Sensor Fusion and Data Processing to Analyse Human Gait and Activities for Healthcare Applications

2021

A thesis submitted to The University of Manchester

for the degree of Doctor of Philosophy (PhD)

in the Faculty of Science & Engineering

Syed Usama Yunas

School of Electrical and Electronic Engineering

List of contents

List of figures

List of tables

List of publications

Journal Papers:

- **Syed Usama Yunas** and Krikor Ozanyan. *Gait Activity Classification using Multi-Modality Sensor Fusion: A Deep Learning Approach*. IEEE Sensors Journal, 3rd May. 2021. *DOI: 10.1109/JSEN.2021.3077698*

- **Syed Usama Yunas** and Krikor Ozanyan. *Gait Activity Classification from Feature-Level Sensor Fusion of Multi-Modality Systems*. IEEE Sensors Journal, 5th Oct. 2020. *DOI: 10.1109/JSEN.2020.3028697*

- Abdullah Alharthi, **Syed Usama Yunas** and Krikor Ozanyan. *Deep Learning for Monitoring of Human Gait: A Review*. IEEE Sensors Journal, 15th Jul. 2019. *DOI: 10.1109/JSEN.2019.2928777*

Conference Papers:

- **Syed Usama Yunas**, Abdullah Alharthi and Krikor Ozanyan. *Multi-modality sensor fusion for gait classification using deep learning*. 2020 IEEE Sensors Applications Symposium (SAS), 9-11th Mar. 2020. *DOI: 10.1109/SAS48726.2020.9220037* ***(Winner Best Paper Award)***

- **Syed Usama Yunas**, Abdullah Alharthi and Krikor Ozanyan. *Multi-modality fusion of floor and ambulatory sensors for gait classification*. 2019 IEEE 28th International Symposium on Industrial Electronics (ISIE). 1st Aug, 2019. *DOI: 10.1109/ISIE.2019.8781127*

- Abdullah Alharthi, **Syed Usama Yunas** and Krikor Ozanyan. *Sensor fusion for analysis of gait under cognitive load: deep learning approach*. 2020 IEEE Sensors Applications Symposium (SAS), 12 Oct. 2020. *DOI: 10.1109/SAS48726.2020.9220046*

Book Chapter:

- Omar Costilla Reyes, Ruben Vera-Rodriguez, Abdullah Alharthi, **Syed Usama Yunas** and Krikor Ozanyan. *Deep learning in gait analysis for security and healthcare*. Deep learning: Algorithms and Applications. Pedrycz, W. & Chen, SM. (eds.). Cham: Springer Nature, Vol. 865. p. 299 334 p. (Studies in computational intelligence; vol. 865), 24th Oct. 2019. *DOI: 10.1007/978-3-030-31760-7_10*

Abstract

The distinctive features of human gait are distributed across modalities based on vision, inertial measurements, pressure, and sound. Gait features pertaining to a single modality have different scales and intensities. Single modality systems suffer from misclassification due to the unavailability of complementary features that provide the semantic information involved in gait activity. We aim to adequately map the complete gait features which is not possible using a simple and feasible modality. In this research work, multi-modality sensor fusion approach has been adapted which is capable to extract and fuse information from two sources and provides maximum description of individual's gait.

Feature level-based sensor fusion is proposed for the spatio-temporal data obtained from 3 inertial sensors based Ambulatory Inertial Sensors (AIS) placed at pelvis and both heels of user and a set of 116 collaborative Floor Sensors (FS), which is novel. The complimentary nature and relationships among datasets of the associated spatio-temporal features are explored using Principal Component Analysis (PCA) and Canonical Correlation Analysis (CCA) techniques. Supremacy of the proposed approach is tested using different machine learning (ML) algorithms. With K-Nearest Neighbour (K-NN) and Kernel Support Vector Machine (K-SVM), our multi-modal sensor fusion approach demonstrates improved f-scores of 95% and 94% respectively, beyond the individual f-scores.

Furthermore, deep learning (DL) models will be utilized to perform automatic feature extraction of the ground reaction force and lower body movements using FS and AIS, simultaneously. Benefits of implementing DL models are twofold: First, the spatio-temporal information from the two modalities are balanced despite disproportionate number of inputs. Second the extracted information is fused over DL model layers whilst reserving the categorical content of each gait activity. This proposed fused approach is further assessed with f-scores using various DL models i.e., LSTM (99.90%), 2D-CNN (88.73%), 1D-CNN (94.97%) and FFNN (89.33%). It is concluded that using given DL models, robustness and execution time are the tradeoff while observing the overall performance of proposed system.

Declaration

No portion of the work referred to in the thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright statement

Acknowledgements

I would like to thank the Almighty Allah who has given me so many chances to live my personality the way I wished for. I have been through hard and even worst, but He never left me alone. During my PhD, I had a lot of chance to think and act. At the end of this PhD, I feel the responsibility to share my knowledge and skills for the betterment of humankind.

I am thankful to my parents and my family who have supported me and prayed for my success throughout my PhD. I am very lucky to have Dr. Krikor Ozanyan as my supervisor and a mentor. He is a very kind human being and shares his valuable experiences which are helpful to solve complex situations. During my PhD, he guided me in steps where he found me lost and pointed things in right direction. I am thankful for his priceless moral and supervisory contributions in my life. I would also like to acknowledge the help and support provided by my co-supervisor Dr. Patricia Scully and advisor Dr. David Foster.

Chapter 1

Introduction

1.1 Gait and its applications

Gait defines the movement of center of gravity of human body during locomotion. Gait can be interpreted as a unique walking sequence that gets effected by multiple independent elements such as age, gender, weight, and height etc. People can be identified and monitored on the basis of their walking behaviours using gait recognition techniques [1]. Walking behaviour in humans forms a unique biometric method just like other biometric modalities i.e., face, fingerprints, iris detection etc.

Many research areas have implementations and wider research in the field of gait analysis. Monitoring of human gait and daily life activities adds an essential medical feature to the health monitor systems which could improve life's quality, prescribe personalised treatments, inform doctors about the health of patient, minimize the health costs and ensure quick response to medical emergencies. Gait is also researched to monitor and examine the pathological conditions such as Parkinson's disease, Alzheimer and other neuromuscular disorders [2]. These days, modern gait labs have been established in many orthopaedic hospitals for routine treatment plans and the follow-up procedures.

In security applications, gait analysis has been exploited in many applications varying from personal access to border control systems [3]. Moreover, gait analysis has implications in sports where athletes are observed and monitored after returning from injuries caused during sports activities [4].

1.2 Aim of research

Floor sensors (FS) are unobtrusive and commonly used to collect footprint information in which body pressure on the feet is used to determine the type of gait activity. Stance and swing phases comprise approximately 60% and 40% of a complete gait cycle respectively. However, there are certain situations in which the range of feet motion during swing phase encloses important gait cycle information. Focusing on the healthcare scenarios for patients having abnormal gait i.e., Hemiplegic gait in which patient will have to circumduct or swing one leg around to step forward. Similarly, Neuropathic gait results in high stepping gait to avoid dragging of toe on ground. Irregular, jerky and involuntary movements in both upper and lower extremities are caused during Choreiform gait. In Myopathic gait, a waddle type walking pattern results due to the inability of patients to stabilize the pelvis as they lift their leg to step forward and cause pelvis to tilt towards the non-weight carrying leg.

Therefore, a complete gait information includes interaction of feet with and above the ground and hence requires additional sensor installation to be installed on lower body parts of the user to capture the intermediate information between ground contacts. In this research work, we aim at acquiring gait information from the lower parts of human body mostly related to healthcare scenarios, e.g. age related factors [5],[6] and cognitive tasks [7],[8].

1.3 Motivation of multi-modality sensor fusion

Different sensing modalities have developed distinct set of features based on bio-mechanical measures related to physical body dimensions, body part masses and time varying forces generated by muscles during the gait cycle. Advances in gait sensing instruments have resulted in the evaluation of many human locomotion characteristics obtained from high quality information. However, the feasibility of a simple and widely used modality to adequately map the complex gait features is still unclear.

Multi-modality sensor fusion results in producing new data representations which are unique to the collection of individual sensors and sensing modalities [9]. Our motivation is to capture the complex nature of gait information using a multisource and multi-modality sensor fusion approach. The research hypothesis for this work is as follows:

'The fusion of gait information (during and between the ground contacts) acquired from multi-modality systems would improve the classification accuracies of an individual's gait activity when compared to a single modality system'.

1.4 Objective of research

Gait feature fusion has been immensely used to study human gait parameters and anomalies associated with motion generated during gait cycle. The objective of this research is to propose a **novel** approach to fuse gait activity information, at feature level, using two different gait sensor systems. To achieve this, an ambulatory inertial sensor (AIS) sub-system is designed and operated to allow signal acquisition protocols suitable for processing and fusion with those from an existing original floor sensor sub-system [10] as illustrated in figure 1.1.

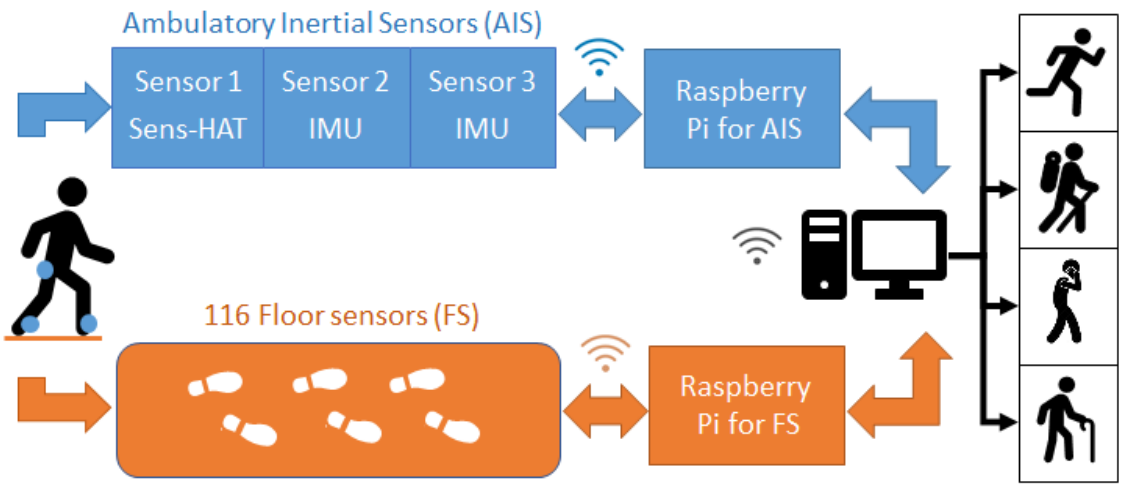


Figure 1.1: Overall diagram of the proposed multi-modality sensor fusion system [133]

Both systems are used to simultaneously capture information from healthy volunteers for multiple cognitive load-based gait activities. We have investigated the individual gait modalities and analysed the complementarity between both modalities gait data at feature level, at the same time allowing closer observation on how the choice of machine learning (ML) models affects the outcome of feature-level fusion. The data obtained from the separate modalities is processed and classified using supervised ML models such as Logistic Regression (LR), K-Nearest Neighbour (K-NN), Naïve Bayes (NB), Linear Support Vector Machine (SVM), Kernel Support Vector Machine (K-SVM), Decision Trees (DT) and

Random Forest (RF). ML models mainly rely on handcrafted features or feature extraction techniques for the classification of gait related activities. Feature domains containing various features increase the chances of redundancy and irrelevancy.

Therefore, deep learning (DL) is called upon to maximize the use of data variance and removes the dependencies on handcrafted features from individual whilst exploring the effectiveness of the combined information from a discriminant angle. Deep learning (DL) is inspired by the biological neural networks and function like human brain. Multi-layered artificial neural network (ANN) is built to learn data representations automatically which make them a choice whilst dealing with high volume datasets. The benefit of utilizing DL models is twofold: First, with minimal pre-processing on complex data, DL models can achieve robust and improved accuracies. Second, the automatic extraction of features from data using DL models layer also reduces the chances of redundancy and leads to substantially robust and accurate results as compared to the other ML techniques. However, accuracy and performance of these systems is highly debated and there is significant amount of work for improving the quality of data from the gait sensors. We will utilize DL models, such as Feed-Forward Neural Network (FFNN), Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) models to automatically extract and fuse rich representations of various gait activity patterns for robust and efficient classification.

1.5 Structure of the thesis

This section describes each chapter included in this thesis.

Chapter 1 introduces the various implementations of human gait analysis and developments in sensor technologies in this field of research. It highlights the need of sensor information fusion within and across the modalities. It also presents the motivation and approach of our multi-modality sensor fusion in which ML and DL based models are utilized to handle gait parameters spread across datasets.

Chapter 2 presents a preliminary background of gait parameters, gait analysis and different factors affecting human gait. Cognitive based gait activities which are explored using multi-modalities in this research and the related research are also introduced. Contribution of different modalities such as inertial sensors, video camera, floor sensors and multi-modalities involved in gait studies is also presented. Theoretical concepts behind various ML and DL models have been explored in literature review.

Chapter 3 describes the multi-modality system involved in the study of gait analysis i.e., FS and AIS. Complete sensing principle and architecture of both systems has been described. The data acquisition and pre-processing methods have been discussed before allocating data to ML or DL models for further processing. Gait activities data has been analysed for mostly visible gait parameters from the datasets from two modalities.

