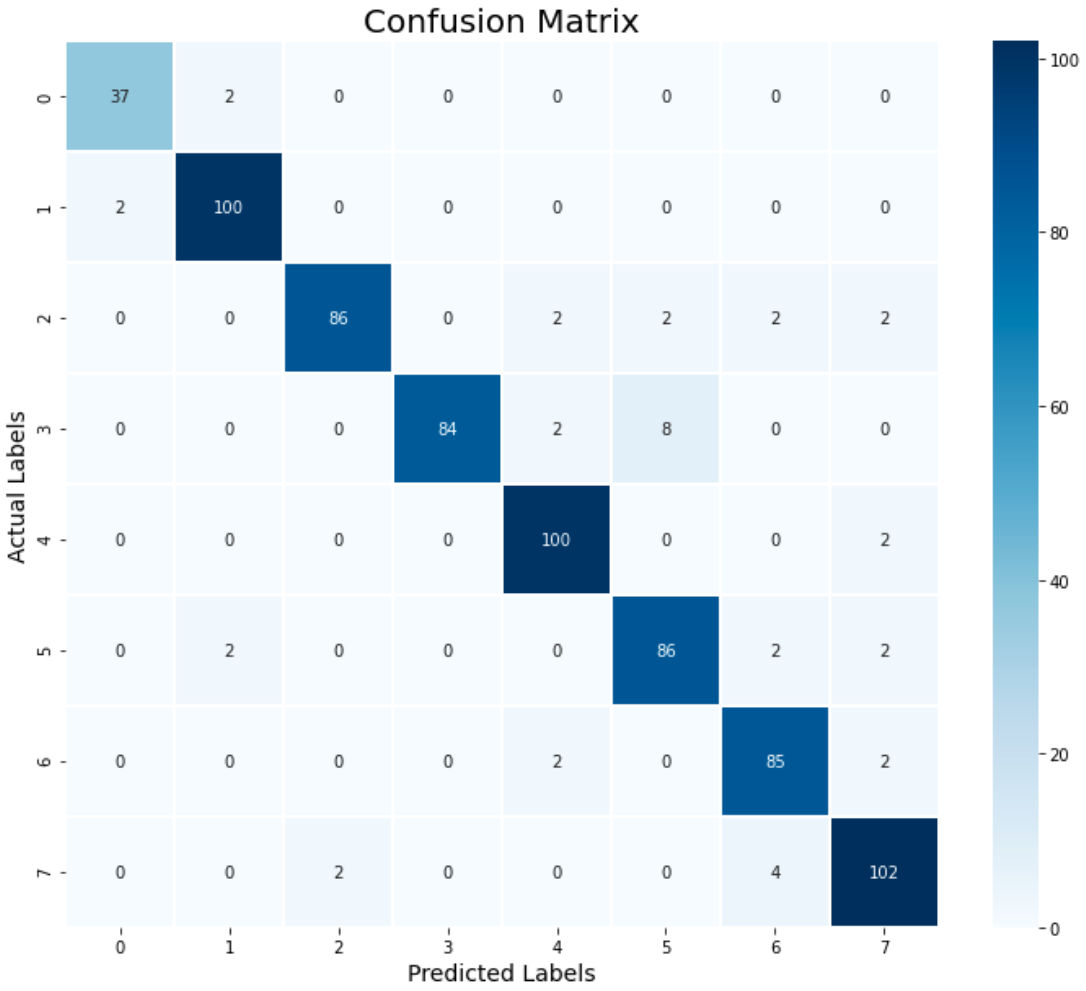
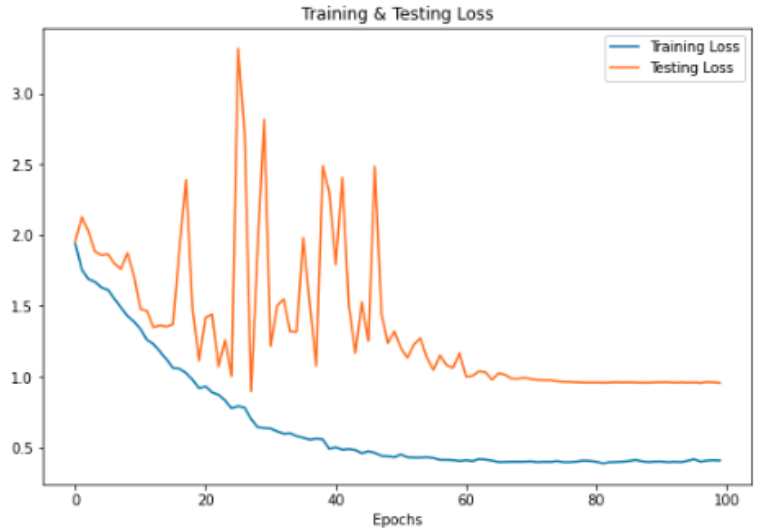


Figure 7.5: Confusion Matrix with MFCCs features (0 = neutral, 1 = calm, 2 = happy, 3 = sad, 4 = angry, 5 = fearful, 6 = disgust, 7 = surprised)



(a) Training and Testing Loss

(b) Training and Testing Accuracy

Figure 7.6: Accuracy and Loss using Mel Spectrogram from RAVDESS dataset

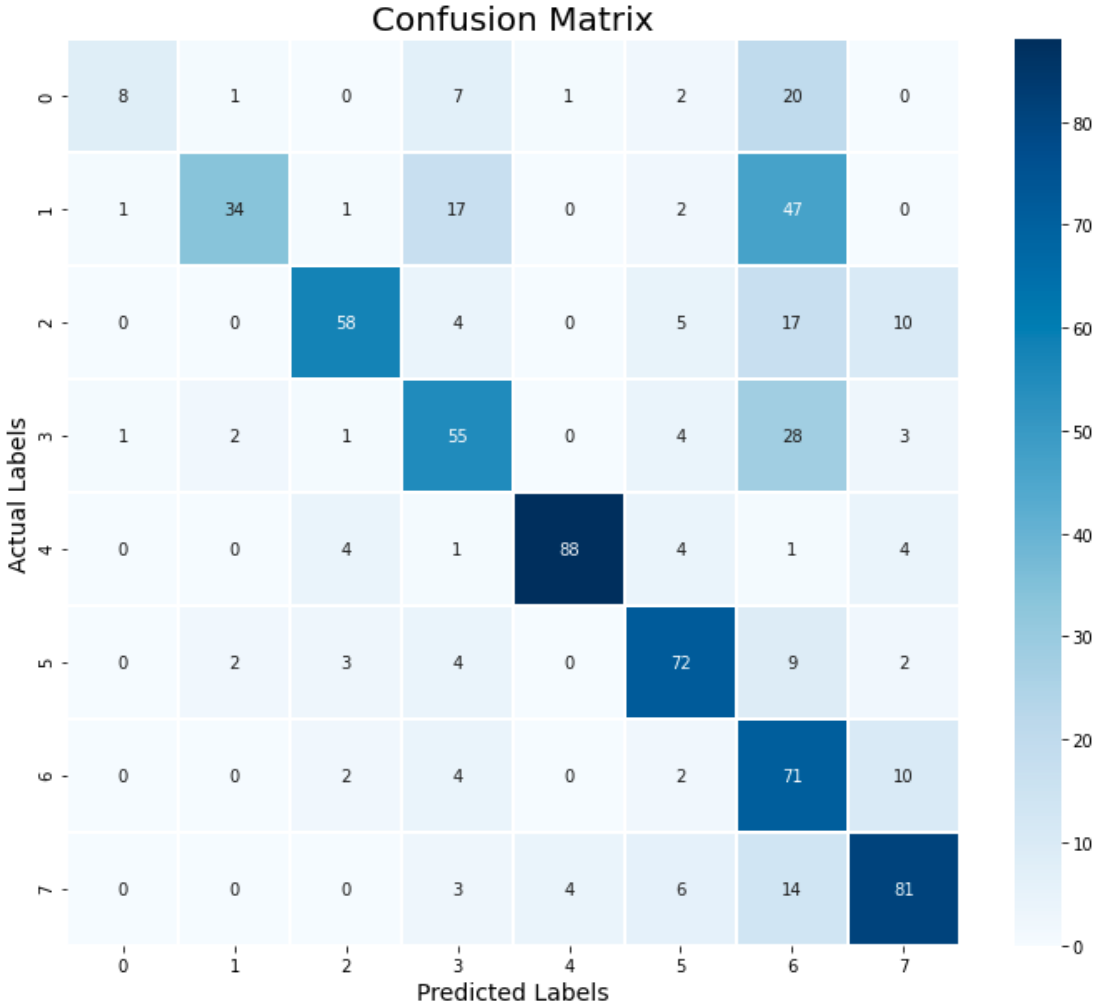


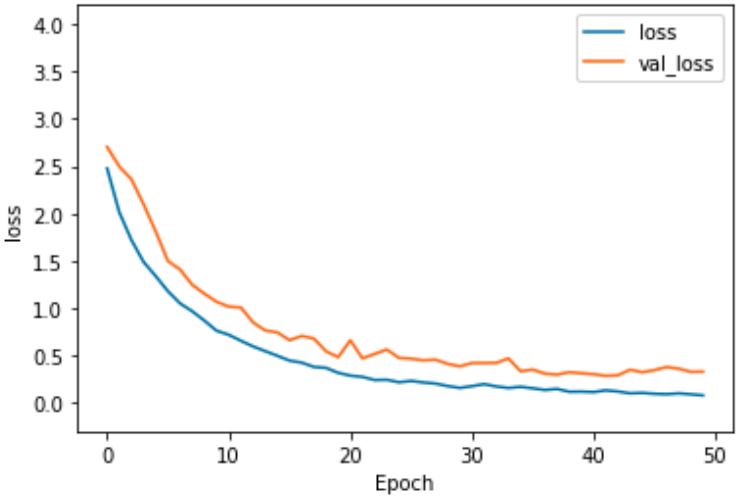
Figure 7.7: Confusion Matrix with Mel Spectrogram on RAVDESS dataset (0 = neutral, 1 = calm, 2 = happy, 3 = sad, 4 = angry, 5 = fearful, 6 = disgust, 7 = surprised)

7.2.2 Analysis

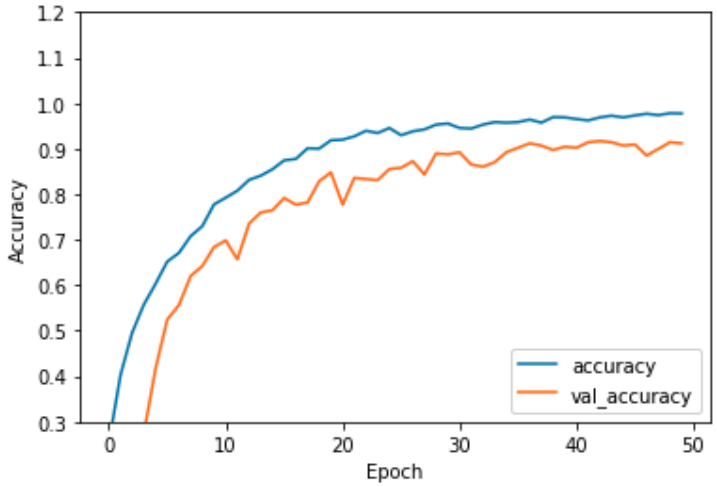
We extracted 40 mel frequency cepstral coefficients (MFCCs) from the environmental dataset and trained a CNN model with four hidden convolutional layers with batch normalization. Figure 7.8(a) shows the loss value on different epochs. The model predicts the 17 classes with 91.18% accuracy and 0.3294 average loss. The confusion matrix is shown in figure 7.9.

7.3 Discussion

In this chapter, we presented our findings from the experiments with eight human emotions and seventeen important environmental sounds. We extracted the mel frequency cepstral coefficients (MFCCs) and mel spectrograms from the human speech audios with various emotions and got the best result with MFCCs features. The accuracy significantly dropped when we used the mel spectrogram features. We also classified different environment sounds with reasonable accuracy. The patients need to limit their presence around environments that negatively affect their physical and mental health. An intelligent system that can track the presence of an individual in risky environments and detect an emotional change from human speech can warn the user and prevent an unexpected situation.



(a) Training and Testing Loss



(b) Training and Testing Accuracy

Figure 7.8: Accuracy and Loss using Mel Spectrogram from RAVDESS dataset

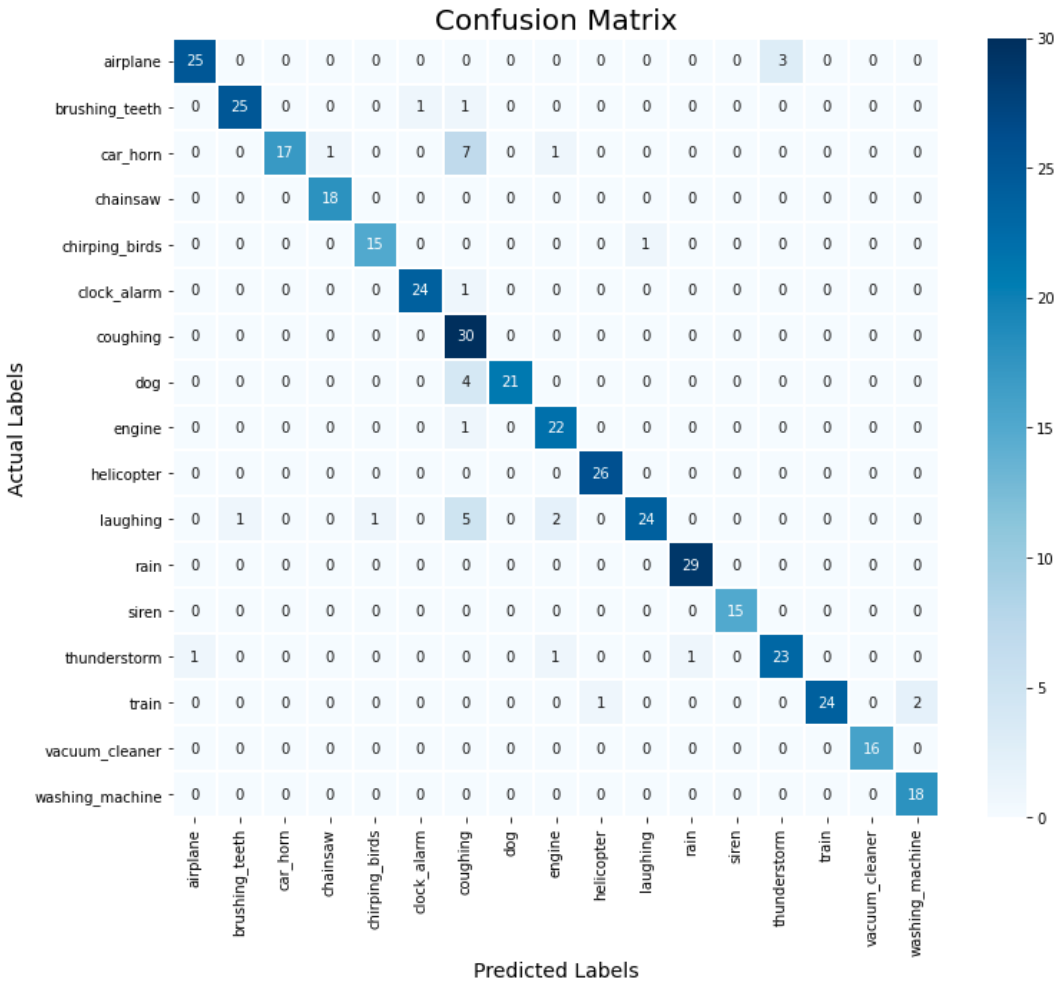


Figure 7.9: Confusion Matrix with MFCCs features for ESC data

Chapter 8
Design and Development of a System Package with Smart Insoles, Metamotion and Smartphone Application