Chapter 4 elaborates the feature level extraction approach using Principal component Analysis (PCA) and Canonical Correlation Analysis (CCA). Analysis of extracted features and feature selection has been performed to calculate the best optimum features from the data of two modalities. Correlation inside the individual and combined dataset has been highlighted. Proposed fusion approach using PCA and CCA has been explained. Datasets obtained from two modalities using four gait activities: walking while subtracting 7, listening to a story, texting on a mobile and talking to operator is described along with the data distribution for training and test sets required for ML model classification. Part of this work has been presented in publications number 1 and 4 under section 'List of Publications'.

Chapter 5 explains the DL models-based approach for the extraction of gait activity features obtained using two modalities. The observation span is increased to seven gait activities including: normal walk, fast walk, walking while subtracting 3, subtracting 7, listening to a story, texting on a mobile phone and talking to operator. Proposed fusion strategy is tested on DL models: Feed Forward Neural Networks (FFNNs), 1D/2D Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM). Datasets comprising gait activities have been described for single and multi-modality cases. Part of this work has been presented in publications number 2 and 5 under section 'List of Publications'.

Chapter 6 presents the results and discussion related to the proposed multi-modality sensor fusion. Selection of features extracted using PCA and CCA has been discussed and comparison has been made between the two methods. Furthermore, the roles of ML and DL models for all gait activities has also been exploited in case of single and multi-modality cases. The contents of this chapter are also presented in publications number 1 and 2 under section 'List of Publications'.

Chapter 7 summarizes the insights of this research and concludes the results presented in this thesis. It also provides the recommendations and future work to be done following the route of the proposed multi-modality sensor fusion.

Appendix A presents the abbreviations used in this research.

Appendix B is furnished with codes written in Arduino, MATLAB and Python Integrated Development Environment (IDE).

Chapter 2

Background theory and literature review

2.1 Gait cycle

Gait defines walking activities in humans. Gait can be defined as a conversion of human brain action to body muscle motion sequences which result in a walking pattern. Commands are originated in human brain which are transferred through spinal cord to the lower body portion and consequently causes body muscular motion which is assisted by joints, bones, and other receptors. Gait can be perceived as a repetitive cycles of each foot results due to a sequence of periodic movements [11],

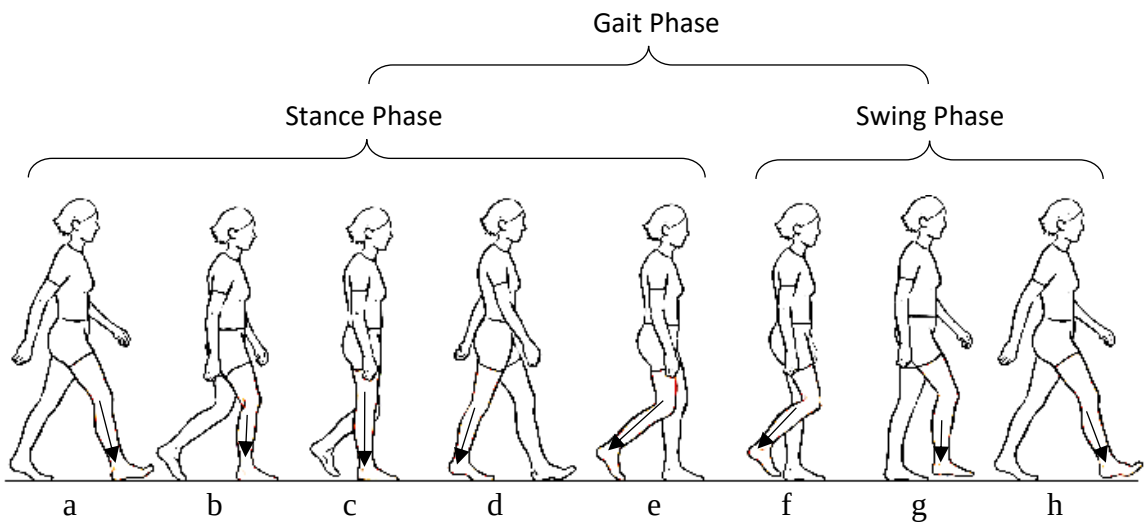


Figure 2.1: Stages of human gait cycle [135]

In figure above, where the subject can be seen while performing gait (with arrows marked on the right foot). It can be seen that going from heel contact of the right foot to the heel contact of the right foot back again, there a number of stages in each phase. These stages are as follows:

2.1.1 Stance phase

a) **Heel strike** – It is a short period which starts when the leading foot touches the ground, and it is the initial phase of double leg support.
b) **Foot flat** – Body adjusts the impact of leading foot on ground in pronation. Knee is flexed and lower body is at its lowest position.
c) **Mid stance** – During this period all body weight is on one leading leg and comprises main time of stance phase. Lower body moves to the highest position and the trailing feet leaves the ground.
d) **Heel off** – The trailing heel (leading previously) leaves the floor during this period. Lower body leaves the highest position whilst the leading foot in contact with the ground.
e) **Toe off** – Toe leaves the floor during this moment. Body moves forward whilst all the body weight starts shifting on the leading leg with knee flex.

Stance phase comprises around 60% of the gait cycle approximately.

2.1.2 Swing phase

f) **Initial swing** – Moment after toe leaves the ground and trailing foot goes in the air. All body weight is on leading leg with straight knee.
g) **Mid swing** – Foot remains in the air and crosses the other mid-stance foot.
h) **Terminal swing** – Moment before the heel strikes the ground and leading foot comes to the ground.

Swing phase comprises the remaining 40% of the gait cycle approximately.

2.2 Gait parameters

For healthcare in clinical environments, various sensing and data processing methods are used [12]:

- Cadence (steps per unit time)
- Stride length
- Acceleration
- Linear and Angular Velocities
- Direction of limb segments

- Step angle
- Step width
- Swing time
- Support time
- Ground reaction force
- Muscles' electrical activity
- Momentum and body forces
- Posture of body

All stated methods provide observational values related to health conditions however, it is not possible to capture the complete variability from gait data.

2.3 Analysis of human gait

Gait analysis is on the way to maturity with applications in many research areas. Many studies have described the potential of gait in differentiating individuals. Walking behaviour changes on the basis of age, weight, height and gender in humans. Human gait can be classified as either natural (used by humans instinctively) or trained (used by humans not instinctively or learned through training) [13]. Abnormal gait is a specific type of gait in which humans walk in a way different than natural. Abnormal gait could be caused by ageing, physical disability or event, such as a stroke. However it could be improved through medical treatments and exercises.

In the medical field, the study of human gait is used to diagnose neurological diseases such as Huntington's, Parkinson's, Alzheimer's, myelopathy, specific types of dementia, neuro-muscular diseases, brain tumors, spinal amyotrophic disease, multiple sclerosis, cerebellar ataxia etc. A non-invasive approach is implemented [2] using Inertial Measurement Units (IMUs) placed on patients leg to early diagnose the effects of Alzheimer's and Parkinson's disease. Abdulhay et al. [14] presented an approach to detect gait impairment and tremor occurrences during different stages of Parkinson's disease. They extracted and used temporal features such as stance, swing phases and stride time to distinguish Parkinson's disease patients from healthy individuals.

Progressive loss of structure or function of neurons in patients suffering from neurodegenerative diseases is studied by Chakraborty et al. [15]. They employed force

sensors in combination with Received Signal Strength Indicators (RSSI) to effectively monitor patients with Parkinson's over wireless network. Furthermore, IMU based sensors have been used to analyse gait parameters from lower limb prosthesis users [16].

Gait analysis in security applications makes it more robust and trustworthy whilst identifying individuals with minimal invasion. Use of CCTV cameras is reported for the identification of individuals [17],[18]. Subject identification is also achieved using footstep signals as ground reaction forces in [19],[20].

Moreover, gait analysis has applications to assess the ability of sportsmen after injuries occurred during sport activities. Gouwanda et al. proposed body mounted sensors to acquire human motion for various indoor and outdoor sports training activities and clinically rehabilitee patients [21]. Lee et al. presented a review to determine the viability of the dual-task paradigm for the comprehensive evaluation of athletes' sports related concussion [4]. Analysis of human gait using different modalities in

2.4 Factors affecting human gait

Gait in humans is a distinguished biometric method just like other biometric modalities like face, fingerprints, iris detection to give valuable information like identity, age, gender, and ethnicity. Gait patterns are difficult to duplicate due to their individual nature. However gait patterns get affected by many factors such as illness [22], fatigue [23], emotions [24], cognitive and motor tasks [25]. In addition, gait is also prone to influence from external factors such as clothing, shoes or carrying load [26].

2.5 Cognitive based gait activities

Gait is no longer considered as an automated activity that utilises minimal higher-level cognitive input. In fact, the multi-faceted neuropsychological effects on gait and the interconnection between the mobility control and related factors incorporate new research pathways [8]. There are several factors that may cause variance in gait patterns during the performance of cognitive activities.

Woollacott et al. [27] reviewed the connection between attention and control of body posture during gait. They highlighted the involvement of cognitive factors to maintain balance during standing and walking patterns. Additional task-based gait is used to observe

the performance among healthy and the unhealthy adults. They concluded that complexity of a dual task whilst gaiting directly effects the attention level of the individual.

O'Sheas et al. [28] observed the performance of simultaneous motor or cognitive tasks such as walking at a certain speed (single task), transferring a coin (motor task) and performing number subtraction (cognitive task) on 15 PD patients. They evaluated that gait changes whilst performing a cognitive and motor demanding task, however an additional secondary task does not necessarily determine the severity of disease. They concluded that task-based gait variations could help to predict falls and the individuals with lower executive control are prone to the risk of falling.

Relationship between walking, thinking and falling is exploited by Herman et al [29]. They reported that cognitive load-based tasks have higher chances on gait disturbance among individuals having fall history. They evaluated the effects of executive control deficits which can lead to fall on 262 healthy older adults. They concluded that future falls can be predicted among individuals with lower executive functions.

Ijmker et al. [30] investigated gait variations in single and dual tasks among healthier and dementia patients. They concluded that gait variability and stability have a direct relationship with executive functions and measures should be deemed during the diagnosis of dementia.

Costilla-Reyes et al. explored the capability of POF based FS (the "intelligent carpet" [31]) to detect changes in gait patterns using 10 manners of walking and 3 cognitive oriented tasks [32]. They demonstrated that raw level data leads to substantial better performance than manually extracted features. Raw information obtained from footprint imaging system evaluated using deep convolutional neural networks (D-CNN) and achieved superior f-score of 97.88% $\pm$1.7% over other ML methods.

Zebin et al. [33] reported 6 daily life including 3 walking activities with 92% average recognition accuracy using only accelerometer and gyroscope data as inputs. They presented Long short-term memory (LSTM) based approach to explore the correlation between successive time-samples in raw datasets. In their further research [34], they compared the overall performance of Deep neural networks (DNNs) (86.55%), LSTM (92.2%) and Convolutional neural networks (CNNs) (96.4%) for the same dataset.

However, implementation of a robust and an accurate multi-modality approach to observe and analyse the effect of cognitive tasks on human gait is still a challenging problem.

2.6 Gait modalities

From the evolutionary research in the field of gait analysis, it has been suggested that various gait modalities have attempted to capture the uniqueness of human gait from the biomedical measures specific to physical dimensions of human body and its parts or the forces generated during a gait cycle. In the past few years, the exponential rise in the efficiency and capabilities of sensor systems has resulted in the production of various gait sensing modalities [35]. Gait sensing modalities can be grouped into three main groups: Inertial sensors, video cameras and floor sensors, which are described as follows:

2.6.1 Video cameras

Human gait is acquired through video cameras. High quality video cameras are required to use under moderate lightening conditions. Image and video processing techniques are used to extract gait features from the data. Normally the aim is to identify the person, or the activities being performed. A prototype of video cameras in shown in figure 2.2. Basic video is captured with two cameras or more with a known focal length at a fixed distance from subject. Results obtained from all cameras are further calibrated to give correct results. The information obtained using video camera is analysed using image processing techniques such as filtering, edge detection, segmentation and thresholding etc [36].

In literature, gait recognition can be further subdivided into skeleton model based and skeleton model free approaches. In model-based approach, the multi-segment skeleton models are fixed on video sequences. This approach requires computational resources as it relies on matching of skeletal segments on the image sequences as proposed by [37],[38]. The extracted features are further classified based on ML or DL models.

On the other hand, model free approaches rely on feature extraction from video sequences and require feature engineering [39],[40]. In this approach, data is mainly represented in the form gait energy images (GEI), silhouettes and chrono-gait images.

There are also many publicly available datasets related to gait based video sequences such as MoBo (The CMU motion of body database) [41], CASIA [42], OU-ISIR treadmill [43], OU-ISIR [44], TUM-GAID [45] and human ID challenge [46]. These datasets are available to test, and train using ML and DL models to access and improve the performance of the datasets.