8.1 System Architecture

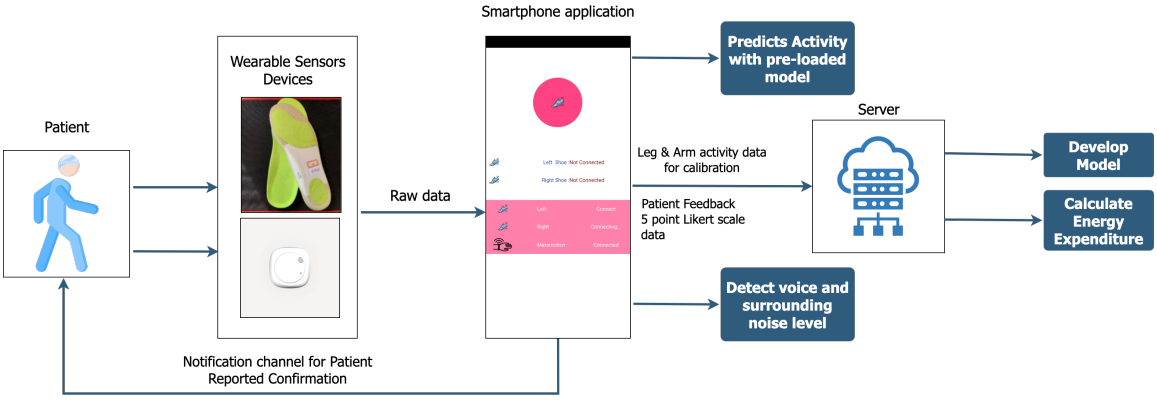
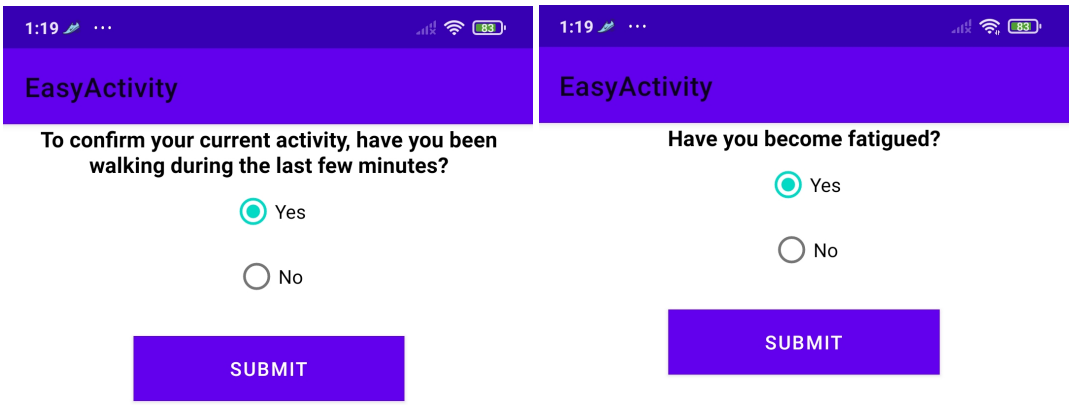


Figure 8.1: Diagram of developed system

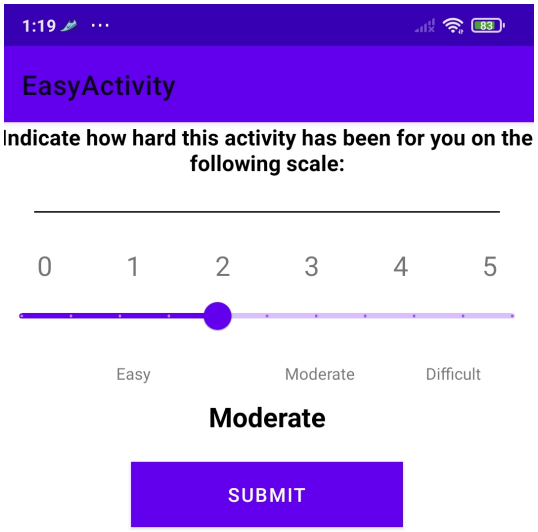
All the activities we detected can lead to severe fatigue for Chronic Fatigue Syndrome patients or Long Covid patients. A high amount of low-level physical activity can limit the patient's ability to perform their daily physical activities. So, they need to be able to measure and increase their awareness of their daily amount of performed activities. There are many subjective techniques available, but they may contain self-reported bias and thus need to be more accurate [63]. An objective measurement is necessary. We built a system combining the smartphone, the smart insoles, and the wrist motion sensor device to detect and objectively measure activities to reduce the possibility of a next-day crash.

Figure 8.1 shows our developed system diagram. The patients are the users of our system who will wear the smart insoles and wrist motion sensor device and connect the devices with the proposed smartphone application via BlueTooth. Figure 8.3(a) shows



(a) Confirmation of activities

(b) Confirmation of fatigue



(c) Perceived Exertion

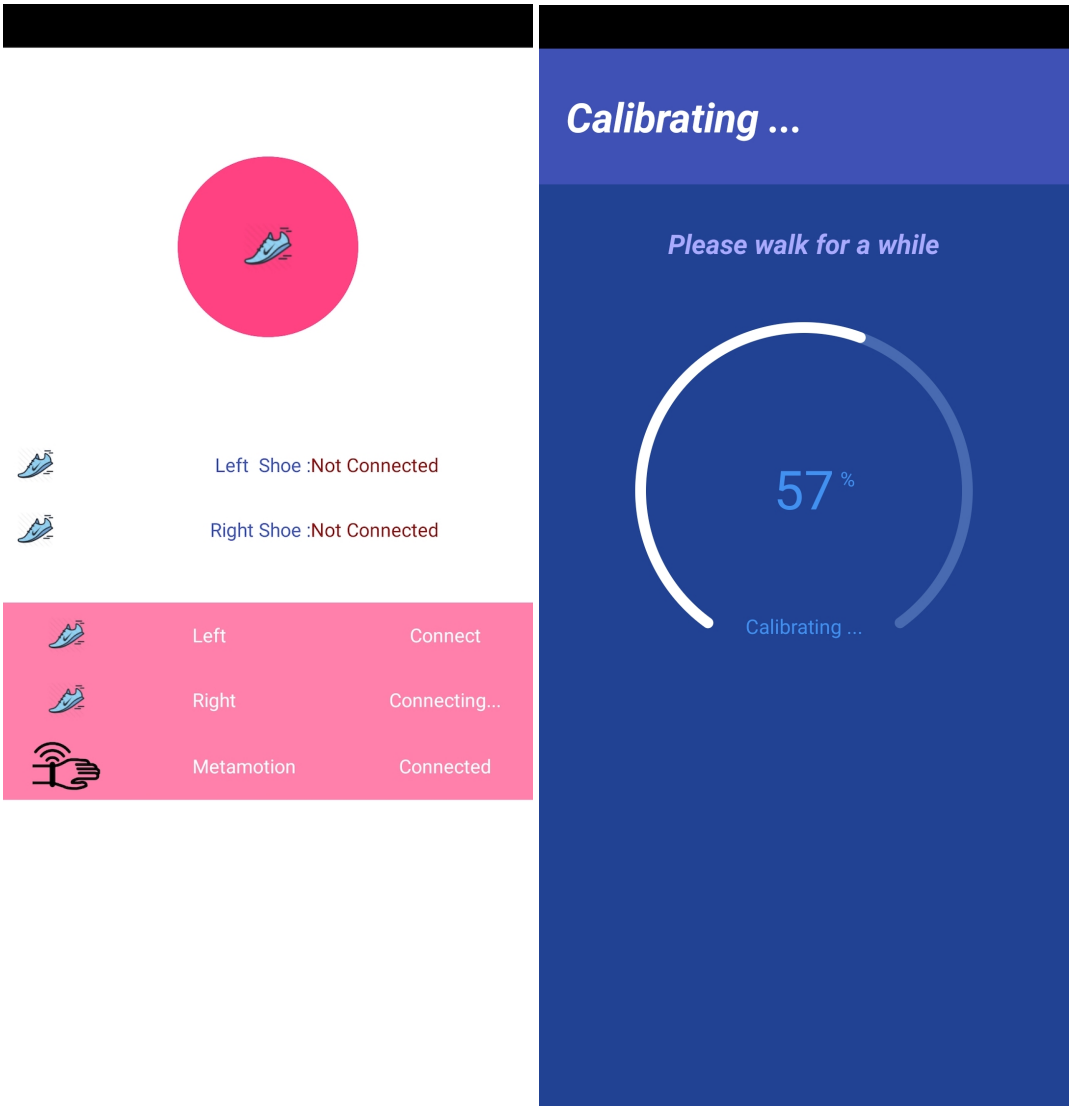
Figure 8.2: Patient Reported Confirmation

the device connection module. The application is the main component of our system, with various responsibilities. It collects the sensors' raw data and predicts the performed activities with pre-loaded machine-learning models in real-time. The application utilizes the built-in smartphone microphone to detect voice activity and exposure to high noise levels. It has a calibration module activated when the user connects to the device for the first time and asks the user to perform certain activities. After the calibration process is completed, the app sends the sensor data to the backend server. The backend server generates a model for that individual, which is loaded into the application. Figure 8.3(b) shows the calibration process developed in the smartphone.

There is a patient-reported confirmation feature developed in the smartphone application. When the application predicts an activity performed for a significant duration, it sends an activity confirmation notification to the user. This feature has two objectives. The first one is to improve the machine learning model's performance by getting confirmation from the user. The second objective is to get a measure of exertion. The app asks the user to fill up a subjective questionnaire as shown in Figure 8.2(a). It gets the confirmation of fatigue and breathlessness from the user (Figure 8.2(b)) and asks the user to scale their exertion on a scale of 0 to 5, 0 being "No Exertion" and 5 being "maximum possible exertion that can be perceived" (Figure 8.2(c)). The app saves the duration of the detected activities in its local database to calculate the energy expenditure.

8.2 Energy Expenditure Estimate

We aim to follow the related works to detect physical and social activity and measure the users' exertion and energy expenditure. We found several subjective measures in the literature using questionnaire-based measurement, but they are prone to bias. We were interested in objective measurement using sensors to get an accurate result. Regression analysis-based techniques were used in [73, 74] to estimate energy expenditure on a specific set of activities without any detection algorithm involved. As humans can perform a wide range of activities, energy expenditure measurement techniques with



(a) Device Connection (b) Calibration

Figure 8.3: Smartphone Application Demonstration

Activity Type	METs
Sitting	$1.0-2.0$
Standing	$1.2-2.3$
Walking	$0.0272*walkingspeed(m/min)+1.2$
Running	$0.093*runningspeed(m/min)-4.7$

Table 8.1. MET value for activities with different intensities .

limited activities may need to be revised. So, it is essential to get the movement of different body parts to detect complex activities involving both hands and legs. We found a few systems that use wearable sensors to detect activities and then measure the energy expenditure [69, 124]. However, they did not consider the effect of social exertion on fatigue.

8.2.1 METs

Sensor-based energy expenditure (EE) estimate techniques are based on approximation. The Metabolic Equivalent of Task (MET) is a standard intensity measurement unit. An established range of MET [124] set for different categories of activities is shown in Table 8.1. The MET value can range from 1.6 to 2.9 for light physical activities, and 3.0 to 5.9 for moderate physical activities [125]. With the duration of detected activities and the approximate values of METs, we can objectively estimate EE from Equation 8.1.

$$Energy(kcal) = 1.05 \times METs \times Weight(kg) \times Duration(hours) \tag{8.1}$$

Also, our application's patient-reported confirmation module (Figure 8.2) provides a subjective measurement of perceived exertion. The Likert scale from Figure 8.2(c) measures the perceived exertion on a scale from No Exertion to Maximum Exertion, as perceived by the user. The energy expenditure estimate and perceived exertion of activities will be shown in a comprehensive report in the application.

8.3 Improving smartphone resource usage by background services

As the smartphone application has many responsibilities to maintain, we need to find an efficient way to handle its tasks. The application need to communicate with the server regularly and read data from the sensor in real-time. If the devices are set at a 20Hz sampling frequency, there will be 20 readings from each of the insoles and 20 additional readings from the wrist device sensor. All the heavy tasks may cause the smartphone's battery to drain faster. So, we explored some performance improvements to handle the responsibilities of the application efficiently. The application runs two different background services with the insoles and the metamotion to collect the data from the sensors. We have another background service recording surrounding noise to detect voice activity and measure the sound level. When the users lock the smartphone and put it in their pocket, a foreground service cannot run and perform any task. In this scenario, a background service is used to perform a task without showing any UI response to the user. The background services in our application continuously collect data from the insoles and the metamotion until the devices are disconnected, or the user manually stops the sensors. The microphone can record surrounding sound in the background for a long time. These operations take many system resources. So, users will have the option to disconnect and reconnect the devices at any time. We have implemented this feature in the device connection module in the application. The application performs all the activity detection and sound listening with the user's consent.