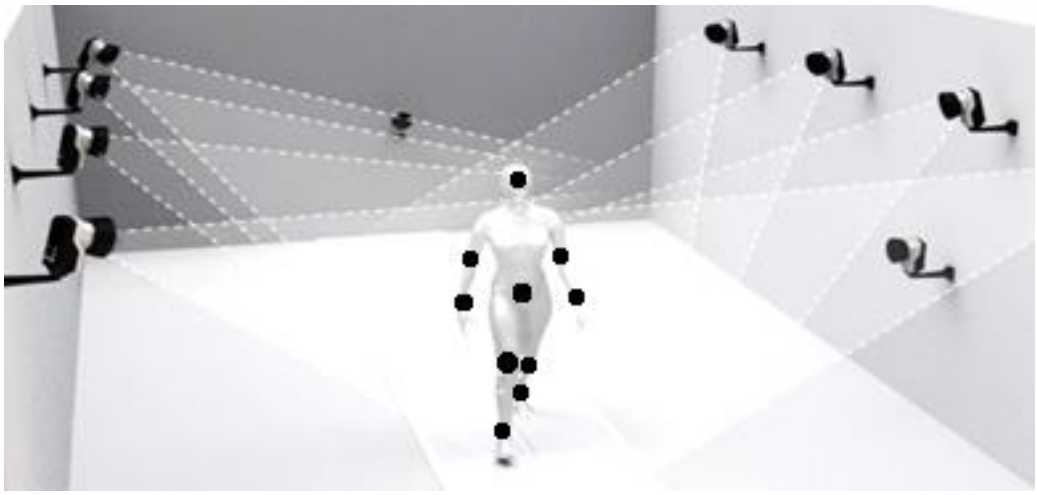


Figure: 2.2 Prototype video cameras with markers on different body parts to create a skeleton [136]

2.6.2 Inertial sensors

Use of ambulatory sensors to monitor and classify human activities and gait has proven to be important [47]. A specific type of sensor, the inertial measurement unit (IMU) has been widely used due to its small size, cost, light weight and high precision characteristics. IMU sensors consist of accelerometer, gyroscope and magnetometer which give information about the acceleration, angular velocity and the heading direction respectively. IMU based sensors are used to calculate the acceleration resulted from the acting forces with the help of an accelerometer whilst a gyroscope is used to measure the angular velocity in the response of rate of change of the sensors orientation. Data obtained from an IMU based sensor provides a comprehensive report on the acceleration, velocity, gravitational forces and human body orientation. IMU sensor could be worn on different parts of human body such as head, chest, waist, thigh, shank and foot [48]. When connected with batteries, microprocessors and communication devices, make an IMU based system. IMU sensors on human body for testing can be seen in figure 2.3.

IMU sensors have been extensively used inside smart phones with advantages of predictable variability and position whilst avoiding the requirement of additional hardware support. The benefits of gait assessment and monitoring in patients can also be realized with smart phone based IMUs [49]. The smart phone now days is capable of performing all necessary tasks such as making decisions and contacting the health providers in case of emergency situations.

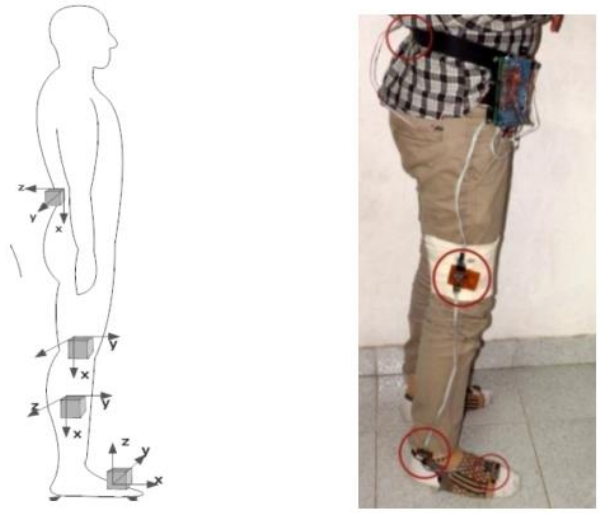


Figure 2.3: IMU sensors on legs to detect gait [50]

For gait analysis, mainly experiments are conducted on Heel strike and Toe-off events which determine the stance time and swing time of subject in gait cycle. Performance is checked on a dataset of information using different classification algorithms and techniques [50]. Panebianco et al. presented a systematic review to assess the human gait and its temporal parameters in terms of accuracy and repeatability using 17 algorithms [51]. 5 IMUs were used, one on the back, two on the shanks and two on the feet. It was determined that for human gait detection and estimation of stance time, algorithms based on the acceleration measurements on the shank and foot perform better than those based on lower trunk. It was established that for human gait detection and estimation of stance time, algorithms based on the acceleration measurements on the shank and foot perform better than those based on the lower trunk. However, the sensor position did not affect the step estimation.

In deviation from normal gait, the analysis of human gait for healthcare has attained the interest of researchers and clinical studies. Various methods and techniques have been developed and proposed to classify and analyze neurodegenerative conditions and prevents of fall among elderly patients. IMU sensors are widely used in recognition of human motion disorders such as Parkinson’s disease (PD) [52], [2] and early detection of Freeze Of Gait (FOG) [53].

However, it is still challenging to directly capture and analyse gait signals due to the complex spatio-temporal nature and the difficulty to relate the data directly specially in case of larger number of observations [48],[51]. Many feature extraction techniques have been

adopted which are time-consuming and require knowledge of the context for which the gait signals are acquired. Feature extraction methods are time consuming and manual therefore, deep DL has appeared as an automatic and reliable tool to hunt the distinctive features of human gait and outperforms the other handcrafted feature-based techniques.

A deep learning approach has been proposed with 5 IMU based sensors attached on lower back, thighs and shanks for human activity recognition has been proposed by Zebin at al [54]. Gait features are automatically extracted from raw data using CNN models and superior performance using CNN is compared with the other machine learning techniques. Similarly, Dehzangi et al. [55] used 5 IMU sensors attached on chest, lower back, right wrist, right knee and right ankle. Data is collected from 10 people which is subject to early and late fusion methods using deep CNNs. Superior performance is achieved using the later fusion method.

2.6.3 Floor sensors

Floor sensors (FS) are used to capture the force produced on the ground by human foot during a gait cycle. This interaction of human body is natural and cannot be changed or altered at will. However, due to temporary or long-term health condition either neurological or physical conditions changes can take place in this interaction. Gait monitoring using FS requires minimal intervention which make FS suitable for long term and continuous data capture [56]. FS are useful to capture the distinct information related to gait events, record the evolution of walking behaviours, and observe any reactions to the psychological and physical involvements.

FS provide an unobtrusive way of acquiring gait information and are mainly installed at the front entrance of buildings or in access control areas. These systems can also be used to identify the location of subject within a certain area [57]. FS can be used on the factory floor to provide data needed for monitoring of the position and activity of ambulatory industrial robots and, in cases of co-occupancy with humans, can provide additional information needed for health and safety.

Mostly, FS comprises a set of sensors or force plates placed on the floor. Resistive, capacitive, inductive or fiber optic-based sensors are commonly used, sample prototype of FS is shown in figure 2.4. Data obtained from these is used to analyse and calculate gait

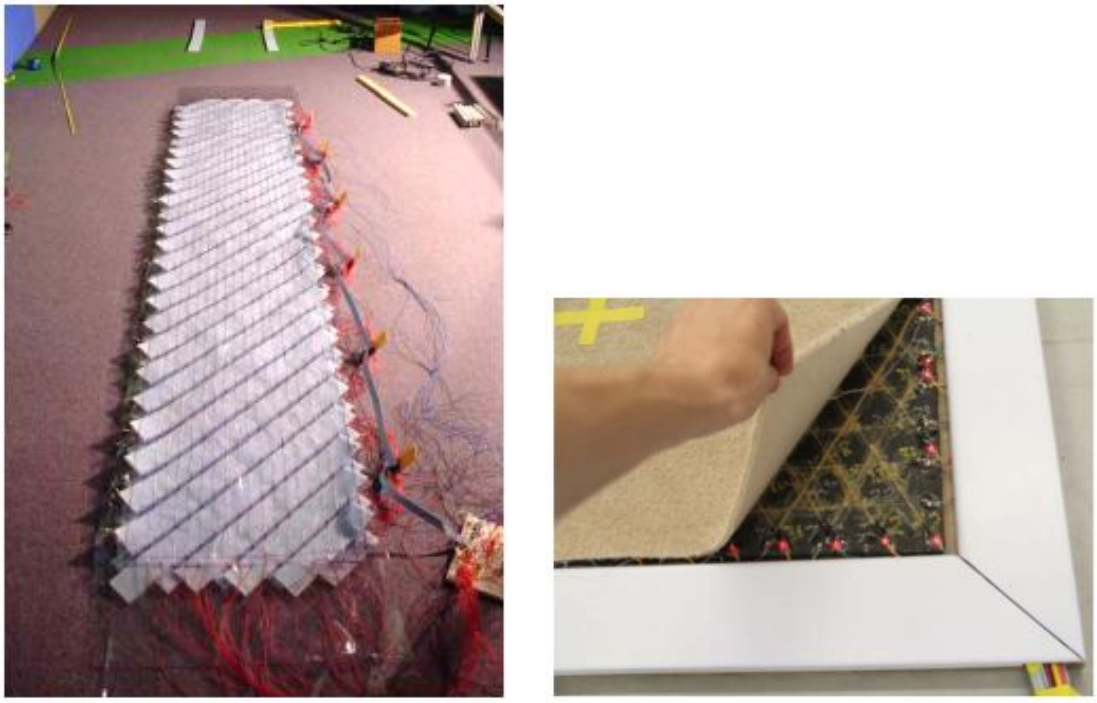


Figure: 2.4: Prototype FS Left: Resistive sensors [105], right: POF based sensors [10]

features. Middleton et al. [57] used different gait features such as stride length, stride cadence and ratio between time on toe to time on heel and achieved 80% recognition accuracy with a resistive sensor mat. Vera-Rodriguez et al. [58] have assembled SFootBD database using 20,000 footstep signals obtained from more than 120 volunteers. Gait analysis has its implementations in healthcare such as diagnosis of flat foot among children [59], monitoring for fall detections in homes [60] and effect of cognitive dual task on human gait [28], [61].

Raw values from floor sensors can be used with ML methods and techniques, circumventing calculations of pre-determined “man-made” gait features e.g., from image reconstructions. Costilla-Reyes et al. [6] used raw readings from floor sensors and achieved 93% of classification accuracy using Support Vector Machine (SVM) on large datasets. Further DL has been used extensively for automatic feature finding from the complex spatio-temporal gait signals. Singh et al. [62] proposed a 17 layer based CNN and gated recurrent units based Inception-v3 model to extract gait features from the images constructed from ground reaction force. 13 people are tested with their samples taken on a 80cm x 80cm grid area of resistive FS and yielded 87.66% accuracy. Again, Costilla-Reyes et al. implemented deep learning models such as Feed forward neural networks (FFNNs) and Recurrent Neural Networks (RNNs) to perform classification of 10 walking manners and 3 dual task [63]. They demonstrated that DL based models outperformed the other ML based approaches with overall accuracy of around 97.88% and few exceptions due to the shortage of training

dataset. They also established UoM-Gait-13 dataset representing raw spatio-temporal gait signals recorded at 1400 frames at 256 Hz using 116 FS.

2.6.4 Multi-modality sensor fusion

Multi-modality sensor fusion combines data from multiple modality sensors resulting in an information category of new representations, which is distinct from the original and individual information dataset [64], [65]. Multi-modal approach is used for higher reliability and accurate identification by combining results obtained from more than one modality. It is obvious to achieve the improved accuracy when information from one modality is integrated with another. Not only just in accuracy, but the multi-modality approaches also create robust and fool proof biometric system as compared to single modality systems. In literature, sensor fusion based on multi-modality systems can be categorised into two main categories:

2.6.4.1 Feature level based multi-modality sensor fusion

Feature-level fusion of different modalities involves extracting features from multiple sensors and generating new representations which can be different [64]. Feature-level fusion is helpful in situations where lower computational cost is a key challenge. The focus on feature-level fusion is essential, since the efficiency of gait analysis depends on employing the maximum variability of data by automatically extracted features, rather than using hand-crafted features based on observational practice.

Shakhnarovich at al. [66] combined data from face recognition and video camera gait features. Canonical view estimation, rendering and recognition has been implemented on image sequences. Face recognition provided 80%, video camera-based gait produced 87% and combined results were 91% respectively. Zhou et al. reported the combination of side face and gait as two biometric sources to achieve recognition [67]. They tested their approach on 45 people and observed an improvement of 100% combined as compared to 64.3% using face recognition and 85.7% using video recognition.

Elena Vildjiounaite et al. [68] proposed a unique way of combining speaker recognition with accelerometer based gait recognition. In a tentative test on 31 users, the EER was 2% - 12%, typically less than half of the EER obtained from individual modalities.

Modalities	Feature Extraction Methods	Accuracy Results		References
		Individual Modality	Multi-Modality	
Video face and Video Gait	PCA based canonical view estimation	Video face =80% Video Gait = 87%	92%	G.Shakhnarovich et al. [66]
Video face and Video Gait	Combined PCA and Multiple Discriminant Analysis (MDA)	Video face = 64.3% Video Gait = 85.7%	88.9%	X.Zhou and B.Bhanu [67]
IMU and Flex	Gaussian Mixture Model (GMM)	-	92.86%	Y.Wang [137]
EMG and IMU	Marginal Discrete Wavelet Transform (mDWT) and Mean	EMG = 76% IMU = 81%	83%	A.Gijsberts and B.Caputo [138]
Kinect and IMU	Depth motion map (DMM) and Linear Discriminant Analysis (LDA)	Kinect= 86.34 IMU = 90.30	98.23%	Chen et al. [139]
Palm Print and Palm Vein	CCA	Palm print = 95.40% Palm Vein = 92.87%	96.60%	X.Yang and D.Sun [140]
EMG and EEG	Linear Discriminant Analysis (LDA)	EMG = 77% EEG = 75%	92%	X.Li et al. [141]
Depth and IMU	Fast Fourier Transform (FFT) and Histograms of Oriented Gradients (HOG)	-	96.91%	Y.Li, et al. [125]
Video and IMU	Activity specific methods and Direct mapping	Video= 82.9% IMU = 78.8%	86.6%	L.Tao et al. [142]
POF and IMU	Compensation equation and Quaternions	-	1% best - 4% worst case error	A.G.Leal-Junior et al. [143]
Inertial & force sensors	Hidden Markov Model (HMM)	-	92.8%	Beil et al., [144]
Inertial, ECG & respiratory sensors	Time domain, frequency domain and non-linear Heart Rate Variability (HRV)	-	92%	Rahman et al. [145]
Video & floor sensors	Enhanced Gait Energy Image (GEI) and Multilinear PCA	Footstep EER = 10.66% Gait EER = 8.43%	EER = 4.8%	Vera-Rodriguez et al. [69]

Table 2.1: Feature level-based fusion of multi-modality systems

Note: EER = Equal Error Rate

Vera-Rodriguez et al. [69] considered gait and footstep biometrics using manual feature extraction techniques to perform fusion using machine learning techniques. This approach is amenable to DL methods for automatic extraction of fused features and improved accuracy.

Table 2.1 reflects our awareness of research on multi-modality sensor fusion at 'feature level' (including some not on human gait data). Here, the calculated accuracy of fusion pertains mainly to the methodology of the data processing and not necessarily to the merit of the sensing results.

2.6.4.2 Deep learning based multi-modality sensor fusion

Deep Learning (DL) has become the state-of-the-art in many pattern classification techniques such as iris [70], face [71], finger-print [72], palm vein [73], ECG [74], human action [75] and gait [76] etc. In many inertial sensor-based modalities, DL is used to fuse the body position and orientation in the artificial neural network layers. Similarly, log of forces related to feet obtained using either switch sensors, pressure plates or POF sensors are fused in case of floor sensors. In table 2.2, we have summarized the research on sensor fusion of multi-modality gait data based on DL.

Mazumder et al. proposed a multi-channel redundant fusion technique to detect stride time and gait phase [77]. Basic theme of prosthesis and lower limb exoskeleton is followed to generate the lower limb joint trajectories. This technique used Radial basis ANN to process the joint trajectories across each gait event and phases. User intention to start, stop and change in a particular pattern is estimated during gait cycles. Modalities involved in this research include a four-channel myoelectric sensor for electromyography (EMG) signals, IMU and foot pressure sensors. The proposed study was conducted on five subjects walking on treadmill, an overall classification accuracy with minimum square error smaller than 0.05 is achieved.

In healthcare context, gait phase detection has been performed using DL methods. Ding et al. [78] used one IMU sensor attached to the shank and 3 foot switches and performed real time gait phase detection. LSTM is used for gait phase recognition with an accuracy of 96.1% as compared to ML methods such as Support vector Machine (SVM) (89.1%) and multi-layer perceptron (MLP) (91.8%). A strong correlation between shank kinematics and gait phase were noticed in this work.

Modalities	DL Models	Extracted Features	Results	References
EMG, inertial & foot pressure sensors	RB-FNN	Stride time	Fused error rate = 0	Mazumder at al. [77]
Inertial & foot switch sensors	LSTM	Gait phase	Accuracy = 91.8%	Ding et al. [78]
Inertial, force sensors & depth camera	DNN	Gait phase	Accuracy = 95 %	Mun et al. [79]
Inertial & force sensitive resistors	ED-FNN	Gait phase detection	MAE = 2.1%$\pm$0.1	Vu et al., [80]
Depth camera & laser range finder (LRF)	LSTM	Gait stability prediction	F-score = 86.79%	Chalvatzaki et al., [81]
Video, inertial & pressure sensors	3D-CNN & LSTM	Gait activity recognition	Accuracy = 91.3%	Kumar et al. [82]
Inertial & force sensors	CNN	Gait recognition	Accuracy = 93.3%	K. Ivanov et al. [83]

Table 2.2: DL based fusion of multi-modality systems

To predict the gait spatio-temporal parameters using deep ANNs, Mun et al. used foot characteristics [79]. To achieve this, they have used foot feature measurement system (FFMS) based on force sensors and RGB-Depth camera along with a set of IMU sensors based on an integrated motion capture system. Gait features such as stride length/time, step length/time, velocity and single/double limb support, swing and stance times are fused over deep layers of ANN. This study was tested with 95% accuracy on 42 subjects for slow, fast and normal walks.

Vu et al. used gait phase detection in gait cycles for powered transtibial prosthesis [80]. They implemented an exponentially delayed fully connected neural network (ED-FNN) and it was tested on 7 subject performing daily walking on flat ground and 15-degree slope. Data is acquired from raw IMU sensors placed on lower shank and two force sensitive resistors for heel strike and toe-off det4ection. The proposed implementation consumes less computational power therefore was found suitable for autonomous systems.

For the elderly people walking with mobility supported platforms, Chalvatzaki et al., [81] presented a method to monitor on-line gait stability. They fused the information obtained from upper portion of body using depth camera and leg motion using LRF. LSTM based network is used to predict the gait stability state as safe or fall risk. They have achieved

accuracy of 84.36% with a combination of LSTM layers and two fully connected layers as compared to only LSTM layers, SVM and rule-based methods.

A multi-modal data acquisition system to implement using evolutionary decision fusion is presented by Kumar et al. [82]. Two LSTM models are used: one after CNN layers and the other to extract the spatio-temporal information from IMU sensors. Grey Wolf Optimizer (GWO) is used to combine the output of LSTM models. An overall accuracy of 91.3% for gait recognition is achieved from 23 subject, including 19 males and 4 females using four different gait activities: normal, fast, walking while listening to music, and walking while watching multimedia content on a mobile.

Ivanov et al. performed identity recognition and test 59 users by acquiring kinetic and kinematic data from multi-modal sensor enabled footwear using inertial and force sensors [83]. CNN are used with four strategies and multiple segmentation overlaps. Parallel multi-cascaded CNN architecture with Extreme Learning Machine (ELM) and 70% overlap attained superior accuracies over any other strategy.

To access our proposed study multi-modality sensor fusion, we have thoroughly investigated and implemented different classification models which are discussed in the next section.

2.7 Machine learning models

Classification is a methodology to predict a set of categories for test data, based on trained/observed data set whose category is known. There are wide applications of classification models in Machine Learning (ML) which is a domain of Artificial Intelligence (AI) and first introduced by Arthur Samuel in 1959 [84]. ML has emerged as a key tool for sensor data analysis, is becoming a centric part of novel sensor design. ML is widely applicable has a major role to play in the field of data processing and sensor fusion in particular [85].

ML models are used to implement complex techniques and methods through predictions which can learn and make decisions on data obtained from multiple sensors. These models are also very useful in exploring the hidden aspects through learning background relationships and trends in data. ML has now entered everyday lives due to the reliable and repeatable results delivered, namely with facial recognitions, Kinect devices,

virtual reality headsets, speech or voice recognition over phones, robot dogs, online retail such as Amazon and Netflix etc. There are several ML models, we have focused a few suitable for our multi-modality sensor fusion approach, details of which are as follows:

2.7.1 Logistic regression (LR)

LR is a linear predictive analysis technique. This technique is used to describe relationship between one dependent variable and one or more independent variables [86]. LR is the extension of linear regression model for classification scenarios. Simple linear regression is given by,

$$y = b_0 + b_1 * x \tag{2.1}$$

Multiple linear regression is given by,

$$y = b_0 + b_1 * x_1 + \cdots + b_n * x_n \tag{2.2}$$

Linear regression works well for regression but not for classification as it is not capable to predict probabilities. Therefore, LR [87] uses sigmoid function to fit the values between 0 and 1 as shown in figure 2.5. Sigmoid curve is placed between the test data values according to the structure of data and predicted values are obtained from the projections of input values on the curve.

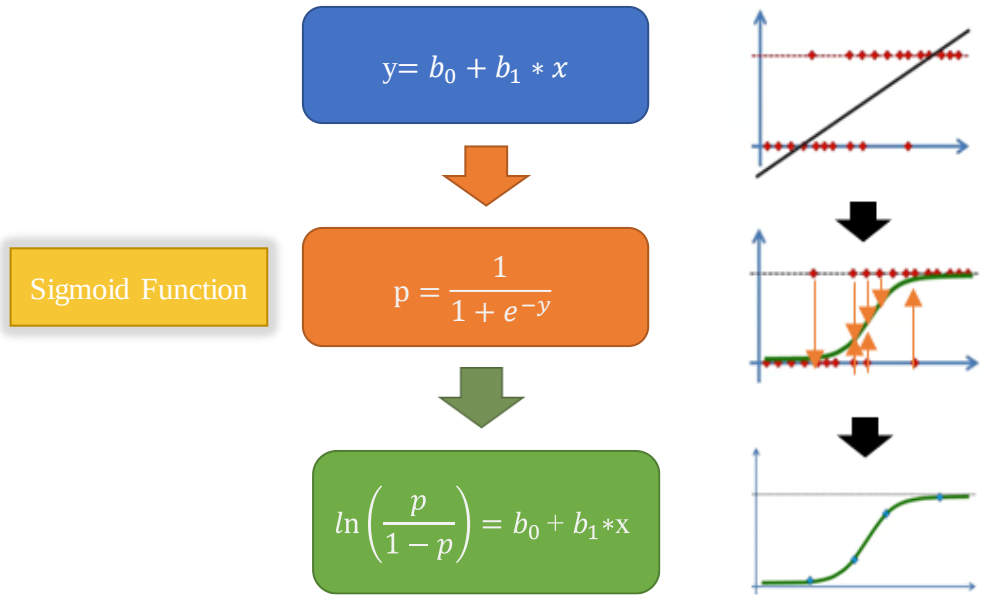


Figure: 2.5 Logistic regression implementation

2.7.2 K-nearest neighbour (K-NN)

K-NN is a non-linear and non-parametric model which uses a database in which data points are distributed among several classes to classify a new sample point [88]. K-NN algorithm can be summarized in the following steps:

i. Select the number of neighbours' k.

ii. For a new data point take the k nearest neighbours based on the Euclidean distance.

iii. Between the k neighbours, count the number of data points belonging to each category.

iv. Assign the new data point to the category for which the most neighbours have been counted.

For example, we have a green sample to classify, if k=1 it should be classified as 'Class 1' member and if k=3 it should belong to 'Class 2' as there are more neighbours as shown in figure 2.6.

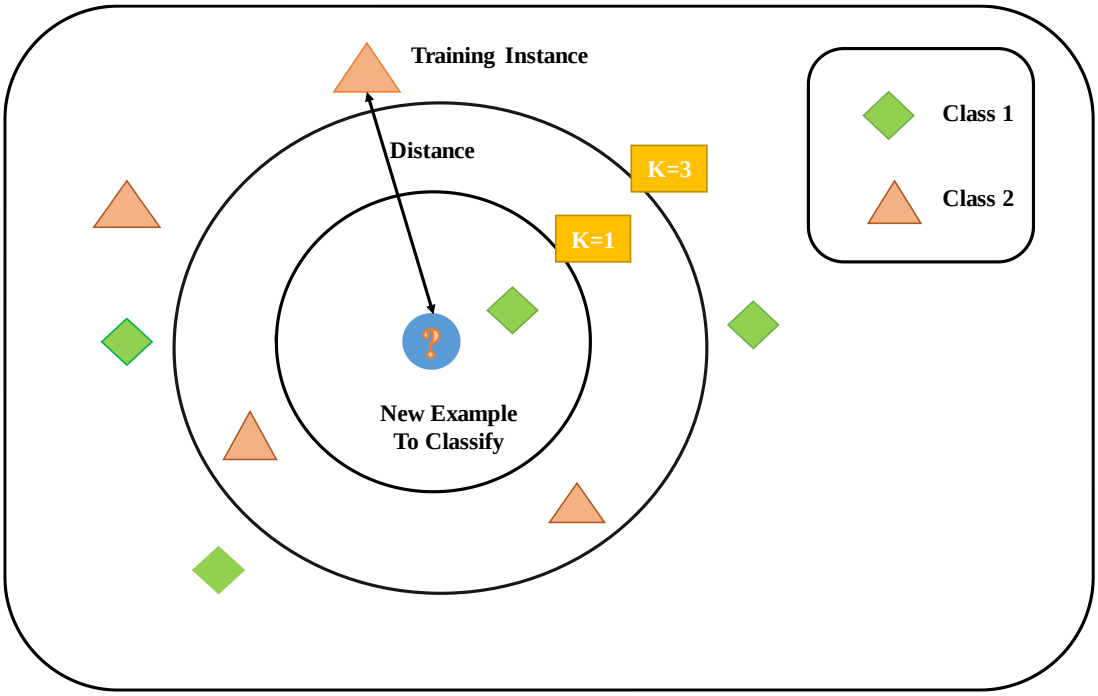


Figure 2.6 Working principle of K-NN

2.7.3 Support vector machine (SVM)

SVM is a non-probabilistic linear classifier. It creates a boundary called maximum margin hyperplane based on its maximum distance between the nearest data values for different categories [89]. These nearest points are called support vectors as shown below,

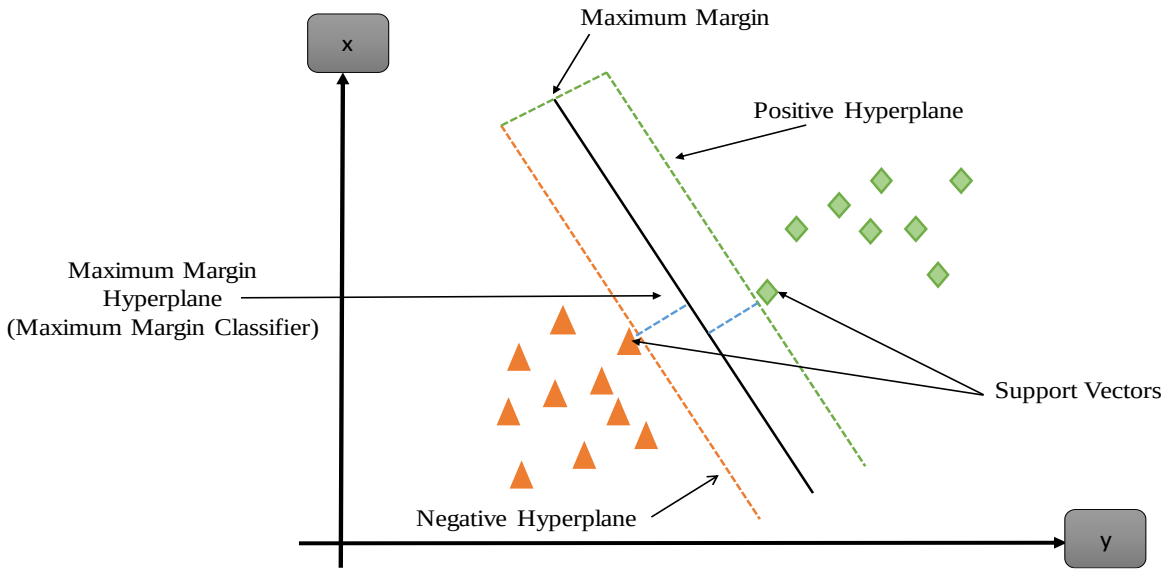
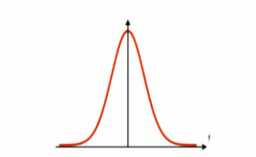


Figure 2.7 SVM example

2.7.4 Kernel-support vector machine (K-SVM)

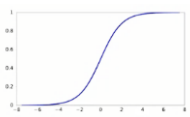
In case of non-linear data, SVM implicitly maps input data from 2D space into higher dimensional space using hyperplane and then projects back the data from higher dimensional space into 2D space [89] as shown in figure 2.8(a) and (b).