8.4 Machine Learning Model and Communication with the server

Our system relies heavily on machine learning algorithms. The machine learning models continuously learn from new data in the server, and with each update, are included as pre-loaded models in the application. To get real-time activity detection using machine learning models, we had the following options:

8.4.1 Send the sensor data to the server at a regular interval

The application must communicate with the server regularly to get real-time detection. The server predicts the activities with the pre-trained model and sends the result back to the application. An application-server communication can take some time to complete, thus increasing the system's latency. The application may have created a new prediction request and added it to the request queue before the previous request was completed. All these operations will consume more device resources and drain the battery faster. Sending raw data to the server also causes high data bandwidth usage. Figure 8.4 shows the communication flow of a batch prediction request between the client application and the backend. With a 2-second segmentation of data, the application sends at least three prediction requests with a large number of raw data to the server. We have three prediction services running in the backend, which process the raw data and send the data to ML models for prediction. The ML models send the result back to the application and save the result in the database. All operations need efficient handling of data and requests to optimize the system's performance, maintaining lower resource usage.

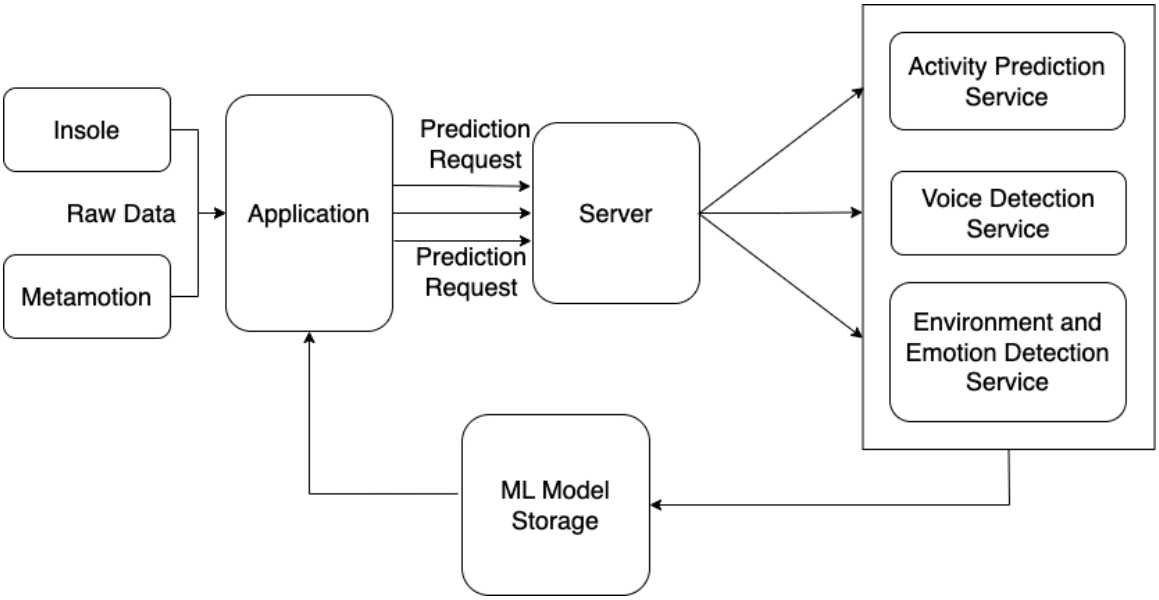


Figure 8.4: Flow of continuous prediction request to server

8.4.2 Pre-load the machine learning model in the application

Modern smartphones can run machine learning algorithms on the phone. We can pre-load our activity detection models when the app is installed. Instead of sending the data to the server, the application can communicate with the pre-loaded models to get a prediction. Also, we run a calibration module on the smartphone when a user installs the application and connects the devices. The calibration module trains a new model for the user, which can be pre-loaded into the device and used to detect activities. This solution can reduce the latency caused by application-server communication but increase the computing complexities of the application. Figure 8.5 shows the system flow for the pre-loaded model. The feature extractor module in the flow receives raw data from the sensor and preprocesses them to extract features. The extracted features are sent to the pre-loaded models in the phone depending on the source and the requirement. The pre-loaded models generate and send the predictions back to the application UI, where the result is displayed. The results are also saved in the application's local database and a remote HIPAA-compliant database. The feature extraction module can cause a significant problem in real-time detection. Though the module can perform complex computations on raw data, it creates a heavy load on system resources for all the computational activities. Performing complex operations every 2 seconds can create system latency if data are not handled well.

8.4.3 Our approach

In our research, our main objective is to provide an on-demand comprehensive summary of performed activities. To provide an on-demand summary, we can be flexible around real-time activity detection to save computational resources. So as an alternative to real-time activity detection, we decided to keep the sensor data in the smartphone memory. The app will contact the server with the necessary data whenever the user asks for a summary. The server will perform necessary computations, return a summary of detected activities with the duration, and estimate the energy expenditure. The

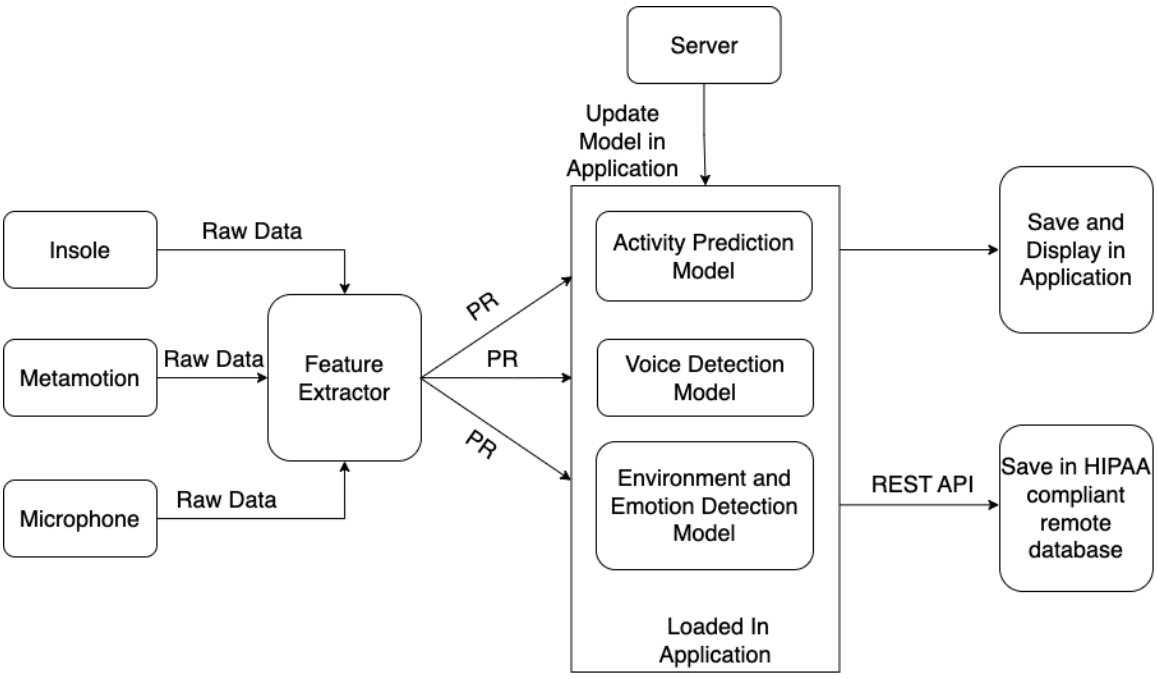


Figure 8.5: Load models locally in application

server communication can also be done in the background when the application detects less movement in the sensor data or during device disconnect. Figure 8.6 shows our approach to reduce system latency while maintaining the requirements. The predictors' output is always saved in a secured database on the server and shown in the application UI. After the result of a batch-prediction request is saved, we remove the raw sensor data from smartphone storage. As a practical solution, we have implemented the on-demand detection feature in our application as the default option, as users are less likely to interact with the application when performing any activity. However, we will also allow real-time detection, but users must select this option if they want it.

8.5 Handling Large Concurrent Traffic

While engineering the backend, we had to remember that the system is expected to handle thousands of concurrent user requests per second. We have to make sure the system is reliable and always available. We added more servers in the backend and made a server pool to scale the system. To use the server pool efficiently, we added a

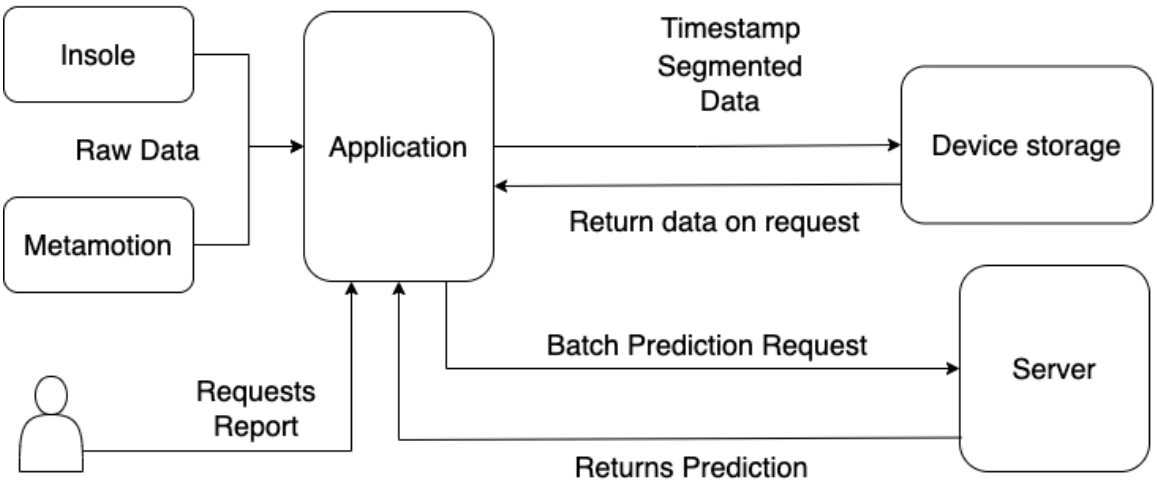


Figure 8.6: High level design of our approach to reduce system latency for on-demand summary

load balancer in front of the server pool. A load balancer distributes incoming requests to a pool of servers and ensures that no server is overloaded. This approach also ensures that other servers will take charge to make the system highly available if any server goes down. Figure 8.7 shows the scaled diagram of the backend. The request from the users first goes through the load balancer, which distributes traffic to the available servers. The distribution algorithm depends on the system's needs. In our case, our load balancer sends a request to a server with the least network connections.

8.6 Calibration

The application starts with a welcome screen, then asks the user to provide basic demographic information. Then it takes the user though the steps to connect the devices. The devices include the left and right insoles and the metamotionC. The application checks if Bluetooth is enabled. If not, then it asks the user to turn the Bluetooth on. After enabling Bluetooth, the application searches for available devices to pair.

In the calibration module, the user is asked to perform 2 minutes of walking, 2 minutes of running, 2 minutes of hand exercise, and 2 minutes of chopping. This helps the application to learn the movement of the user. The calibration progress can be saved

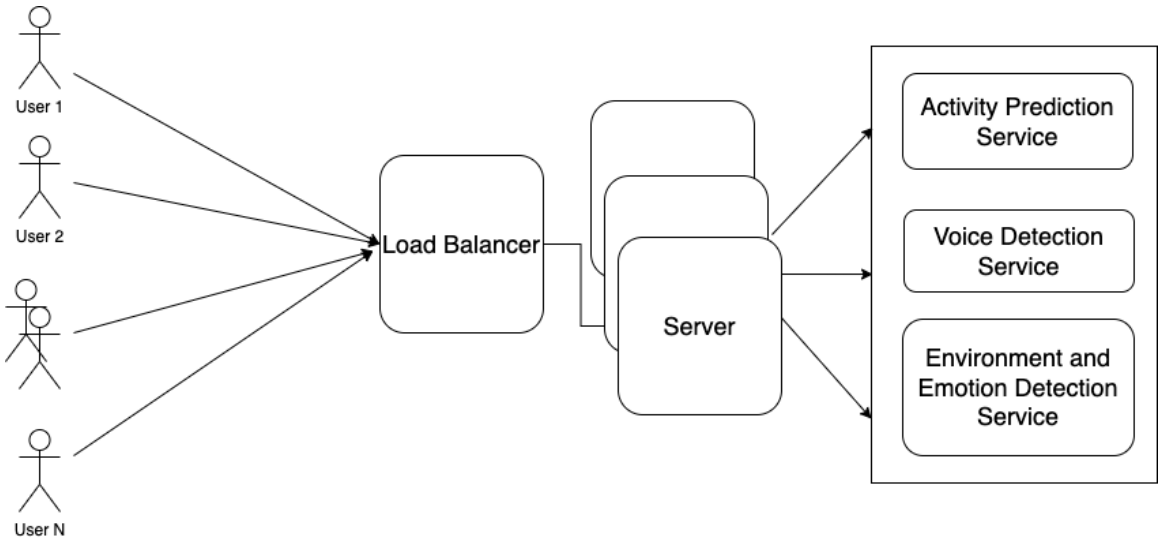
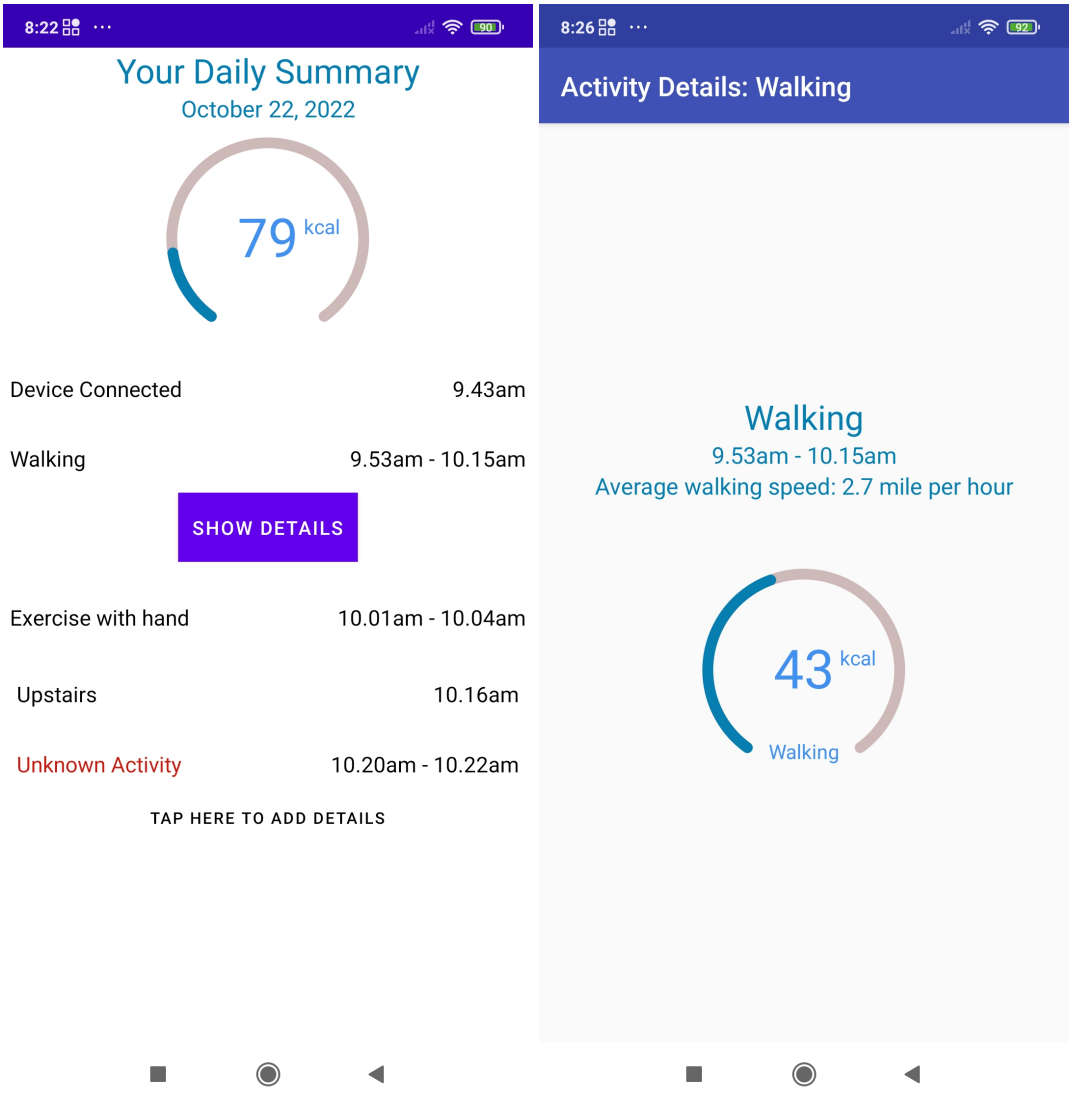