This procedure makes SVM a highly computationally intensive method in case of large datasets. Therefore, SVM makes use of 'kernel trick' in order to avoid map to and from higher dimensional space. Following filters are some of the filters used to avoid mapping data into higher dimension space.



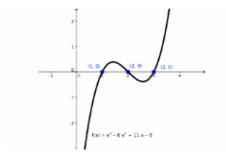
Gaussian Kernel:

$$K(\vec{x}, \vec{l^i}) = e^{-\frac{||\vec{x}-\vec{l^i}||}{2\sigma^2}^2} \tag{2.3}$$



Sigmoid Kernel:

$$K(X, Y) = tanh(\gamma . X^T Y + r) \tag{2.4}$$



Polynomial Kernel:

$$K(X, Y) = (\gamma . X^T Y + r)^d, \ \gamma > 0 \tag{2.5}$$

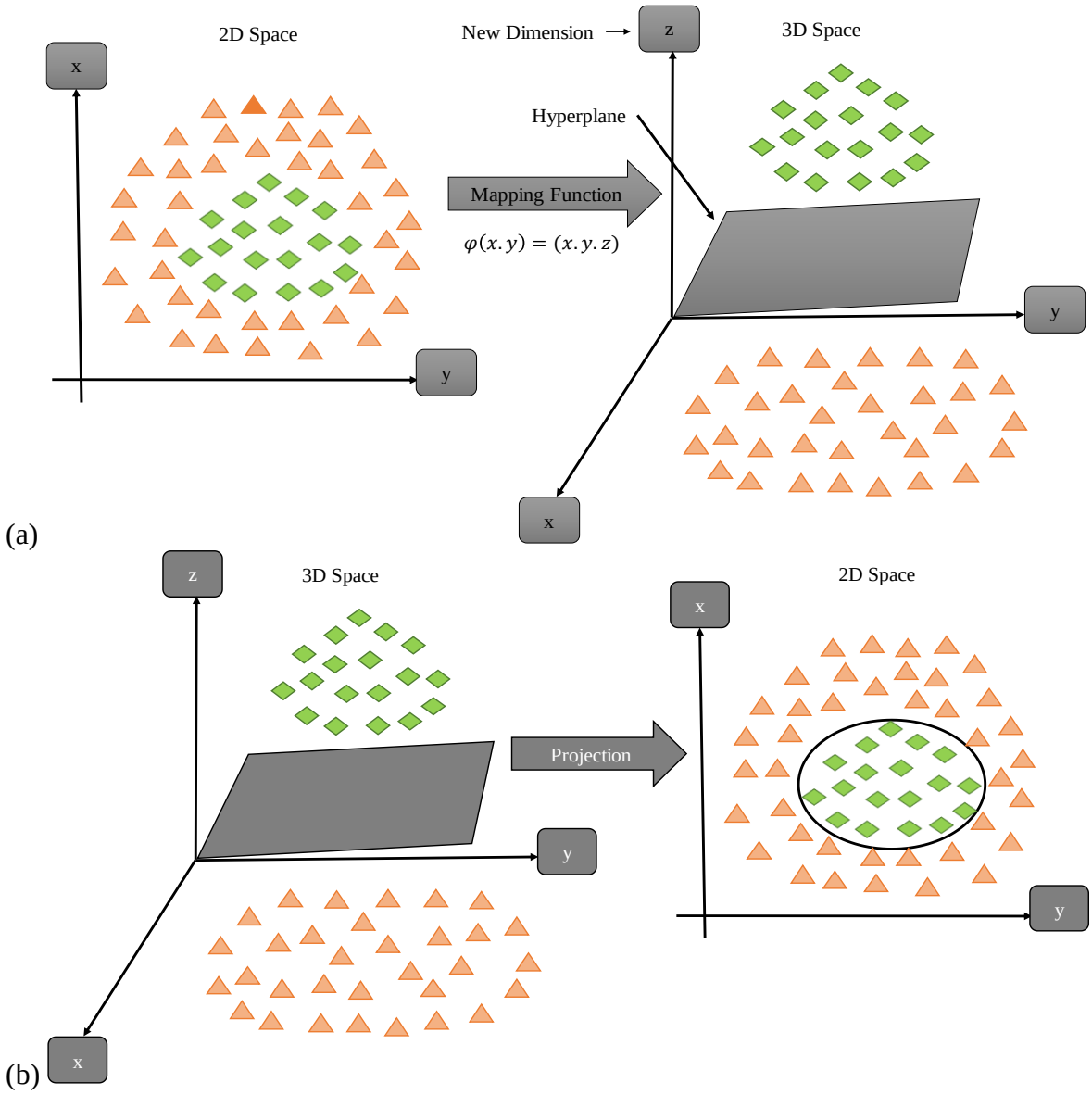


Figure 2.8 (a) Mapping of data in 2D space to a higher space using hyperplane
(b) Projection of higher space data back into 2D space

2.7.5 Naïve Bayes (NB)

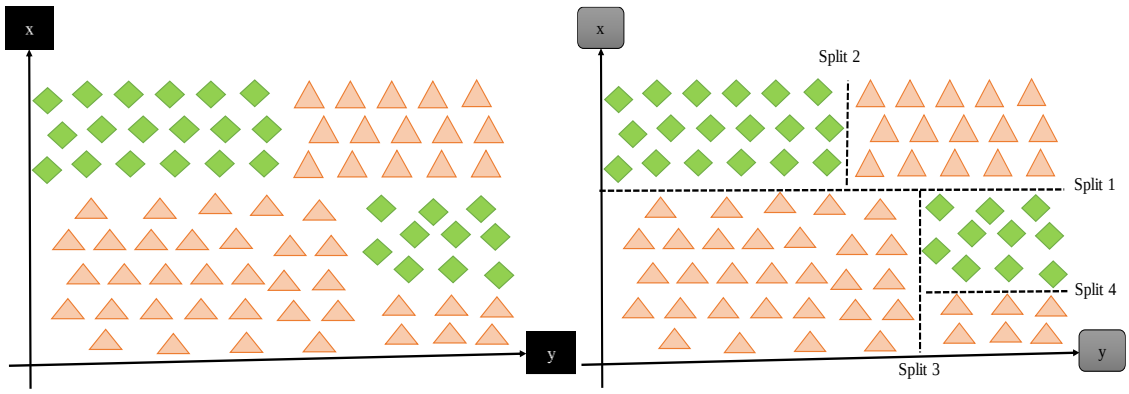
Naïve Bayes is a predictive analysis technique which is based on probability function to perform classification [90]. Basic equation used in naïve Bayes is as follows:

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)} \tag{2.6}$$

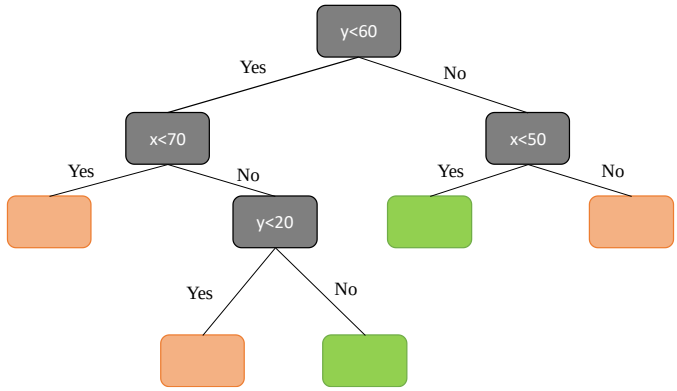
wherer P(c|x) iw the posterior probability, P(x|c) is the likelihood, P(c) is the class priority probability and p(x) is the predictor prior probability. Naïve Bayes uses conditional independence assumption whilst performing classification. Disadvantage is that it does not learn interaction between independent variables.

2.7.6 Decision tree (DT)

Decision tree works in the form of trees and breakdown a dataset into smaller datasets whilst building an associated decision tree at the same time [91]. This technique finds optimal splits that are going to maximize the number of points in each portion where a leaf represents a classification, or a decision and topmost decision node is called a root node. DT example is shown in figure 2.9. Decision trees are extremely fast at classification however a minor change in data may lead in major changes to the decision strategy.



(a)



(b)

Figure 2.9 (a) Sample data set before and after DT classification algorithm, (b) Example criteria to split data among branches and terminal leaves

2.7.7 Random forest (RF)

RF makes use of ensemble learning in which many models are combined to form a bigger algorithm with better classification and improved results [92]. RF algorithm can be summarised as follows:

i. Select ‘k’ data points or nodes randomly from the training set.

ii. For each ‘k’ nodes build the decision tree.

iii. Select the number ‘n’ trees and repeat steps 1 and 2.

iv. For a new data point, use the ‘n’ trees to predict the category and assign the new node to the category that has majority of votes.

Basically, random forest method is an ensemble of decision trees as shown in figure below, these decision trees are trained with bagging method. Main idea of bagging is to use a combination of learning algorithms to improve the overall result. However deep decision tree might cause over fitting on data.

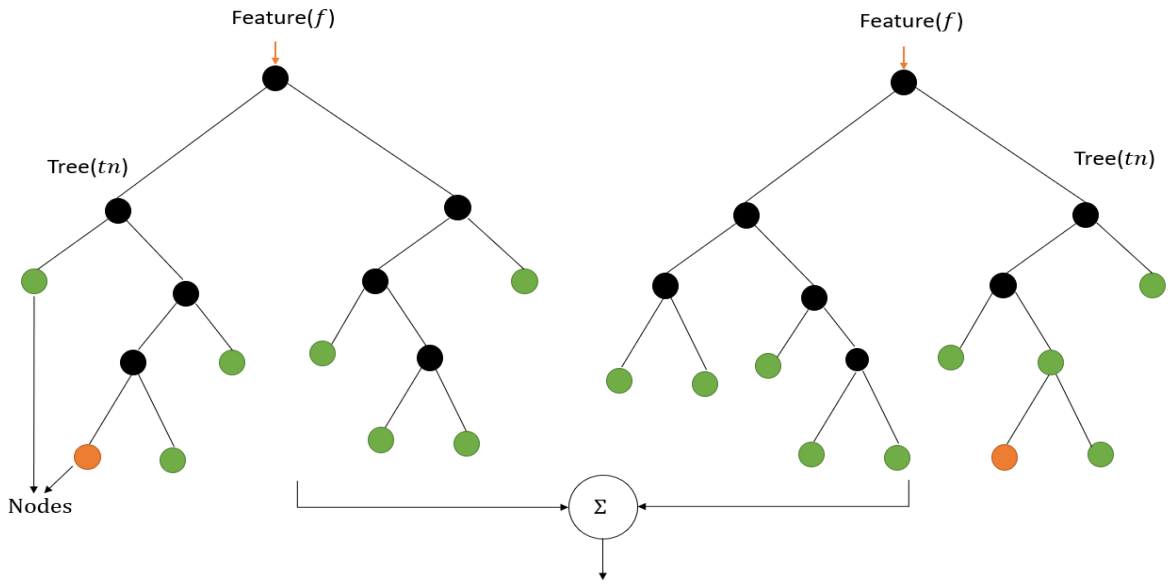


Figure 2.10 Example of RF algorithm

2.8 Deep learning models

Deep Learning (DL) is the most emerging and powerful branch of ML. DL algorithms can be used for a variety of complex tasks such as:

- Artificial neural networks for regression and classification
- Convolutional neural networks for computer vision
- Recurrent neural networks for time series analysis
- Self-organizing maps for feature extraction
- Deep Boltzmann machines for recommendation systems
- Auto encoders for recommendation systems

Focusing the multi-modality sensors fusion, we have selected the following models:

2.8.1 Feed-forward neural network (FFNN)

The neural network in which output from one layer in fed to the next layer in forward direction without any loops in the network is called a feed-forward neural network [93]. The basic architecture of a FFNN model consists of an input layer, few hidden layers and an output layer of neurons as shown below,