Figure 8.7: High level design of our approach to reduce system latency for on-demand summary

and resumed later. Also, this is an optional feature, as an elderly patient might not be able to complete all the steps. After the calibration step is done, the application sends the data to the server. The data will be used as retraining data for the machine learning models.

8.7 On-Demand Comprehensive Summary

The comprehensive summary in the application shows the time and duration of each performed activity the app can predict. From the duration, users' weight, and activity type, the application estimates energy expenditure and displays that information in the UI. In Figure 8.8(a), we show the daily summary and duration of performed activity for a particular day. The "Show details" button appears when the user taps on a particular activity. Tapping on the button will move the user to another UI, shown in Figure 8.8(b), that shows some additional information and estimated energy expenditure for that particular activity. We also provide an option to get a report from any previous day. The 'unknown activity' in Figure 8.8(a) represents an activity that our machine learning models could not predict. We ask the user to add details for that activity, such as activity



(a) Overall Summary (b) Activity Details

Figure 8.8: Comprehensive Summary of Performed Activities

name and level of fatigue, and feed the information to our machine learning model for future prediction.

8.8 Maintaining and Updating Machine Learning Models

The advantage of using different sensor devices as data sources is that we can extend our system to support a wide range of physical activities. Users will continuously add new data to the system, which we can use to retrain our models. We can get new data

on already-known activities from the calibration module. We will get data on new activity from the 'unknown activity' from the comprehensive summary and patient-reported confirmation module. We will regularly feed the new data to the machine learning models to keep them updated and increase their performance.

Model update or retraining should be done in one of the following scenarios:

- The accuracy of a model decreases significantly or below a threshold
- On arrival of a large amount of new data

As sensors continuously send raw data to the server, we expect to have a large amount of data daily. So we set up an automatic model retraining job in the server at midnight to check the data quality and retrain specific models. The updated models are saved in the model storage if they show better validation accuracy. The model train and retraining flow is shown in figure 8.9. Whenever there is new data, it goes through the model retrain strategy to decide if the model needs to be retrained. We are scheduled to manually retrain the models if we get low validation accuracy on retraining. If we get low test accuracy on the new data, the model is retrained too.

8.9 Remote Monitoring Elderly Patient

Remote monitoring technology for managing elderly patients can benefit them by preventing symptoms related to chronic diseases. Elderly patients may find it difficult to visit a healthcare provider for consultation whenever they have fatigue or malaise symptoms. Some older adults also face the barrier of communication and understanding technology to self-diagnose their current health situation, even with a proper diagnostic tool. Our proposed system tracks the patients' activity patterns and estimates energy expenditure to let them know their daily activity summary. However, some older patients may find it hard to track and self-diagnose their condition using technology. So a remote monitoring system is needed to manage these patients who need the support

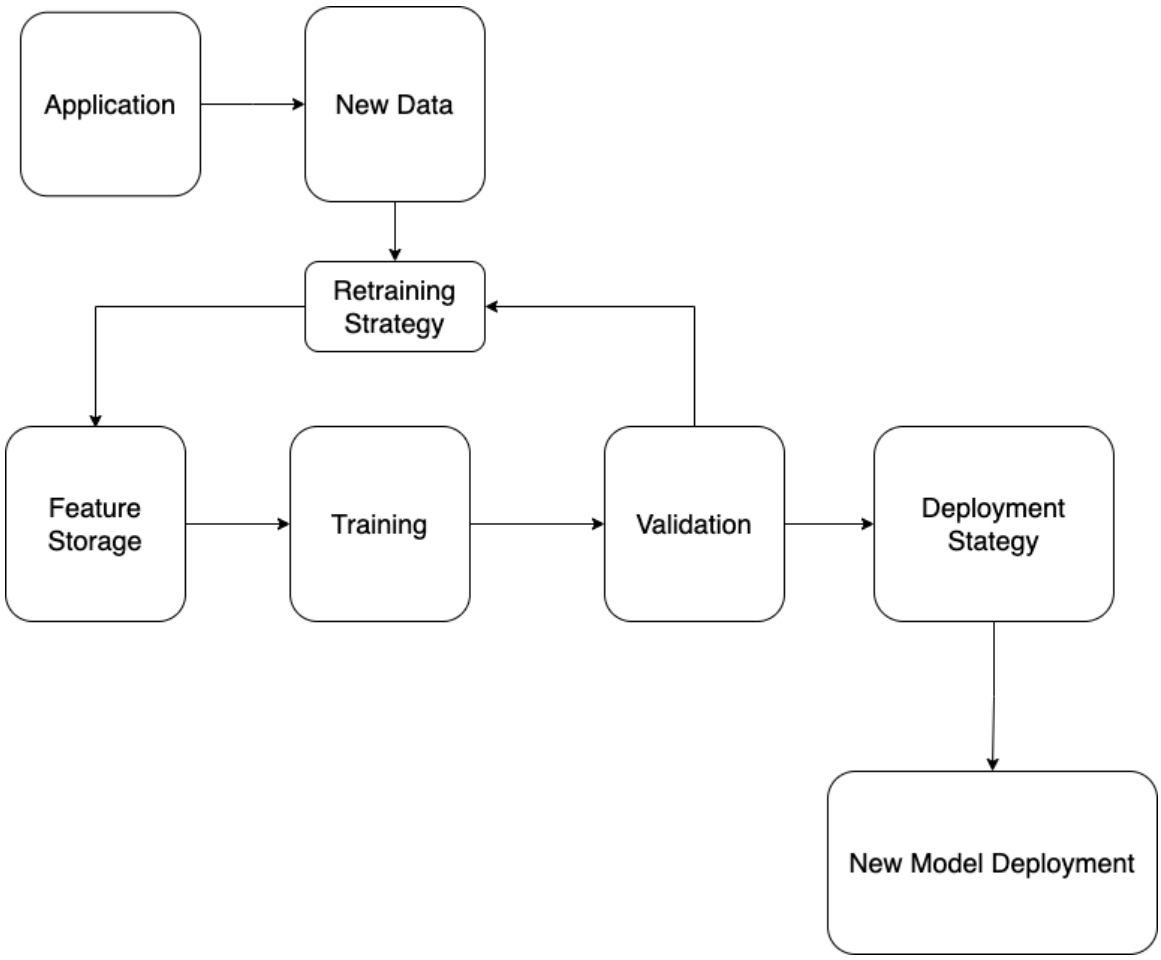


Figure 8.9: Model Retrain Strategy

of caregivers and family members in difficult situations. Using our sensor-based activity detection and measurement setup, we developed a remote monitoring tool that can alert caregivers and family members of any abnormal findings about the patient. We extended our object measurement system described in section 8.1 and implemented the following features to develop the remote monitoring tool:

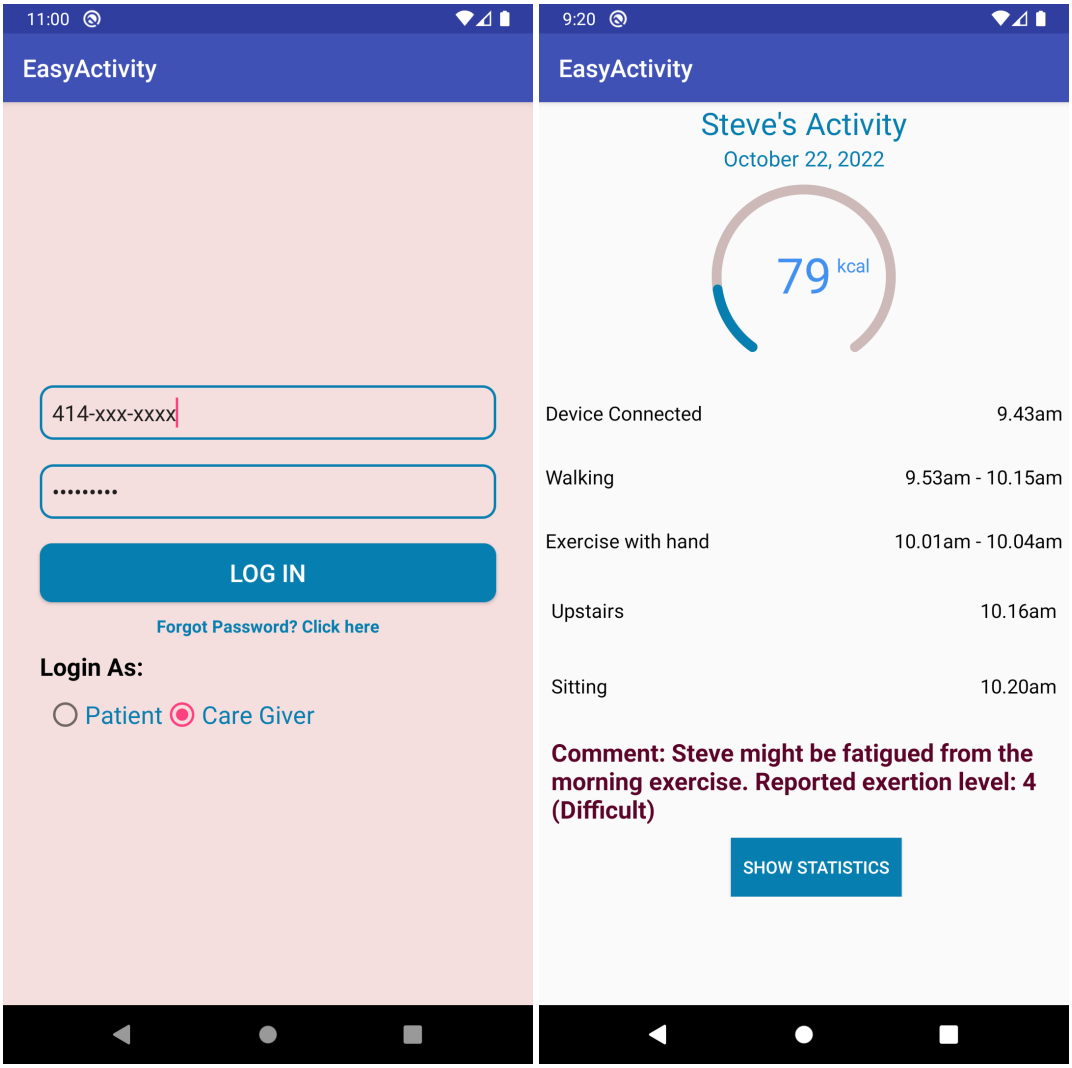
- **Login as Caregiver or Family member**: Caregivers are paid or non-paid members of a person's network who can help a person perform daily activities. A family member can also play the role and help patients manage their chronic diseases. In the remote monitoring system, the caregiver, on behalf of the patient, can log in and view the patient's current condition. Figure 8.11(a) shows the login screen

Figure 8.10: Alert for unusual activity sensing

of the remote monitoring tool where an individual can sign in as a patient or caregiver. After logging in, the caregiver can see a list of assigned patients and check their statistics, reports, and current health conditions. As the application is connected to the server and can send real-time information about the patient, it can also report any irregularity found in the patient's daily activities. Elderly patients living alone with limited physical capability can benefit from this use case when they need help but do not have enough self-care ability.

- **Notify caregivers of unusual activity:** The remote monitoring tool notifies the caregivers and family members in case of over-exertion or unusual activity. In the event of any emergency or abnormal sensing, the application sends an alert to the application of the caregiver via the notification system if the caregiver and the patient are not in close proximity. Figure 8.10 shows a sample alert notification sent to the caregiver about the cared patient's probable condition.

- **View current condition:** Family members or any other authorized person can view the current condition of the patient from their application. The comprehensive report from the patient application is sent to the authorized person if they request it. Figure 8.11(b) shows the patient report on the caregiver application. It shows similar information as the patient's application and comments about the patient's current status. In the future, we will implement an auto-generated recommendation system based on the patient's current condition and health history. Figure 8.12 shows the high-level system diagram of the remote monitoring tool to



(a) Login (b) Patient report in caregiver app

Figure 8.11: Remote Monitoring Tool

manage the patient's chronic conditions. There are two main modules: Patient and Caregiver. These two modules communicate with the cloud server to share patient information. In the backend, the server is HIPAA compliant, and the necessary authentication protocol has been followed to secure the patient information and ensure no unauthorized third party can access patient information saved in the cloud.

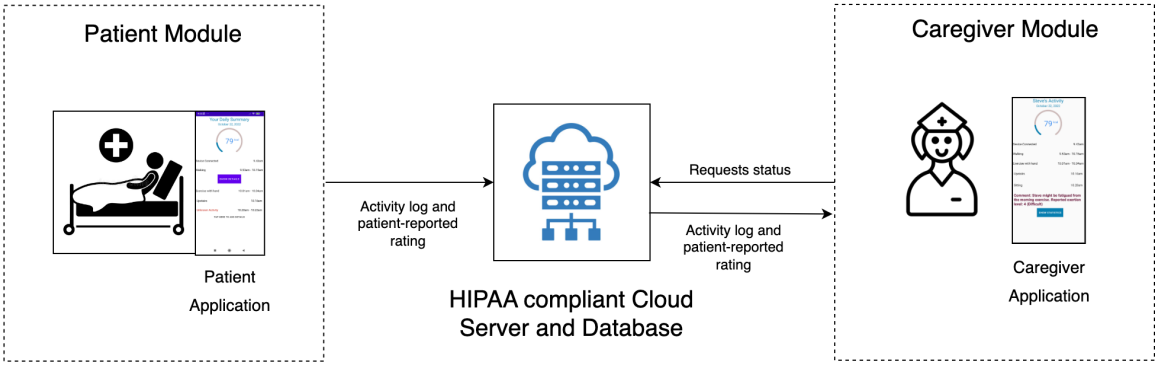


Figure 8.12: Remote Monitoring Tool

8.10 Six Minute Walk Test

The six-minute walk test (6MWT) is a standard test to assess physical capacity, especially for older adults. The American Thoracic Society first introduced it in 2002 [126] with a practical guideline to objectively evaluate a person's functional capacity. This test aims to assess a patient's capability to walk as far as possible in six minutes. Patients might be breathless or exhausted during a six-minute walk, so they can slow down or stop. The distance covered in six minutes can be interpreted to describe a person's capacity to perform daily activities of living.

The distance covered in the six-minute walking test varies with health status and age. Significant differences were found in the 6MWT-distances for elderly persons [127]. So, the six-minute walk test is a good representation of a person's health status and can be used to set an initial limit for their daily exertion. As our system already has an integrated leg activity detection module, we could extend this system to implement a smartphone-based six-minute walk test and use the result to assess an individual's physical condition and health status.

8.10.1 Implementation of Six-Minute Walk Test Using Our System

We implemented the following components to run a 6MWT test properly:

- **Instructions:** On the first screen of the test, we briefly inform the users about the

test and instruct them on how to start the test. We advise them to take a rest if they feel breathless or exhausted. We put a start button on this page. We instruct the user to press the button when ready to start the test. Figure 8.13(a) shows the instruction screen of the six-minute walk test.

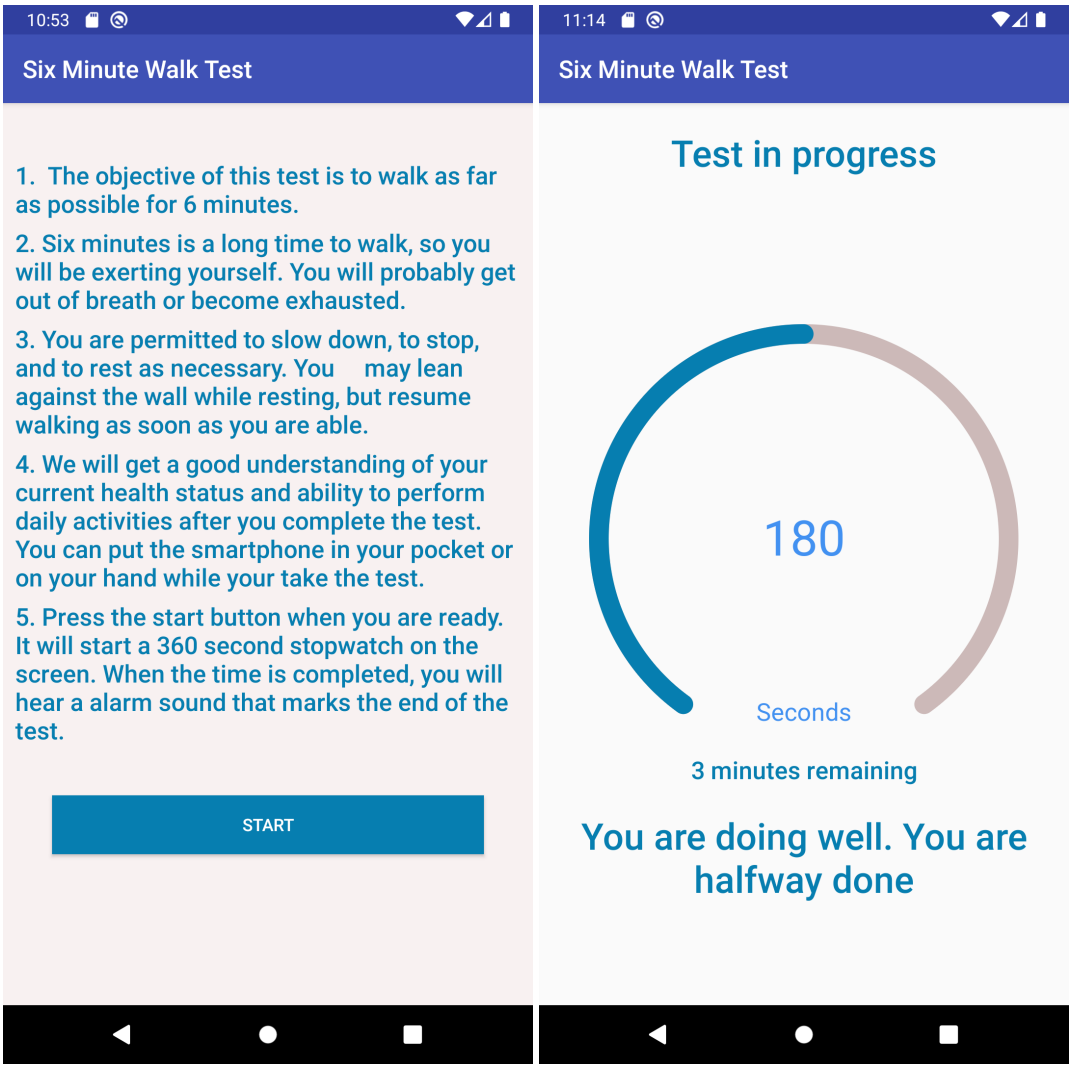
- **Stopwatch:** We implemented a stopwatch to measure the time of the test and let the user know after six minutes from the start of the test. We put a start button, tapping on which will start the stopwatch, and the user will be asked to start walking. After six minutes, an alarm will go off to let the user know the test is complete.

- **Encouragement during the test:** The American Thoracic Society (ATS) instructed to encourage the patient at each minute of the test with standard phrases of encouragement in even tones [126]. When patients walk, we show the encouragement phrases on the screen and notification bar. Figure 8.13(b) shows the encouragement phrases after three minutes of the test have passed. The patients can hold the phone in their hands or put it in their pockets. In the case of holding the phone in hand, they can read the encouragement text every minute. The following encouragement phrases are shown at different points of the test with the recommendation from the ATS:

  - After the first minute: "You are doing well. You have five minutes to go".
  - At the end of the second minute: "Keep up the good work. You have four minutes to go".
  - When 3 minutes remain: "You are doing well. You are halfway done".
  - At the end of the fourth minute: "Keep up the good work. You have only two minutes left".
  - One minute remaining at stopwatch: "You are doing well. You have only one minute to go".

– After six minutes is completed: "Well Done. Stop walking and take some rest".

Additionally, as our system can detect different leg activities, we check whether the patient is walking or resting (sitting or standing), then show some encouragement phrases to keep them going. In the event of rest, we record the timestamp for future research. The resting event indicates that the patient is exhausted and



(a) Instructions for six-minute walk test

(b) Test in progress - Encouragement Phrase at halfway

Figure 8.13: Six-minute walk test demonstration

might feel breathless. So we show an intermediate questionnaire to the patient at the time of rest to better understand their health status at that particular time.

- **Distance Measurement System:** The main objective of the 6MWT is to calculate the distance covered in six minutes. Our system can detect if a person is walking or resting. As a smartphone-based setup, we took advantage of the GPS to measure the distance walked every second. Android location manager provides an API that returns the updated location every n seconds or if it detects a change of d meter in the distance traveled from the last location update. We set the value n as 1 second and d as 1 meter. So we can receive the location update every second, or if the patient walks for 1 meter. Finally, at each location update, we measure the distance between the current and previous location and add it to the cumulative distance. At the end of the test, we show a report of the six-minute walk test and display the distance covered. We also show the patient a subjective questionnaire to understand their exertion during the test. We save the report and the patient-reported answer on the server for further analysis.

8.11 HIPAA compliancy

The Health Insurance Portability and Accountability Act (HIPAA) is an act to protect sensitive patient health information from being disclosed. Under this act, any database with protected health information must implement access control for security purposes. The engineers must implement backup strategies in case of system unavailability. The data and the data communication must be encrypted for security purposes.

As a healthcare system, which tracks and records patients' information and their daily performed activities with energy expenditure estimation, our system's backend must be HIPAA compliant. We have implemented necessary authentication protocols with encryption to secure our database. We are using Amazon Relational Database Service (Amazon RDS), which is HIPAA eligible, to store and transmit patients' protected

health information. We have also created multiple backups to the database to ensure high availability. All of these approaches make our software system compliant with HIPAA.

8.12 Technologies Used

We used the following technologies to build our system:

- To develop the smartphone application, we used the android platform and Java as the programming language. Android platforms come with in-build SQLite data to save data locally. We used a wrapper of the database: Room. The Room persistence library enables efficient handling of large numbers of structured data with less code. We store segmented sensor data with duration in multiple CSV files and save the location of the files in a structured model using Room. Whenever the user requests a summary, we query the database to fetch the saved data and send them to the server.

- We used Django REST Framework (DRF) to build the web services in python. DRF is a powerful tool for building web APIs for application-server communication. Through DRF, we exposed several RESTful Application Programming Interfaces (APIs) that can accept raw data and return the result to the application in JSON format.

- For machine learning development, initially, we used Jupyter Notebook for data visualization, feature extraction, and training and testing ML models. We used *scikit-learn* for machine learning and *keras* for deep learning approaches. After developing the models, we used *TensorFlow* to create production-grade ML models. We chose this platform because it provides the functionality to build and deploy the android compatible models in production.

- We deployed our backend system in Amazon Web Service (AWS). AWS provides a

reliable and scalable solution with *Amazon SageMaker* to build and deploy machine learning systems. We used Amazon RDS to store encrypted user data.

8.13 Discussion

This section describes the development of a system that can handle large and real-time sensor data to predict human activity. We developed the system to be easily scaled and extended for new unseen data. In the backend, we took precautions to make the system highly reliable and available by creating a server pool and multiple database replicas. The database replicas will be synced at regular intervals so that they can take over in case of any unexpected situation. We described how we plan to maintain the machine learning models to make them efficient and accurate. We developed features and data pipelines to use new data and retrain the models automatically using a scheduler. The calibration module, patient-reported confirmation, and 'unknown activity' from the report provide new data to the machine learning retrain algorithm. We have described the on-demand comprehensive summary module, which can be extended to support a wide range of reports. We made the system HIPAA compliant to protect sensitive user data. While developing the application and the backend, we always intended to use the latest technologies and practice recommended strategies. As we expect a tremendous amount of traffic in the backend, we applied necessary load-balancing techniques to prevent the servers from being overworked and shut down. With all the pipelines being set, we plan to extend the scope of the system for real-life usage and monitor the performance of different components to improve the system's efficiency and accuracy.

Chapter 9
Conclusion

9.1 Summary

This dissertation presents the development of different components of a practical tool with wearable sensor-based devices and a smartphone application to help patients avoid over-exertion. We used a pair of sensor-based insoles and a wristband to significantly decrease patient discomfort and remove existing challenges in this area. As both hand and leg movements are tracked, the system is highly extensible to include more activities in the future. We developed a smartphone application as the core component of the system with the capabilities of detecting voice activity and high noise exposure. We also developed two CNN models to detect human emotions from speech and detect surrounding environments. Every system component contributes to generating a comprehensive daily summary of performed exertion and energy expenditure. Users can access their personalized comprehensive daily summary report any time they want and track their energy expenditure history.

9.2 Contributions

This dissertation contributes to the design and development of a system to solve the current problems faced by patients with illness-limiting capacities. These patients need to be aware of their daily limit of functional activities. They need to know when they should stop before over-exerting themselves. Our comprehensive literature survey revealed much literature on activity recognition using sensors. Also, there is research where energy expenditure is calculated using commercial accelerometers. However, we found very few patient-oriented systems that use wearable sensors to detect activities and measure energy expenditure from those activities. We developed a system that can detect changes in exertion and alert the patients and their caregivers through the smart-

phone's notification system about their exertion level.

9.3 Broader Impact

Monitoring elderly patients' daily activities is an emerging topic with the recent development of sensor-based tools. Patient-reported survey tools are not practical for elderly patients because of the reporting bias in subjective methods. Older adults may have difficulties subjectively reporting their exertion level, resulting in inaccurate information. They need to be physically active to maintain a healthy lifestyle. However, they should also be concerned about over-exerting themselves. An objective and direct monitoring method using a sensor-based setup is essential in providing accurate information about performed physical activity. Such tools can warn the patients before they reach the limit of their exertion level.

Severe and very severe ME/CFS patients with limited functional capacities face many challenges, both physical and mental, from getting out of bed, leaving the house, and getting to medical appointments. Treatment can be frustrating when medical providers fail to properly consider their symptoms. Medical providers may lack enough information to diagnose and manage these patients, especially in the case of very severe CFS patients. These patients need proper care and direction on managing their disease to control or avoid the symptoms and maintain a regular life. Our system tracks physical and social activity patterns and corresponding user-reported breathlessness and fatigue level. The system will help medical providers assess patients based on objective and accurate data. Caregivers can also suggest necessary precautions to avoid severe symptoms.