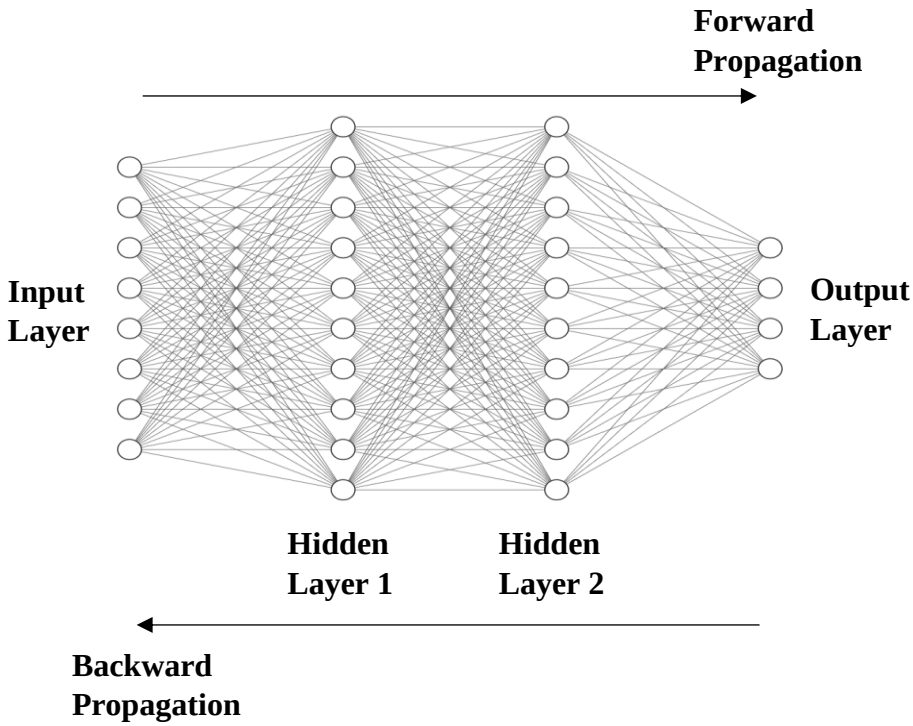


Fig. 2.11 A simple FFNN

The complete FFNN works as follows:

i. Weights are initialized with a value close to zero.

ii. **Forward propagation** – Neurons get activated, the outputs from each layer are passed in forward direction to the next layer. The weight of every neuron is multiplied by the input and passed through the activation function as shown in figure 2.12. Propagation continues until a prediction is achieved. For fused case, forward propagation takes places over the fused layers. The output of a k_{th} neuron is given by,

$$Z_k = \sum_{i=0}^{n} w_{ik} x_i + b_k \tag{2.7}$$

where *i* enumerates all neurons in the layer, *n* is the total number of neurons in the layer, *x* is the input, *w* is weight and *b* is bias of the k_{th} neuron.

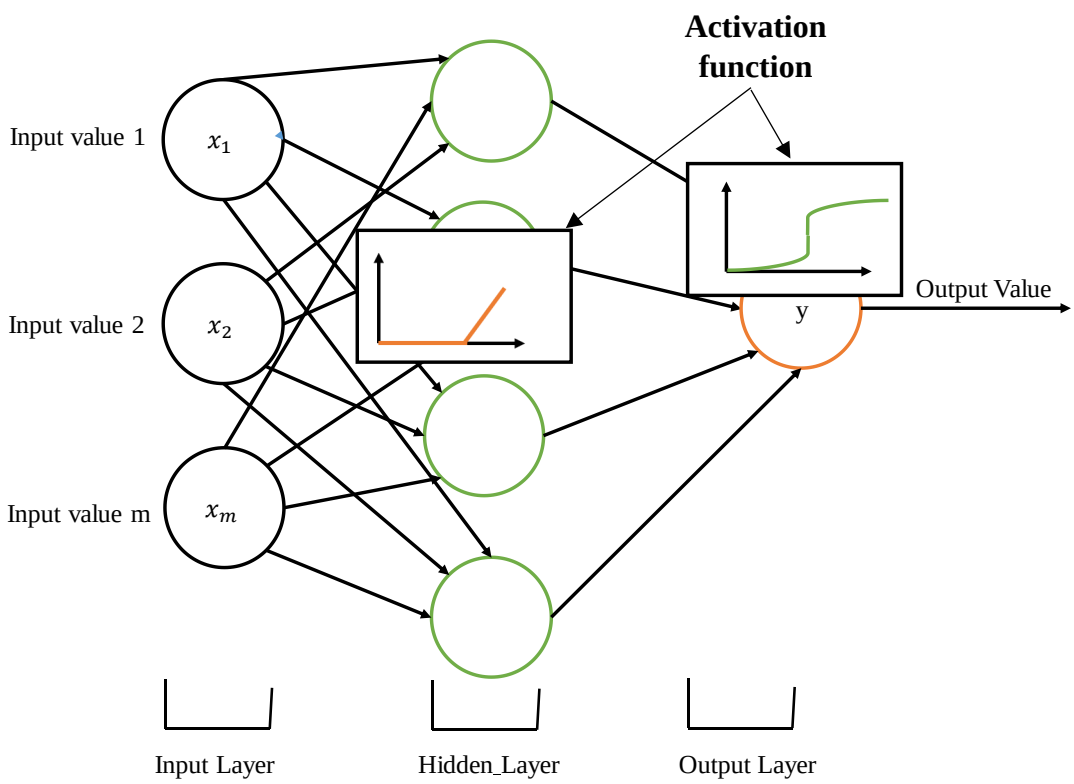
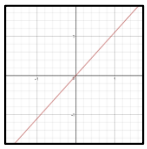


Fig. 2.12 Forward propagation using activation functions

The effect of every weight is determined by the activation function which allows the model to achieve a desired output between a certain range normally between 0 and 1. Activation function is given by,

$$A_k = f(Z_k) \tag{2.8}$$

In order to solve the complex problems, the network should be able to introduce the non-linearity. This purpose is achieved using activation function as well as the biases. There are many possible activation functions that exist:



Linear:

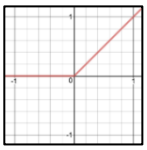
$$A = Z = w.x + b \tag{2.9}$$

Sigmoid:

$$A = \sigma(Z) = \frac{1}{1+e^{-z}} \tag{2.10}$$

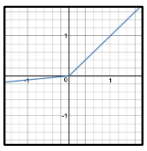
Hyperbolic tangent:

$$A = \sigma(Z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \tag{2.11}$$



Rectified Linear Unit (ReLU):

$$A = \max(0, Z) \tag{2.12}$$



Leaky ReLU:

$$A = \max(0.1\, x\, Z, Z) \tag{2.13}$$

iii. The predicted results are compared with the actual results and the error is quantified with the help of a cost function. Cost functions are found in literature, such as mean-squared-error [94] and cross-entropy [95]. We have used cross-entropy which uses logarithmic function to handle very small errors.

iv. **Back propagation** – The error is back propagated in the form of updated weights send to the neurons layer-wise in backward direction as shown in figure 2.13. Gradient based algorithms like stochastic gradient descent [96], conjugate gradient [97] and Adam [98] are the commonly used methods for error optimization, later is used to determine the learning rate of new weights and biases in our research. In the output layer, Z_k is the input vector for a linear classifier, namely a *Softmax* function given by

$$f_i(z) = \frac{e^{z_i}}{\sum_k e^{z_k}} \tag{2.14}$$

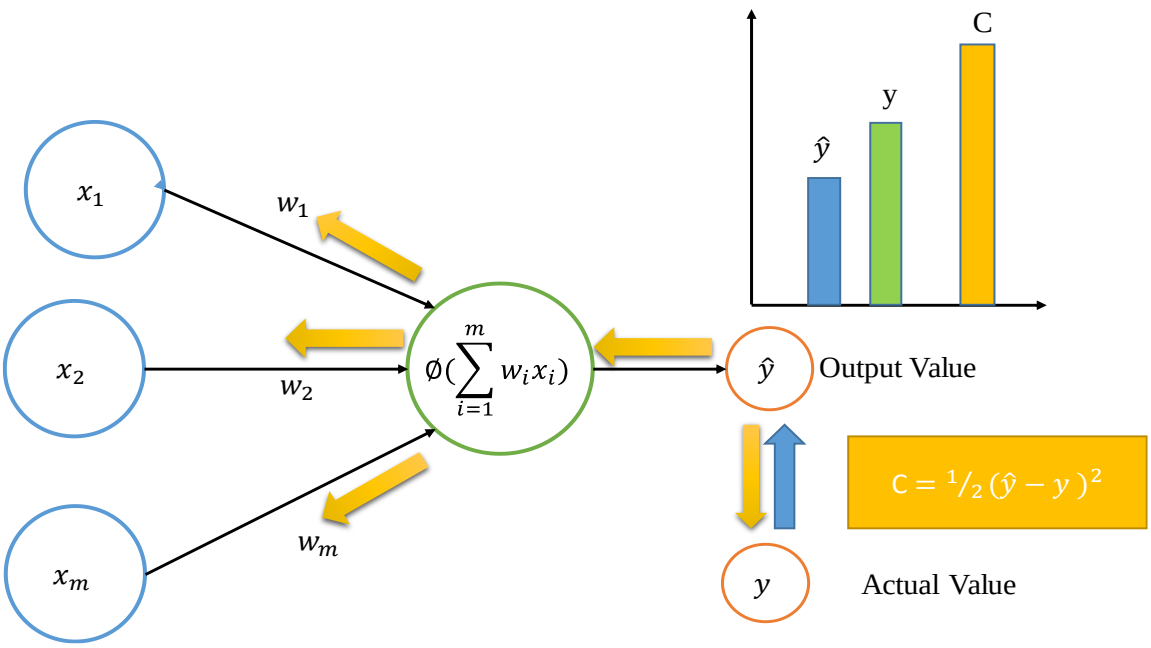


Fig 2.13: Backward propagation adjusted using gradient based algorithms

v. Steps 1-5 are repeated, and weights are updated after each observation (reinforcement learning) or a batch of observations (batch learning) from the training set.

vi. One epoch is completed when one whole training set passes through the FFNN.

2.8.2 Convolutional neural network (CNN)

CNN is a typical DL model which uses different levels of abstraction to learn the hierarchical representations of patterns existing in the dataset [99]. CNN are mainly used in the analysis of visual imagery. CNN consists of an input layer, convolution layers, down sampling or pooling layers, flattening layers, fully connected layers and an output layer.

The CNN functionality can be summarized as follows:

i. **Convolution** – Convolution operation is a combined integration of two functions, and it shows the modified behaviour of one function due to the other. Considering a case of images, suppose first function is the input image and the second function is the feature detector, kernel or filter then after performing the convolution operation result will be an extracted 'feature map' as shown in figure 2.14. Convolution operation also helps to reduce the size of input image however the amount of information does not get lost. Resulting feature map has features which are the integrals of input information and the filter.

A desired set of feature maps are created during convolution operations. Each feature map has its own filter applied on the input image. To increase the non-linearity in all feature maps, an activation function is required as shown in figure

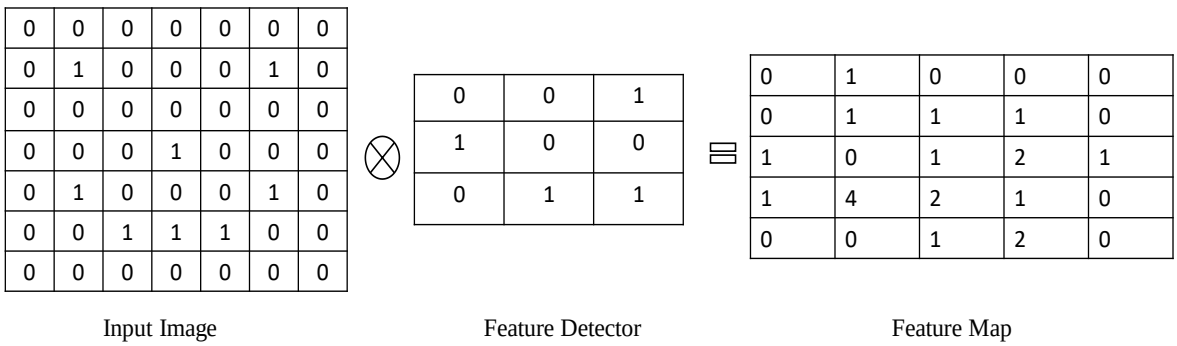


Fig 2.14: Convolution operation

2.15. Mainly used activations functions are sigmoid, tanh, ReLU (Rectified Linear Unit) and Leaky ReLU (as mentioned earlier in FFNNs).

The convolution operation is applied to automatically extract the unique variability features from the dataset. The complete output of a convolution layer is given by:

$$c_t^k = \sigma(b_k + \sum_{i=0}^{n} w_i^k x_{t+i-1}^k) \tag{2.15}$$

where σ is the activation function, b_k is the bias, w_i^k is weight of the k_{th} feature map and n is the size of the convolution kernel.