Besides the real-time exertion monitoring and energy expenditure measurement, our system can play a long-term impact in the research of diagnosis and management of various diseases that directly affect the patient's functional capacity. Much research has been done on this topic. However, before sensor improvement, most research focused on subjectively diagnosing the issue. The recent improvement in motion sensor

technology and their integration into the smartphone and other wearable devices have encouraged researchers to work on physical activity detection and measurement to contribute to health care. The six-minute walk test integrated into our system will provide motion sensor data of patients' activity with their health status. The data can be further used in studies to assess the improvement in the health status of the patients and advise on daily physical activity requirements. The patient's daily activity and pattern history can be further studied to identify the possible risk of fatigue, malaise, and other severe symptoms that lead to chronic fatigue.

9.4 Future Works

Though we had excellent accuracy in detecting the presence of the voice in an audio stream with a relatively small dataset, we want to explore more speech and contextual data in our future work. Detecting the conversational event and the intensity of the conversation will help us assess the mental state of the patients. With the detection of voice and silence in audio, we want to calculate the duration of the conversation a user participates in. Conversational events and discussion participation also indicate how extroverted a person is, which is essential in detecting social fatigue. Our end goal is calculating energy expenditure and make patients aware of their daily physical and social exertion. We developed the application to get patient-reported confirmation of fatigue and breathlessness on a scale from 0 to 5. We plan to combine the objective measurements and subjective responses and introduce a dedicated threshold variable of daily exertion. This value will define the limit of the daily effort of an individual. Any step above the limit may be harmful to the patient. In future work, we aim to establish this threshold level of low-level detected activities and use the application's notification system to warn the patients if they are close to the threshold level and help them avoid chronic conditions.

Appendix A
Appendix

Algorithm 1 shows the segmentation algorithm for accelerometer data from the smartphone. Using the same technique, we segmented data from all other sensors.

Figure A.1 shows the mean and standard deviation from segmented windows from smartphone accelerometer and gyroscope.

Table A.1 shows the accelerometer data from smartphone for downstair activity.

We show the CNN structure of environment classification in Figure A.2. The data, analysis, and smartphone application code are uploaded to a private repository in bitbucket.

Algorithm 1 Segmentation of accelerometer data using Sliding Windows Technique

Require: x, y, z data from accelerometer

Initialize $acc\_x\_list \leftarrow []$
Initialize $acc\_y\_list \leftarrow []$
Initialize $acc\_z\_list \leftarrow []$
Initialize $window\_size \leftarrow 40, step\_size \leftarrow 20$
Initialize $features \leftarrow []$
repeat
    $window\_data \leftarrow data[i : i + window\_size]$
    $acc\_x\_list.append(window\_data['acc\_x'])$
    $acc\_y\_list.append(window\_data['acc\_y'])$
    $acc\_z\_list.append(window\_data['acc\_z'])$
    $i \leftarrow i + step\_size$
until $i > data\_length - window\_size$
$features['acc\_x\_mean'] \leftarrow mean(acc\_x\_list)$
$features['acc\_y\_mean'] \leftarrow mean(acc\_y\_list)$
$features['acc\_z\_mean'] \leftarrow mean(acc\_z\_list)$
$features['acc\_x\_std'] \leftarrow std(acc\_x\_list)$
$features['acc\_y\_std'] \leftarrow std(acc\_y\_list)$
$features['acc\_z\_std'] \leftarrow std(acc\_z\_list)$
return $features$

	acc_x_mean	acc_y_mean	acc_z_mean	gyr_x_mean	gyr_y_mean	gyr_z_mean	acc_x_std	acc_y_std	acc_z_std	gyr_x_std	gyr_y_std	gyr_z_std	labels
0	-6.674043	-3.254170	6.692817	-0.000149	0.000256	-0.000230	0.278124	0.484667	0.484174	0.001221	0.001258	0.001686	0
1	-0.234398	-8.996938	1.723647	-0.569178	0.026165	0.015576	2.024504	5.085234	2.660973	1.945862	0.841515	0.334295	4
2	-6.391364	-2.758563	7.188082	-0.000121	0.000155	0.000377	0.168945	0.295789	0.295064	0.001113	0.000962	0.001990	0
3	0.022457	-11.275578	1.464336	0.143270	-0.709319	0.032554	3.420297	4.038558	6.454860	1.629073	0.970576	0.292585	5
4	0.305723	-10.773964	2.269097	-0.215896	-0.439901	-0.071882	4.444639	4.696731	5.042735	1.878505	1.012513	0.563872	5
5	0.309316	-10.196235	2.157828	0.441844	-0.026072	0.095817	2.257909	4.636901	3.399804	1.899496	0.937060	0.386559	4
6	-6.902199	-3.650470	6.297201	0.000183	-0.000296	-0.000706	0.010195	0.013220	0.006617	0.002238	0.002071	0.002216	0
7	-1.249482	-9.926803	0.730182	0.006832	-0.000216	0.006708	0.106248	0.061505	0.169460	0.017103	0.032116	0.007251	1
8	2.087599	-9.883385	2.213163	0.171979	-0.048696	0.019937	2.707987	4.204495	4.483099	1.600852	1.160786	0.421243	5
9	-0.875368	-9.949620	0.862113	-0.005998	-0.001009	-0.001568	0.042323	0.026208	0.074206	0.005838	0.003998	0.002259	1
10	-2.273010	-10.368170	-0.132370	-0.313646	0.082976	-0.137401	3.691904	6.199822	5.937161	1.595391	0.778713	0.538893	2
11	0.239368	-10.288281	1.480686	0.185268	1.450516	-0.160192	2.717574	4.545650	4.274085	1.601888	0.646074	0.193768	4
12	0.500056	-10.302953	2.041707	0.211008	0.180575	-0.031728	3.413471	4.055614	5.427653	1.596454	1.009959	0.548947	5
13	-1.455134	-12.509968	1.482243	0.166433	0.031295	0.114879	7.678457	15.335597	13.360322	3.081788	1.891330	1.056534	3
14	0.296800	-10.546034	-0.072064	0.527724	-1.602003	0.294775	3.324941	4.354658	6.124863	1.546473	0.622028	0.670030	5
15	-0.901838	-9.961717	0.801567	-0.006251	-0.002940	0.001315	0.088021	0.041213	0.150042	0.011794	0.022706	0.006141	1
16	-0.917588	-9.942314	0.872534	-0.010372	0.005003	0.007580	0.189684	0.036651	0.135766	0.009879	0.021979	0.007596	1
17	-0.007366	-10.287503	2.113212	0.594303	0.076859	0.037161	2.081880	5.419417	4.134995	2.082624	0.904789	0.327500	4
18	-0.934716	-9.954232	0.733177	-0.012736	0.014810	0.017627	0.080976	0.029551	0.077292	0.007358	0.009101	0.004294	1
19	-2.084067	-10.328645	-0.402819	-0.219857	-0.049649	-0.113180	3.718970	5.233394	5.286391	1.647443	0.719584	0.530317	2
20	-0.005809	-11.792701	2.429714	-0.637355	0.359666	-0.084286	2.728908	4.638384	4.095661	2.029121	1.043769	0.428204	4
21	-1.237505	-12.385582	0.524530	0.143569	-0.182011	0.135944	6.145075	15.183642	11.408798	2.935529	1.621820	1.016438	3
22	-1.283617	-9.924647	0.759527	0.010807	0.002740	0.003425	0.161694	0.070160	0.162781	0.013890	0.028183	0.004813	1
23	-0.886627	-9.944410	0.876785	0.010880	-0.017993	0.001961	0.190895	0.141535	0.379222	0.089018	0.160586	0.031342	1
24	-1.931894	-10.047056	0.081666	-0.265363	0.988390	-0.012099	3.401090	5.030198	6.017145	1.712894	0.852624	0.448180	2
25	-1.434413	-12.438523	0.483208	0.044200	0.254308	0.225287	6.189586	16.478706	11.364395	3.010813	2.026834	1.224018	3
26	-1.291223	-9.922192	0.712396	-0.004333	0.021648	-0.000776	0.092197	0.048178	0.122456	0.010212	0.019285	0.009680	1
27	-2.091253	-12.410076	1.439363	-0.874233	-0.384537	-0.135552	7.227722	16.174725	14.542873	3.292781	1.687517	0.959679	3
28	0.047011	-8.976217	1.255870	0.237833	0.334379	-0.154174	2.304738	4.858559	3.848972	2.549453	0.893448	0.450023	4
29	-0.881596	-9.949620	0.872054	-0.000585	-0.003712	-0.003779	0.053386	0.027186	0.087805	0.008491	0.007242	0.003559	1

Figure A.1: Smartphone segmented data

Timestamp	X	Y	Z	Label
1667627219715	1.96740615	-7.67044258	-2.3683691	down
1667627219746	1.56496572	-7.19134665	-5.05849171	down
1667627219770	-0.77781218	-7.54108667	-2.2749455	down
1667627219770	-0.28194812	-7.89322186	1.30389929	down
1667627219806	1.96021974	-9.03347015	-3.74337387	down
1667627219806	3.17472744	-11.66849613	-8.68524647	down
1667627219837	0.3600401	-15.6928997	-9.0805006	down
1667627219864	-1.27367628	-19.90894318	-3.46070743	down
1667627219864	0.64749753	-21.92832947	-1.78147697	down
1667627219902	1.29906774	-23.40634155	-0.36095825	down
1667627219904	-3.02477098	-18.92919159	-0.51666439	down
1667627219926	-4.55548191	-7.95789957	-0.75621223	down
1667627219965	-6.13410187	-2.84115744	-3.83200669	down
1667627219966	-4.7854476	-3.81851268	-2.68457246	down
1667627220016	-1.82224095	-5.85466957	-0.18608832	down
1667627220016	-1.08682895	-5.80675983	0.0103409	down
1667627220036	-1.4988513	-4.58506584	-0.74663031	down
1667627220046	-1.94441032	-3.79934907	-0.70590723	down
1667627220066	-2.57921195	-3.37295365	-0.36335373	down
1667627220102	-2.9073925	-2.79564357	0.00315447	down
1667627220106	-2.59118938	-2.21114683	0.69544774	down
1667627220149	-1.8102634	-2.51297688	1.34462249	down
1667627220165	-0.90716803	-5.33245516	2.83221459	down
1667627220166	1.12659323	-12.76083374	3.87903881	down
1667627220217	3.93409395	-17.77457047	0.72898442	down
1667627220217	5.26358461	-15.55156708	-8.94874954	down
1667627220235	4.35809422	-13.5705061	-24.38042068	down
1667627220267	-0.852072	-9.50058842	-10.69744778	down
1667627220287	-5.14237404	-8.2118206	-3.36488819	down
1667627220288	-7.62408972	-11.6253767	-1.67368042	down
1667627220349	-6.39281368	-15.08923912	4.95460844	down
1667627220350	-3.06549406	-15.83183765	6.12120628	down
1667627220351	5.52948332	-18.26085281	-3.56610847	down
1667627220369	8.55497169	-18.74473953	-8.70201397	down
1667627220408	-0.44723615	-13.34533119	-5.02974606	down
1667627220409	-6.41437292	-9.72576332	5.02168179	down
1667627220436	-3.36013794	-9.80002308	7.9561429	down

Table A.1. Smartphone accelerometer raw data for downstair activity

Layer (type)	Output Shape	Param #
conv2d_18 (Conv2D)	(None, 11, 85, 16)	160
conv2d_19 (Conv2D)	(None, 9, 83, 16)	2320
batch_normalization_4 (BatchNormalization)	(None, 9, 83, 16)	64
module_wrapper_2 (ModuleWrapper)	(None, 9, 83, 16)	0
conv2d_20 (Conv2D)	(None, 7, 81, 32)	4640
conv2d_21 (Conv2D)	(None, 5, 79, 32)	9248
conv2d_22 (Conv2D)	(None, 3, 77, 64)	18496
conv2d_23 (Conv2D)	(None, 1, 75, 32)	18464
batch_normalization_5 (BatchNormalization)	(None, 1, 75, 32)	128
module_wrapper_3 (ModuleWrapper)	(None, 1, 75, 32)	0
global_average_pooling2d_3 (GlobalAveragePooling2D)	(None, 32)	0
dense_6 (Dense)	(None, 32)	1056
dense_7 (Dense)	(None, 17)	561

Total params: 55,137
Trainable params: 55,041
Non-trainable params: 96

Figure A.2: CNN structure for environment classification

BIBLIOGRAPHY

[1] J. A. Ricci, E. Chee, A. L. Lorandeau, and J. Berger, "Fatigue in the us workforce: prevalence and implications for lost productive work time," *Journal of occupational and environmental medicine*, pp. 1–10, 2007.

[2] "Chronic fatigue syndrome (cfs) - better health channel." `https://www.betterhealth.vic.gov.au/health/conditionsandtreatments/chronic-fatigue-syndrome-cfs`. (Accessed on 10/25/2022).

[3] E. W. Clayton, "Beyond myalgic encephalomyelitis/chronic fatigue syndrome: an iom report on redefining an illness," *Jama*, vol. 313, no. 11, pp. 1101–1102, 2015.

[4] "Report: What does covid-19 recovery actually look like? – patient led research collaborative." `https://patientresearchcovid19.com/research/report-1/`. (Accessed on 09/20/2022).

[5] M. T. Wallin, W. J. Culpepper, J. D. Campbell, L. M. Nelson, A. Langer-Gould, R. A. Marrie, G. R. Cutter, W. E. Kaye, L. Wagner, H. Tremlett, *et al.*, "The prevalence of ms in the united states: a population-based estimate using health claims data," *Neurology*, vol. 92, no. 10, pp. e1029–e1040, 2019.

[6] C. Blease, H. Carel, and K. Geraghty, "Epistemic injustice in healthcare encounters: evidence from chronic fatigue syndrome," *Journal of Medical Ethics*, vol. 43, no. 8, pp. 549–557, 2017.

[7] E. R. Unger, J.-M. S. Lin, H. Tian, B. H. Natelson, G. Lange, D. Vu, M. Blate, N. G. Klimas, E. G. Balbin, L. Bateman, *et al.*, "Multi-site clinical assessment of myalgic encephalomyelitis/chronic fatigue syndrome (mcam): design and implementation of a prospective/retrospective rolling cohort study," *American journal of epidemiology*, vol. 185, no. 8, pp. 617–626, 2017.

[8] L. Darbishire, L. Ridsdale, and P. Seed, "Distinguishing patients with chronic fatigue from those with chronic fatigue syndrome: a diagnostic study in uk primary care.," *British Journal of General Practice*, vol. 53, no. 491, pp. 441–445, 2003.

[9] A. Mahmood, P. Sridevi, and S. I. Ahamed, "Demo paper: A package of objective measurement tools for physical and social exertional activities for patients with illness-limiting capacities," in *ACM/IEEE International Conference on Connected Health: Applications,Systems and Engineering Technologies (CHASE' 22)*, 2022.

[10] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu, "Sensor-based activity recognition," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 790–808, 2012.

[11] L. M. Dang, K. Min, H. Wang, M. J. Piran, C. H. Lee, and H. Moon, "Sensor-based and vision-based human activity recognition: A comprehensive survey," *Pattern Recognition*, vol. 108, p. 107561, 2020.

[12] H. Zou, J. Yang, H. Prasanna Das, H. Liu, Y. Zhou, and C. J. Spanos, "Wifi and vision multimodal learning for accurate and robust device-free human activity recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pp. 0–0, 2019.

[13] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *Pattern recognition letters*, vol. 119, pp. 3–11, 2019.

[14] Z. Hussain, M. Sheng, and W. E. Zhang, "Different approaches for human activity recognition: A survey," *arXiv preprint arXiv:1906.05074*, 2019.

[15] R.-A. Voicu, C. Dobre, L. Bajenaru, and R.-I. Ciobanu, "Human physical activity recognition using smartphone sensors," *Sensors*, vol. 19, no. 3, p. 458, 2019.

[16] A. Subasi, A. Fllatah, K. Alzobidi, T. Brahimi, and A. Sarirete, "Smartphone-based human activity recognition using bagging and boosting," *Procedia Computer Science*, vol. 163, pp. 54–61, 2019. 16th Learning and Technology Conference 2019Artificial Intelligence and Machine Learning: Embedding the Intelligence.

[17] B. Cates, T. Sim, H. M. Heo, B. Kim, H. Kim, and J. H. Mun, "A novel detection model and its optimal features to classify falls from low-and high-acceleration activities of daily life using an insole sensor system," *Sensors*, vol. 18, no. 4, p. 1227, 2018.

[18] F. Kawsar, M. K. Hasan, R. Love, and S. I. Ahamed, "A novel activity detection system using plantar pressure sensors and smartphone," in *2015 IEEE 39th Annual Computer Software and Applications Conference*, vol. 1, pp. 44–49, IEEE, 2015.

[19] A. J. A. Majumder, S. I. Ahamed, R. J. Povinelli, C. P. Tamma, and R. O. Smith, "A novel wireless system to monitor gait using smartshoe-worn sensors," in *2015 IEEE 39th Annual Computer Software and Applications Conference*, vol. 2, pp. 733–741, IEEE, 2015.

[20] L. Roland, L. Lidauer, G. Sattlecker, F. Kickinger, W. Auer, V. Sturm, D. Efrosinin, M. Drillich, and M. Iwersen, "Monitoring drinking behavior in bucket-fed dairy calves using an ear-attached tri-axial accelerometer: A pilot study," *Computers and Electronics in Agriculture*, vol. 145, pp. 298–301, 2018.

[21] J. Qi, P. Yang, A. Waraich, Z. Deng, Y. Zhao, and Y. Yang, "Examining sensor-based physical activity recognition and monitoring for healthcare using internet of things: A systematic review," *Journal of biomedical informatics*, vol. 87, pp. 138–153, 2018.

[22] W. Wang and C. Miao, "Activity recognition in new smart home environments," in *Proceedings of the 3rd International Workshop on Multimedia for Personal Health and Health Care*, pp. 29–37, 2018.

[23] M. Gochoo, T.-H. Tan, S.-H. Liu, F.-R. Jean, F. S. Alnajjar, and S.-C. Huang, "Unobtrusive activity recognition of elderly people living alone using anonymous binary sensors and dcnn," *IEEE journal of biomedical and health informatics*, vol. 23, no. 2, pp. 693–702, 2018.

[24] M. K. O'Brien, N. Shawen, C. K. Mummidisetty, S. Kaur, X. Bo, C. Poellabauer, K. Kording, and A. Jayaraman, "Activity recognition for persons with stroke using mobile phone technology: toward improved performance in a home setting," *Journal of medical Internet research*, vol. 19, no. 5, p. e7385, 2017.

[25] H. Wang, J. Zhao, J. Li, L. Tian, P. Tu, T. Cao, Y. An, K. Wang, and S. Li, "Wearable sensor-based human activity recognition using hybrid deep learning techniques," *Security and communication Networks*, vol. 2020, 2020.

[26] M. Altini, J. Penders, R. Vullers, and O. Amft, "Estimating energy expenditure using body-worn accelerometers: a comparison of methods, sensors number and positioning," *IEEE journal of biomedical and health informatics*, vol. 19, no. 1, pp. 219–226, 2014.

[27] D. Figo, P. C. Diniz, D. R. Ferreira, and J. M. Cardoso, "Preprocessing techniques for context recognition from accelerometer data," *Personal and Ubiquitous Computing*, vol. 14, no. 7, pp. 645–662, 2010.

[28] E. J. Huang, K. Yan, and J.-P. Onnela, "Smartphone-based activity recognition using multistream movelets combining accelerometer and gyroscope data," *Sensors*, vol. 22, no. 7, p. 2618, 2022.

[29] Z. Chen, Q. Zhu, Y. C. Soh, and L. Zhang, "Robust human activity recognition using smartphone sensors via ct-pca and online svm," *IEEE transactions on industrial informatics*, vol. 13, no. 6, pp. 3070–3080, 2017.

[30] G. Filios, S. Nikoletseas, C. Pavlopoulou, M. Rapti, and S. Ziegler, "Hierarchical algorithm for daily activity recognition via smartphone sensors," in *2015 Ieee 2nd World Forum on Internet of Things (Wf-Iot)*, pp. 381–386, IEEE, 2015.

[31] O. Dehzangi, B. A. Bache, and O. Iftikhar, "Activity detection using fusion of multi-pressure sensors in insoles," in *2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 3315–3321, IEEE, 2018.

[32] F. Shahmohammadi, A. Hosseini, C. E. King, and M. Sarrafzadeh, "Smartwatch based activity recognition using active learning," in *2017 IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE)*, pp. 321–329, IEEE, 2017.

[33] S. Mekruksavanich and A. Jitpattanakul, "Smartwatch-based human activity recognition using hybrid lstm network," in *2020 IEEE SENSORS*, pp. 1–4, IEEE, 2020.

[34] G. M. Weiss, J. L. Timko, C. M. Gallagher, K. Yoneda, and A. J. Schreiber, "Smartwatch-based activity recognition: A machine learning approach," in *2016 IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI)*, pp. 426–429, IEEE, 2016.

[35] Y. Dong, J. Scisco, M. Wilson, E. Muth, and A. Hoover, "Detecting periods of eating during free-living by tracking wrist motion," *IEEE journal of biomedical and health informatics*, vol. 18, no. 4, pp. 1253–1260, 2013.

[36] G. Laput and C. Harrison, "Sensing fine-grained hand activity with smartwatches," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–13, 2019.

[37] H. Li, S. Chawla, R. Li, S. Jain, G. D. Abowd, T. Starner, C. Zhang, and T. Plötz, "Wristwash: towards automatic handwashing assessment using a wrist-worn device," in *Proceedings of the 2018 ACM international symposium on wearable computers*, pp. 132–139, 2018.

[38] R. I. Ramos-Garcia and A. W. Hoover, "A study of temporal action sequencing during consumption of a meal," in *Proceedings of the International Conference on Bioinformatics, Computational Biology and Biomedical Informatics*, BCB'13, (New York, NY, USA), p. 68–75, Association for Computing Machinery, 2013.

[39] H. Han and S. W. Yoon, "Gyroscope-based continuous human hand gesture recognition for multi-modal wearable input device for human machine interaction," *Sensors*, vol. 19, no. 11, p. 2562, 2019.

[40] L. M. S. Morillo, L. Gonzalez-Abril, J. A. O. Ramirez, and M. A. A. de la Concepcion, "Low energy physical activity recognition system on smartphones," *Sensors*, vol. 15, no. 3, pp. 5163–5196, 2015.

[41] J. Pansiot, D. Stoyanov, D. McIlwraith, B. P. Lo, and G.-Z. Yang, "Ambient and wearable sensor fusion for activity recognition in healthcare monitoring systems," in *4th international workshop on wearable and implantable body sensor networks (BSN 2007)*, pp. 208–212, Springer, 2007.

[42] D. Yazdansepas, A. H. Niazi, J. L. Gay, F. W. Maier, L. Ramaswamy, K. Rasheed, and M. P. Buman, "A multi-featured approach for wearable sensor-based human activity recognition," in *2016 IEEE International Conference on Healthcare Informatics (ICHI)*, pp. 423–431, 2016.

[43] H. F. Nweke, Y. W. Teh, U. R. Alo, and G. Mujtaba, "Analysis of multi-sensor fusion for mobile and wearable sensor based human activity recognition," in *Proceedings of the international conference on data processing and applications*, pp. 22–26, 2018.

[44] C. Zhu and W. Sheng, "Multi-sensor fusion for human daily activity recognition in robot-assisted living," in *2009 4th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 303–304, 2009.

[45] S. Chernbumroong, A. S. Atkins, and H. Yu, "Activity classification using a single wrist-worn accelerometer," in *2011 5th International Conference on Software, Knowledge Information, Industrial Management and Applications (SKIMA) Proceedings*, pp. 1–6, IEEE, 2011.

[46] F. G. Da Silva and E. Galeazzo, "Accelerometer based intelligent system for human movement recognition," in *5th IEEE International Workshop on Advances in Sensors and Interfaces IWASI*, pp. 20–24, IEEE, 2013.

[47] M. Shoaib, S. Bosch, O. D. Incel, H. Scholten, and P. J. Havinga, "Complex human activity recognition using smartphone and wrist-worn motion sensors," *Sensors*, vol. 16, no. 4, p. 426, 2016.

[48] A. Parate, M.-C. Chiu, C. Chadowitz, D. Ganesan, and E. Kalogerakis, "Risq: Recognizing smoking gestures with inertial sensors on a wristband," in *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '14, (New York, NY, USA), p. 149–161, Association for Computing Machinery, 2014.

[49] G. Laput, R. Xiao, and C. Harrison, "Viband: High-fidelity bio-acoustic sensing using commodity smartwatch accelerometers," in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pp. 321–333, 2016.

[50] A. Moschetti, L. Fiorini, D. Esposito, P. Dario, and F. Cavallo, "Recognition of daily gestures with wearable inertial rings and bracelets," *Sensors*, vol. 16, no. 8, p. 1341, 2016.

[51] A. M. Khan, Y.-K. Lee, S. Y. Lee, and T.-S. Kim, "A triaxial accelerometer-based physical-activity recognition via augmented-signal features and a hierarchical recognizer," *IEEE transactions on information technology in biomedicine*, vol. 14, no. 5, pp. 1166–1172, 2010.

[52] E. K. Antonsson and R. W. Mann, "The frequency content of gait," *Journal of Biomechanics*, vol. 18, no. 1, pp. 39–47, 1985.

[53] U. Maurer, A. Smailagic, D. Siewiorek, and M. Deisher, "Activity recognition and monitoring using multiple sensors on different body positions," in *International Workshop on Wearable and Implantable Body Sensor Networks (BSN'06)*, pp. 4 pp.–116, 2006.

[54] Y. Hanai, J. Nishimura, and T. Kuroda, "Haar-like filtering for human activity recognition using 3d accelerometer," in *2009 IEEE 13th Digital Signal Processing Workshop and 5th IEEE Signal Processing Education Workshop*, pp. 675–678, 2009.

[55] S. Leikas and V.-J. Ilmarinen, "Happy now, tired later? extraverted and conscientious behavior are related to immediate mood gains, but to later fatigue," *Journal of Personality*, vol. 85, no. 5, pp. 603–615, 2017.

[56] D. Wyatt, T. Choudhury, J. Bilmes, and J. A. Kitts, "Inferring colocation and conversation networks from privacy-sensitive audio with implications for computational social science," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 1, pp. 1–41, 2011.

[57] H. Lu, D. Frauendorfer, M. Rabbi, M. S. Mast, G. T. Chittaranjan, A. T. Campbell, D. Gatica-Perez, and T. Choudhury, "Stresssense: Detecting stress in unconstrained acoustic environments using smartphones," in *Proceedings of the 2012 ACM conference on ubiquitous computing*, pp. 351–360, 2012.

[58] S. Abdullah, M. Matthews, E. Frank, G. Doherty, G. Gay, and T. Choudhury, "Automatic detection of social rhythms in bipolar disorder," *Journal of the American Medical Informatics Association*, vol. 23, no. 3, pp. 538–543, 2016.

[59] H. Dubey, J. C. Goldberg, M. Abtahi, L. Mahler, and K. Mankodiya, "Echowear: smartwatch technology for voice and speech treatments of patients with parkinson's disease," in *Proceedings of the conference on Wireless Health*, pp. 1–8, 2015.

[60] H. Dubey, J. C. Goldberg, K. Mankodiya, and L. Mahler, "A multi-smartwatch system for assessing speech characteristics of people with dysarthria in group settings," in *2015 17th International Conference on E-health Networking, Application Services (HealthCom)*, pp. 528–533, 2015.

[61] A. Monteiro, H. Dubey, L. Mahler, Q. Yang, and K. Mankodiya, "Fit: A fog computing device for speech tele-treatments," in *2016 IEEE international conference on smart computing (SMARTCOMP)*, pp. 1–3, IEEE, 2016.

[62] Y. Komatsu, K. Shimojo, Y. Nishiyama, and K. Sezaki, "Toward measuring conversation duration using a wristwatch-type wearable device," in *2022 IEEE International Conference on Smart Computing (SMARTCOMP)*, pp. 150–152, IEEE, 2022.

[63] J. J. Reilly, V. Penpraze, J. Hislop, G. Davies, S. Grant, and J. Y. Paton, "Objective measurement of physical activity and sedentary behaviour: review with new data," *Archives of disease in childhood*, vol. 93, no. 7, pp. 614–619, 2008.