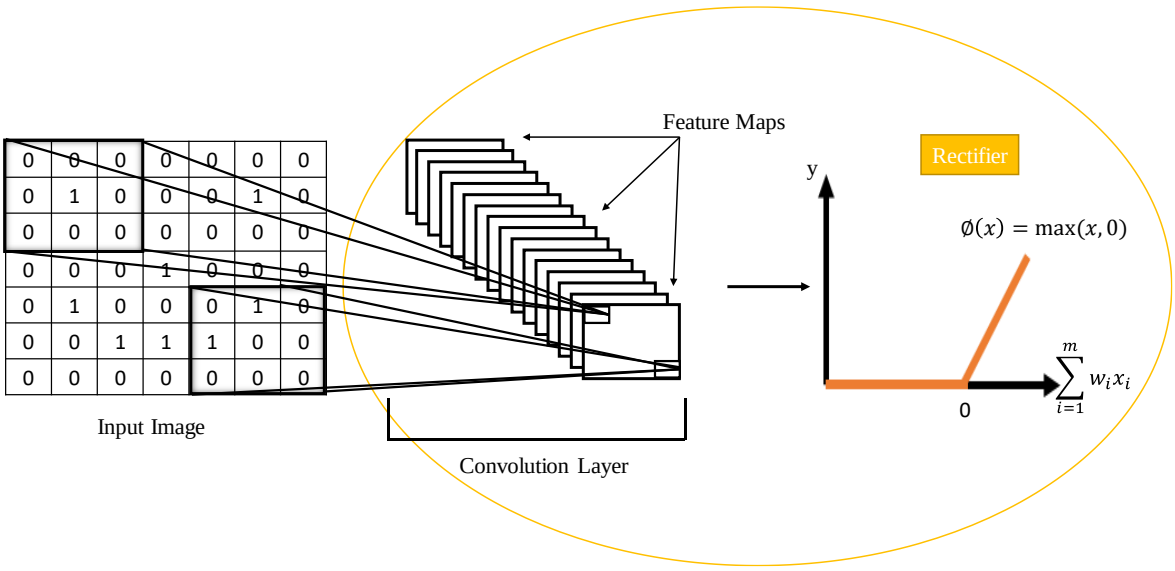


Fig 2.15: Convolution operation with ReLU

ii. **Pooling** – Max pooling is used to down-sample the large volume of data after convolution. Max-pooling outputs the maximum value from the nearby input values as given by:

$$m_t^k = max(m_{txL+o}^k) \tag{2.16}$$

where L is the stride length and o is the pooling size.

iii. **Flattening** – Extracted features are the 2D feature maps and required to get aligned in a 1D feature vector of inputs for fully connected layers:

$$f_i(z) = [f_1, f_2, ... f_k] \tag{2.17}$$

where k is the number of outputs from last pooling layer.

iv. **Full connection** – A fully connected layer is a specific type of hidden layer which is fully connected to its input layer. CNN make use of fully connected layers in which all inputs are connected to the hidden layer completely. Fully connected layers in CNNs do the discriminative learning just like ANNs. Complete convolution network can be seen in figure below,

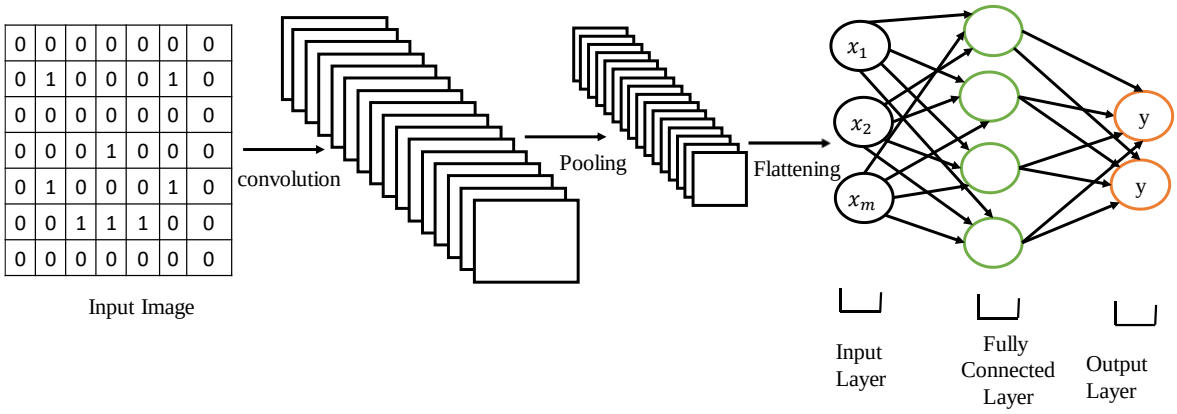


Fig 2.16: Implementation of complete CNN

2.8.3 Long short-term memory (LSTM)

Recurrent neural networks were introduced in the 80's [100], [101] for modelling of time-series data. The basic structure of RNNs is similar to FFNNs, where connections exist among hidden layer units based on time delays. These connections retain the information from previous inputs and help to find out the temporal correlations between events which are spread out in the dataset. However, the network output while cycling around recurrent connections gets affected from exponentially vanishing or exploding gradients [102]. Therefore, the efficient gradient-based technique, Long Short-Term Memory (LSTM), is introduced to cover the time lag between the time steps by enforcing constant error flow within special cells [103]. A single LSTM cell from a single hidden layer has three inputs: present input(x_t), previous output(h_{t-1}), previous memory(c_{t-1}) and two outputs: new output(h_t), new memory(c_t), as shown in figure 2.17,

LSTM cells are implemented with three gates and operations similar to a conventional FFNN, details of which are described as follows:

i). The forget gate f_t controls the contribution of previous cell c_{t-1} to pass through the current cell as c_t,

$$f_t = \sigma(w_f[x_t, h_{t-1}] + b_f) \tag{2.18}$$

where w_f,b_f and σ are the associated weights, bias and activation respectively.

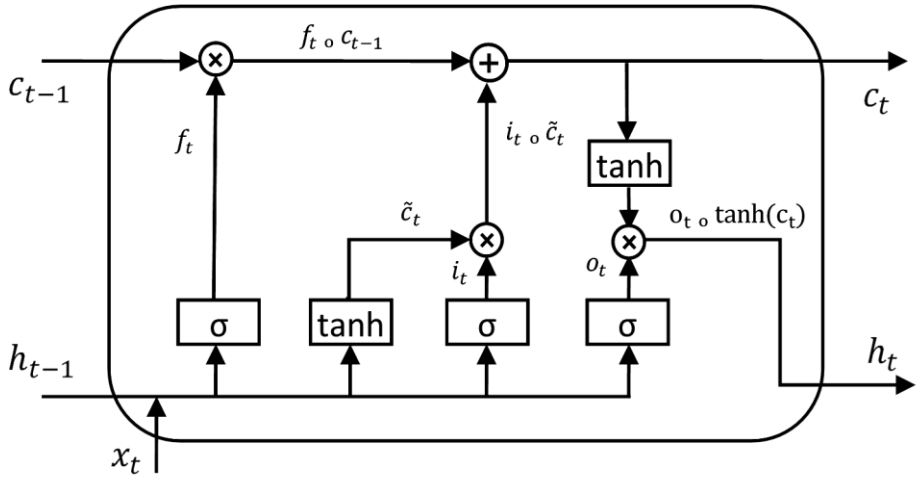


Fig 2.17: Illustration of a single cell in a LSTM layer

ii). The input gate i_t handles the new input contribution in the current cell, using a sigmoid activation function as,

$$i_t = \sigma(w_i[x_t, h_{t-1}] + b_i) \tag{2.19}$$

where w_i, b_i and σ are the associated weights, bias and activation respectively.

iii). $\tilde{c}_t$ is the short-term memory that is created using current input and previous output as,

$$\tilde{c}_t = tanh(w_c[x_t, h_{t-1}] + b_c) \tag{2.20}$$

where w_c, b_c and $tanh$ are the associated weights, bias and activation respectively. Short-term memory holds the information to store during every iteration of predictions.

iv). Multiplication of steps ii and iii results in filtered memory information (related to previous inputs), which is added to forget gate outputs (related to present inputs) to update the new memory as,

$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c}_t \tag{2.21}$$

v). Output gate *reads* the input as o_t and outputs the information relevant to the new memory as h_t,

$$o_t = \sigma(w_o[x_t, h_{t-1}] + b_0) \tag{2.22}$$

where w_o, b_o and σ are the associated weights, bias and activation function respectively.

$$h_t = o_t \circ tanh(c_t) \tag{2.23}$$

2.9 Evaluation of classification models

The evaluation of a classification is not straight forward and requires a procedure to follow. The initial and crucial step to prepare a ML or DL based classification model is data pre-processing. During this step, dataset is prepared to be fed into the model. A set of independent variables are used to predict a dependent or set of dependent variables. Libraries are used to perform specific job. Libraries and database are imported in an Integrated Development environment (IDE) using a programming language. Missing dataset values are dealt with and scaling is performed where required.

In any classification model, data is split in to training and test sets before feeding it. Models learn from the training data and assessment is performed on test datasets. The performance on test dataset should not be much different than the training dataset which might shows that the model has learned the correlation between the values and not merely the values themselves. Outcome possibilities from a classification procedure are judged using the following analytical quantities:

- True Negatives (TN) – Correct prediction, model predicts the negative class.
- True Positives (TP) – Correct prediction, model predicts the positive class.
- False Negatives (FN) – Incorrect prediction, model predicts the negative class.
- False Positive (FP) – Incorrect prediction, model predicts the positive class.

2.9.1 F-score measure

F-score is a measure to find out the usefulness of a classification procedure. It considers both precision and recall in order to calculate the score. F-score should be higher for better predictive power of the classification procedure. The lowest and highest possible values of f-score are 1 and 0 respectively where 1 represents perfect classification procedure.

$$0 \leq F \leq 1$$

F-score is the harmonic mean representation of precision and recall,

$$F = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}} \tag{2.24}$$

$$F = 2 \text{ x } \frac{Precision\ x\ Recall}{Precision + Recall} \tag{2.25}$$

where, precision is defined as the ratio between correct positive cases divided by the sum of correct and incorrect positive cases, recall is defined as the ratio between correct positive cases divided by the sum of correct and incorrect false cases and accuracy is the ratios of total correct predictions divided by sum of all predictions,

$$\text{Precession} = \frac{TP}{TP+FP} \tag{2.26}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{2.27}$$

$$\text{Accuracy} = \frac{TP+TN}{Total} \tag{2.28}$$

In this chapter we have surveyed the literature about cognitive based gait activities, gait sensing modalities and feature extraction techniques used in our multi-modality sensor fusion approach. Also, we have explored the classification models used for classification of cognitive based gait activities involved in our research.

Chapter 3

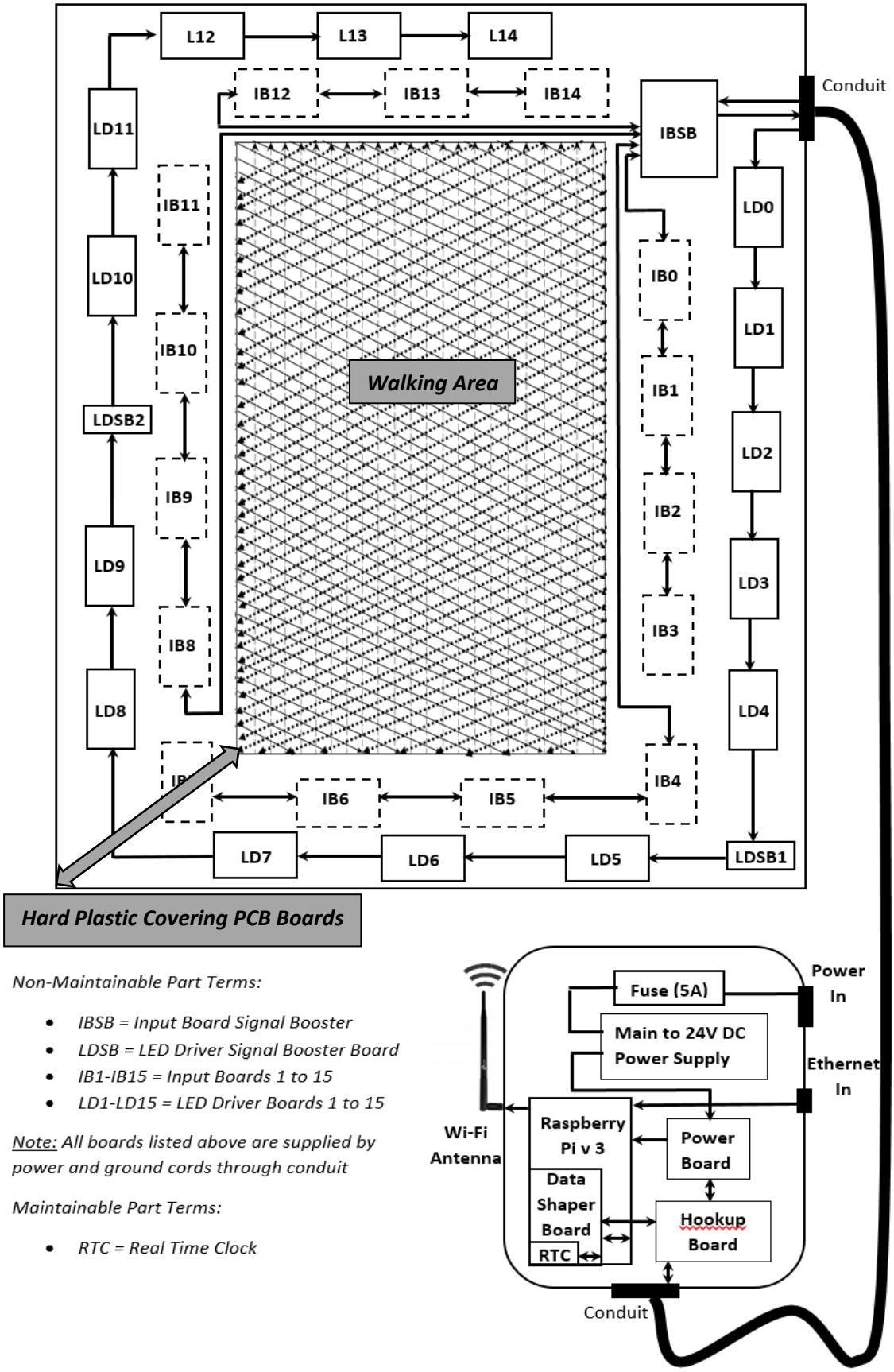
Data acquisition and analysis using gait sensor systems

Floor sensors (FS) are unobtrusive and mainly based on resistive plates, capacitive plates, piezoelectric sensors, or fiber optic cables. These systems are typically installed indoors, in controlled environments such as offices and buildings. Most FS have been employed to record physiologically defined features, such as centre of pressure, step length and cadence, rather than for collecting raw data over longer periods of time [104],[105]. Recently, inertial sensors have been actively and widely used to acquire gait information due to their small size, weight and cost [47]. An important factor to consider is that although ambulatory inertial sensors (AIS) are non-invasive, they require the individual to cooperate in wearing them on different body parts such as head, waist, chest, thigh, shank and foot to record gait signals [48].