[64] K. B. Adamo, S. A. Prince, A. C. Tricco, S. Connor-Gorber, and M. Tremblay, "A comparison of indirect versus direct measures for assessing physical activity in the pediatric population: a systematic review," *International Journal of Pediatric Obesity*, vol. 4, no. 1, pp. 2–27, 2009.

[65] D. Arvidsson, M. Leijon, J. Sundquist, K. Sundquist, U. Lindblad, and L. Bennet, "Cross-cultural validation of a simple self-report instrument of physical activity in immigrants from the middle east and native swedes," *Scandinavian journal of public health*, vol. 42, no. 3, pp. 255–262, 2014.

[66] J. M. Tucker, G. J. Welk, and N. K. Beyler, "Physical activity in us adults: compliance with the physical activity guidelines for americans," *American journal of preventive medicine*, vol. 40, no. 4, pp. 454–461, 2011.

[67] T. Palombo, A. Campos, S. D. Vernon, and S. Roundy, "Accurate and objective determination of myalgic encephalomyelitis/chronic fatigue syndrome disease severity with a wearable sensor," *Journal of Translational Medicine*, vol. 18, no. 1, pp. 1–9, 2020.

[68] H. Luo, P.-A. Lee, I. Clay, M. Jaggi, and V. De Luca, "Assessment of fatigue using wearable sensors: a pilot study," *Digital biomarkers*, vol. 4, no. 1, pp. 59–72, 2020.

[69] B. Dong, S. Biswas, A. Montoye, and K. Pfeiffer, "Comparing metabolic energy expenditure estimation using wearable multi-sensor network and single accelerometer," in *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 2866–2869, IEEE, 2013.

[70] Z. S. Maman, Y.-J. Chen, A. Baghdadi, S. Lombardo, L. A. Cavuoto, and F. M. Megahed, "A data analytic framework for physical fatigue management using wearable sensors," *Expert Systems with Applications*, vol. 155, p. 113405, 2020.

[71] A. Mehmood, A. Nadeem, M. Ashraf, M. S. Siddiqui, K. Rizwan, and K. Ahsan, "A fall risk assessment mechanism for elderly people through muscle fatigue analysis on data from body area sensor network," *IEEE Sensors Journal*, vol. 21, no. 5, pp. 6679–6690, 2020.

[72] H. Besson, S. Brage, R. W. Jakes, U. Ekelund, and N. J. Wareham, "Estimating physical activity energy expenditure, sedentary time, and physical activity intensity by self-report in adults," *The American journal of clinical nutrition*, vol. 91, no. 1, pp. 106–114, 2010.

[73] P. Alinia, R. Saeedi, B. Mortazavi, A. Rokni, and H. Ghasemzadeh, "Impact of sensor misplacement on estimating metabolic equivalent of task with wearables," in *2015 IEEE 12th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*, pp. 1–6, IEEE, 2015.

[74] B. Mortazavi, N. Alsharufa, S. I. Lee, M. Lan, M. Sarrafzadeh, M. Chronley, and C. K. Roberts, "Met calculations from on-body accelerometers for exergaming movements," in *2013 IEEE International Conference on Body Sensor Networks*, pp. 1–6, IEEE, 2013.

[75] "Measuring physical activity - monitoring & promotion of physical activity." `http://getmovingvcepe2015.weebly.com/measuring-physical-activity.html`. (Accessed on 09/15/2022).

[76] J. Lee, J. Lim, and J. Lee, "Compensated heading angles for outdoor mobile robots in magnetically disturbed environment," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 2, pp. 1408–1419, 2017.

[77] F. Wittmann, O. Lambercy, and R. Gassert, "Magnetometer-based drift correction during rest in imu arm motion tracking," *Sensors*, vol. 19, no. 6, p. 1312, 2019.

[78] A. A. Aguileta, R. F. Brena, O. Mayora, E. Molino-Minero-Re, and L. A. Trejo, "Multi-sensor fusion for activity recognition—a survey," *Sensors*, vol. 19, no. 17, p. 3808, 2019.

[79] M. Hynes, H. Wang, and L. Kilmartin, "Off-the-shelf mobile handset environments for deploying accelerometer based gait and activity analysis algorithms," in *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 5187–5190, IEEE, 2009.

[80] K. Lyden, S. L. Kozey-Keadle, J. W. Staudenmayer, and P. S. Freedson, "Validity of two wearable monitors to estimate breaks from sedentary time," *Medicine and science in sports and exercise*, vol. 44, no. 11, p. 2243, 2012.

[81] M. Mathie, *Monitoring and interpreting human movement patterns using a triaxial accelerometer*. University of New South Wales Sydney, 2003.

[82] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *ACM SigKDD Explorations Newsletter*, vol. 12, no. 2, pp. 74–82, 2011.

[83] C. Randell and H. Muller, "Context awareness by analysing accelerometer data," in *Digest of Papers. Fourth International Symposium on Wearable Computers*, pp. 175–176, IEEE, 2000.

[84] J. Farringdon, A. J. Moore, N. Tilbury, J. Church, and P. D. Biemond, "Wearable sensor badge and sensor jacket for context awareness," in *Digest of Papers. Third International Symposium on Wearable Computers*, pp. 107–113, IEEE, 1999.

[85] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," in *International conference on pervasive computing*, pp. 1–17, Springer, 2004.

[86] Y. Kawahara, H. Kurasawa, and H. Morikawa, "Recognizing user context using mobile handsets with acceleration sensors," in *2007 IEEE International Conference on Portable Information Devices*, pp. 1–5, Ieee, 2007.

[87] J. Yang, "Toward physical activity diary: motion recognition using simple acceleration features with mobile phones," in *Proceedings of the 1st international workshop on Interactive multimedia for consumer electronics*, pp. 1–10, 2009.

[88] A. Khan, N. Hammerla, S. Mellor, and T. Plötz, "Optimising sampling rates for accelerometer-based human activity recognition," *Pattern Recognition Letters*, vol. 73, pp. 33–40, 2016.

[89] Y.-J. Hong, I.-J. Kim, S. C. Ahn, and H.-G. Kim, "Mobile health monitoring system based on activity recognition using accelerometer," *Simulation Modelling Practice and Theory*, vol. 18, no. 4, pp. 446–455, 2010.

[90] B. H. Dobkin, X. Xu, M. Batalin, S. Thomas, and W. Kaiser, "Reliability and validity of bilateral ankle accelerometer algorithms for activity recognition and walking speed after stroke," *Stroke*, vol. 42, no. 8, pp. 2246–2250, 2011.

[91] M. S. Aung, S. B. Thies, L. P. Kenney, D. Howard, R. W. Selles, A. H. Findlow, and J. Y. Goulermas, "Automated detection of instantaneous gait events using time frequency analysis and manifold embedding," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 21, no. 6, pp. 908–916, 2013.

[92] K. Altun and B. Barshan, "Human activity recognition using inertial/magnetic sensor units," in *International workshop on human behavior understanding*, pp. 38–51, Springer, 2010.

[93] Y. Nam and J. W. Park, "Physical activity recognition using a single triaxial accelerometer and a barometric sensor for baby and child care in a home environment," *Journal of Ambient Intelligence and Smart Environments*, vol. 5, no. 4, pp. 381–402, 2013.

[94] N. Nurwulan and B. C. Jiang, "Window selection impact in human activity recognition," *International Journal of Innovative Technology and Interdisciplinary Sciences*, vol. 3, no. 1, pp. 381–394, 2020.

[95] G. Wang, Q. Li, L. Wang, W. Wang, M. Wu, and T. Liu, "Impact of sliding window length in indoor human motion modes and pose pattern recognition based on smartphone sensors," *Sensors*, vol. 18, no. 6, p. 1965, 2018.

[96] P. Nurmi and P. Floréen, "Online feature selection for contextual time series data," in *Subspace, Latent Structure and Feature Selection techniques: Statistical and Optimisation perspectives Workshop (SLSFS05)*, Citeseer, 2005.

[97] J. Suto, S. Oniga, and P. P. Sitar, "Comparison of wrapper and filter feature selection algorithms on human activity recognition," in *2016 6th international conference on computers communications and control (ICCCC)*, pp. 124–129, IEEE, 2016.

[98] A. Loddo, B. Pes, and D. Riboni, "Feature selection in mobile activity recognition: A comparative study," in *2021 22nd IEEE International Conference on Mobile Data Management (MDM)*, pp. 181–186, 2021.

[99] G. Bieber and C. Peter, "Using physical activity for user behavior analysis," in *Proceedings of the 1st international conference on PErvasive Technologies Related to Assistive Environments*, pp. 1–6, 2008.

[100] J. Mantyjarvi, J. Himberg, and T. Seppanen, "Recognizing human motion with multiple acceleration sensors," in *2001 ieee international conference on systems, man and cybernetics. e-systems and e-man for cybernetics in cyberspace (cat. no. 01ch37236)*, vol. 2, pp. 747–752, IEEE, 2001.

[101] H. Ghasemzadeh, V. Loseu, E. Guenterberg, and R. Jafari, "Sport training using body sensor networks: A statistical approach to measure wrist rotation for golf swing," in *Proceedings of the Fourth International Conference on Body Area Networks*, pp. 1–8, 2009.

[102] J. A. Ward, P. Lukowicz, and G. Tröster, "Gesture spotting using wrist worn microphone and 3-axis accelerometer," in *Proceedings of the 2005 joint conference on Smart objects and ambient intelligence: innovative context-aware services: usages and technologies*, pp. 99–104, 2005.

[103] A. Gupta, K. Gupta, K. Gupta, and K. Gupta, "A survey on human activity recognition and classification," in *2020 international conference on communication and signal processing (ICCSP)*, pp. 0915–0919, IEEE, 2020.

[104] S. G. Trost, Y. Zheng, and W.-K. Wong, "Machine learning for activity recognition: hip versus wrist data," *Physiological measurement*, vol. 35, no. 11, p. 2183, 2014.

[105] "Mmc – metamotionc – mbientlab." `https://mbientlab.com/store/metamotionc/`. (Accessed on 09/24/2022).

[106] S. Balli, E. A. Sağbaş, and M. Peker, "Human activity recognition from smart watch sensor data using a hybrid of principal component analysis and random forest algorithm," *Measurement and Control*, vol. 52, no. 1-2, pp. 37–45, 2019.

[107] H. Zhang, M. Alrifaai, K. Zhou, and H. Hu, "A novel fuzzy logic algorithm for accurate fall detection of smart wristband," *Transactions of the Institute of Measurement and Control*, vol. 42, no. 4, pp. 786–794, 2020.

[108] J. C. Y. Wong, J. Wang, E. Y. Fu, H. V. Leong, and G. Ngai, "Activity recognition and stress detection via wristband," in *Proceedings of the 17th International Conference on Advances in Mobile Computing & Multimedia*, pp. 102–106, 2019.

[109] M. Cescon, D. Choudhary, J. E. Pinsker, V. Dadlani, M. M. Church, Y. C. Kudva, F. J. Doyle III, and E. Dassau, "Activity detection and classification from wristband accelerometer data collected on people with type 1 diabetes in free-living conditions," *Computers in Biology and Medicine*, vol. 135, p. 104633, 2021.

[110] R. K. Jazani, M. Saremi, T. Rezapour, A. Kavousi, and H. Shirzad, "Influence of traffic-related noise and air pollution on self-reported fatigue," *International journal of occupational safety and ergonomics*, vol. 21, no. 2, pp. 193–200, 2015.

[111] "Github - gkonovalov/android-vad: This vad library can process audio in real-time utilizing gmm which helps identify presence of human speech in an audio sample that contains a mixture of speech and noise.." `https://github.com/gkonovalov/android-vad`. (Accessed on 09/29/2022).

[112] M. Cengiz and İ. A. Kubilay, "Filtering and analysing human voice using digital filters,"

[113] R. Bachu, S. Kopparthi, B. Adapa, and B. Barkana, "Separation of voiced and unvoiced using zero crossing rate and energy of the speech signal," in *American Society for Engineering Education (ASEE) zone conference proceedings*, pp. 1–7, American Society for Engineering Education, 2008.

[114] Y. Hu and P. C. Loizou, "Subjective comparison and evaluation of speech enhancement algorithms," *Speech communication*, vol. 49, no. 7-8, pp. 588–601, 2007.

[115] K. J. Piczak, "Esc: Dataset for environmental sound classification," in *Proceedings of the 23rd ACM international conference on Multimedia*, pp. 1015–1018, 2015.

[116] T. Sainburg, "timsainb/noisereduce: v1.0," June 2019.

[117] "Sound intensity and sound level | physics." `https://courses.lumenlearning.com/atd-austincc-physics1/chapter/17-3-sound-intensity-and-sound-level/`. (Accessed on 09/29/2022).

[118] "What noises cause hearing loss?." `https://www.cdc.gov/nceh/hearing\_\\loss/what\_noises\_cause\_hearing\_loss.html`. (Accessed on 09/29/2022).

[119] "Sound meter - apps on google play." `https://play.google.com/store/apps/details?id=com.splendapps.decibel`. (Accessed on 09/29/2022).

[120] S. R. Livingstone and F. A. Russo, "The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english," *PloS one*, vol. 13, no. 5, p. e0196391, 2018.

[121] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, vol. 8, pp. 18–25, 2015.

[122] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights into imaging*, vol. 9, no. 4, pp. 611–629, 2018.

[123] T. Münzel, M. Sørensen, F. Schmidt, E. Schmidt, S. Steven, S. Kröller-Schön, and A. Daiber, "The adverse effects of environmental noise exposure on oxidative stress and cardiovascular risk," *Antioxidants & redox signaling*, vol. 28, no. 9, pp. 873–908, 2018.

[124] Y. Kawahara, N. Ryu, and T. Asami, "Monitoring daily energy expenditure using a 3-axis accelerometer with a low-power microprocessor," *International Journal on Human-Computer Interaction*, vol. 1, no. 5, pp. 145–154, 2009.

[125] P. A. G. A. Committee *et al.*, "Physical activity guidelines advisory committee report, 2008," *Washington, DC: US Department of Health and Human Services*, vol. 2008, pp. A1–H14, 2008.

[126] A. C. on Proficiency Standards for Clinical Pulmonary Function Laboratories *et al.*, "Ats statement: guidelines for the six-minute walk test," *Am J Respir Crit Care Med*, vol. 166, pp. 111–117, 2002.

[127] I. Bautmans, M. Lambert, and T. Mets, "The six-minute walk test in community dwelling elderly: influence of health status.," *BMC geriatrics*, vol. 4, no. 1, pp. 1–9, 2004.