In our research, two systems i.e., AIS and FS are used for gait acquisition and data processing. FS were already existing however AIS were developed in this research. Selection of the AIS sensors under the highlights of their measures and abilities to support our proposed research is discussed in section 3.2.1. During our experiments, user is asked to put AIS sensors on their body whilst performing gait activities on FS. Signals are collected from both systems wirelessly into servers where it is pre-processed and prepared for further processing and analysis. Details of the proposed multi-modality system are stated as follows:

3.1 Floor sensors (FS)

In this work, an original FS system (size: 2 m x 1 m approx.) is used to acquire the spatio-temporal dynamics of the ground reaction force during the chosen gait activities. FS comprises 116 plastic optical fiber (POF) sensor elements, each terminated with an LED as



Non-Maintainable Part Terms:

- *IBSB = Input Board Signal Booster*
- *LDSB = LED Driver Signal Booster Board*
- *IB1-IB15 = Input Boards 1 to 15*
- *LD1-LD15 = LED Driver Boards 1 to 15*

Note: All boards listed above are supplied by power and ground cords through conduit

Maintainable Part Terms:

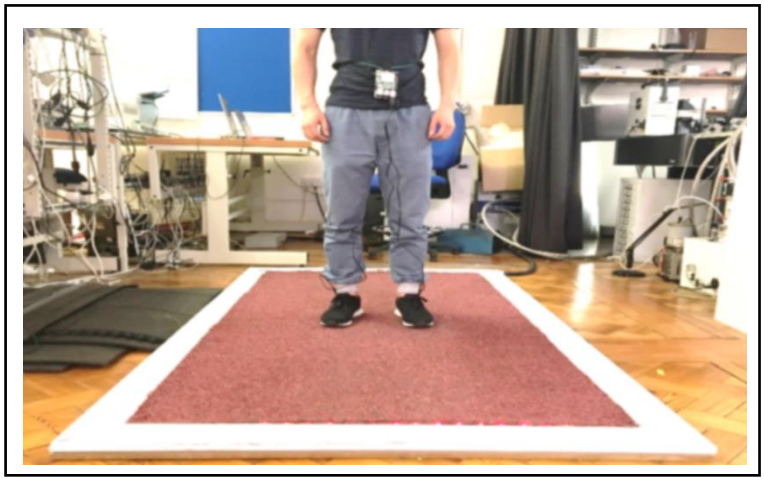
- *RTC = Real Time Clock*

Fig. 3.1 Present design of FS available at UoM

a light source and a photodiode as detector. The three-ply arrangement of POF based cables with circuit boards and wires are enclosed around the periphery of the FS and connecting with an umbilical cord to a R-Pi in a shielded box, as shown in figure 3.1. The set of POF sensors provides efficient sampling of the spatial-temporal distribution of the integrated transmission losses resulting from the applied pressure on the contact surface The R-Pi is used to transfer information to external workstation using a Wi-Fi connection.

3.1.1 Hardware implementation of FS

FS is an illustration of POF based carpet which is meant for human to manoeuvre. Plastic optic fiber (POF) based carpet was introduced by The University of Manchester (UoM), a collaboration between CEAS, EEE and NMHS. The Mk1 design made use of 256 wires to control 116 light emitting diodes (LEDs) and 116 photo detectors (PDs) coming out of the non-maintainable part of the sensor and going to an integrated circuit board with FPGA used in combination with National Instrumentation Data Acquisition Devices (NIDAQ) such as NI-9172, NI-9401, and NI-9205 for faster data acquisition and processing.



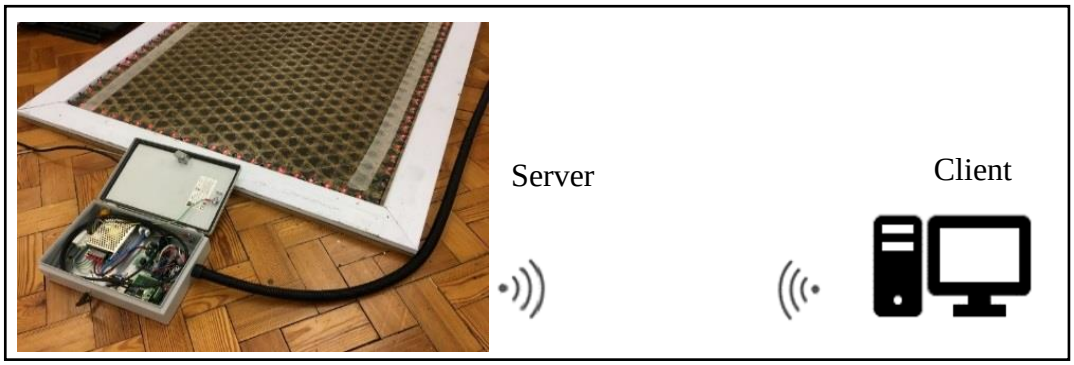


Fig 3.2: Top: User standing on the FS, Bottom: Overall connection of 116 POF (server) sensors to the outside workstation (client) through a dedicated R-Pi

To perform real time monitoring of data during acquisition needed hardware changes in PCB and upgraded expensive NI hardware and software.

To monitor the signal on real time, a comparatively cheaper and distributed approach has been implemented compared to the previous FS. The current version of FS contains POF sensors and associated electronics closed in a hard shell located at the rectangular boundaries called non-maintainable part with a total 8 wires including the power and ground wires coming out of it. This structure makes it non degradable by humans during their gait activities. Also, there is a maintainable part which is used for data filtering, processing and control. This maintainable part is connected through an umbilical to the non-maintainable part with a minimum number of wires for ease of maintenance. Both parts together make a 'server' which could be installed at any location such as hospitals and care homes. Now this server could be accessed through Ethernet or Wi-Fi interface from a remote computer called 'client' to monitor and observe gait activities, as shown in figure 3.2. Though, performance changes should be made from the maintainable part at server.

3.1.2 Software implementation of FS

The current FS has been built on Server Client based model approach. Server comprised all floor sensors connected through an umbilical cord to a 2x1 feet metal box containing a raspberry pi, CPLD and power supply connection. Client could be any computer connected through an Ethernet wire or wireless connection over internet and capable to run python IDE. Upon receiving a request of connection from client, the server would establish a secure connection with client and would start sending data frames up to 256Hz maximum. Server could also be able to save data on its memory with temporal information to allow testing and further analysis on data.

Server program for this system is developed in python language, the code is provided in Appendix B.1.2. This program is capable to perform mapping, calibration, converting raw values from binary to decimal format, data processing and filtering on the input information. Socket programming is used at server to hold information for a requesting remote client. Server must have the knowledge of IP address and port number of the remote client as this information is required to authenticate the remote client over the network.

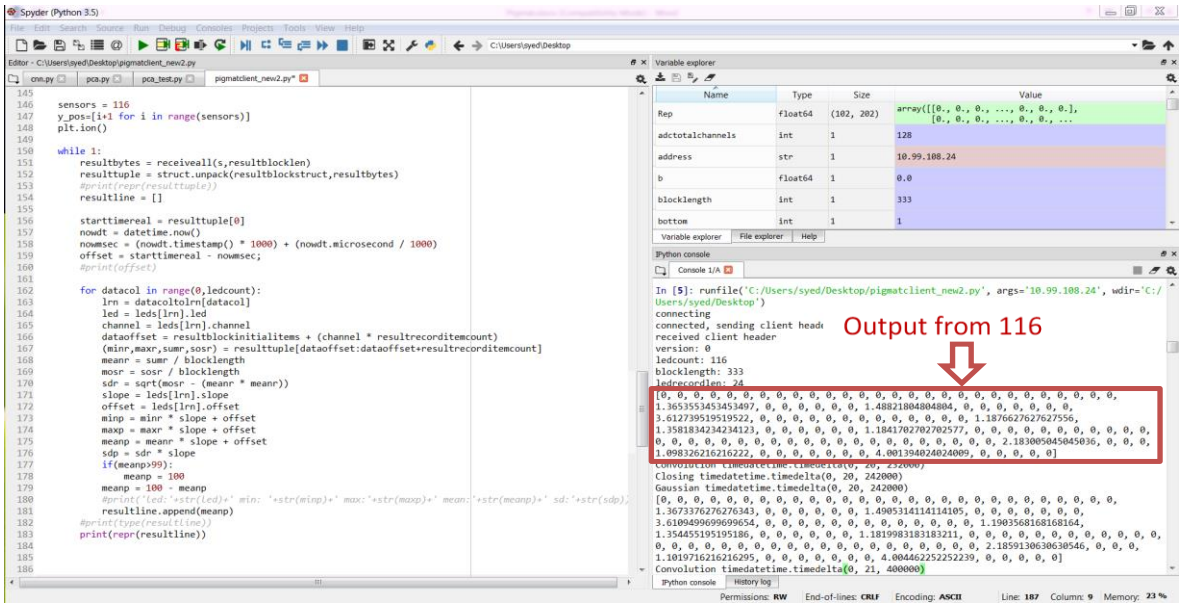


Fig 3.3: Screen shot of client application

Python is used to develop a client which could be used on any machine capable of running python IDE. Client program is required to have the knowledge of IP address and port number information of server as well. Upon successful connection, client node will be receiving up to 256 frames/second maximum from 116 FS in the form of a long string. Output frame speed per second could be altered by changing the block size of each set of information from sensors. Snapshot of client's screen receiving 116 inputs from server is shown in figure 3.3.

3.1.3 Image reconstruction of FS

In order to analyse the output on screen for better visualization and understanding by a common user, a graphical user interface (GUI) of the output has also been created. This GUI developed in MATLAB is shown in the figure 3.4. Following steps have been adapted to perform the image reconstruction on individual outputs from 116 FS:

i). **Ply arrangement** – Arrangement of individual sensor values in three plies as hardware system looks at the output as a long string of values. The FS has 116 fibers, the distribution of fibers is made in three plies as shown in figure 3.5, ply-1 comprising 22 fibers (number 1 to 22) at 90-degree angle from horizontal axis, ply-2 is diagonal containing 47 fibers (number 23 to 69) at 225-degree angle from horizontal axis and ply 3 is also diagonal including 47 fibers (number 70 to 116) at 315-degree angle from horizontal axis.

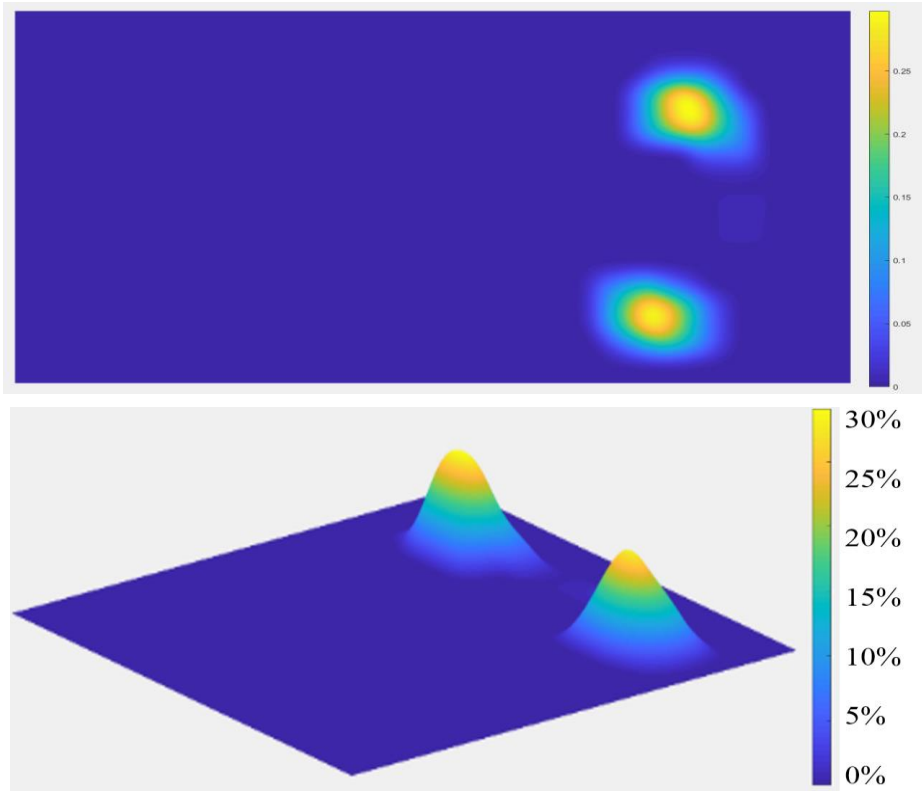


Fig 3.4: Top: top-view; Bottom: side-view displaying foot prints of the user standing on FS

Note: The legend bar represents the percentage of light stopped by exerting pressure on the POF based FS

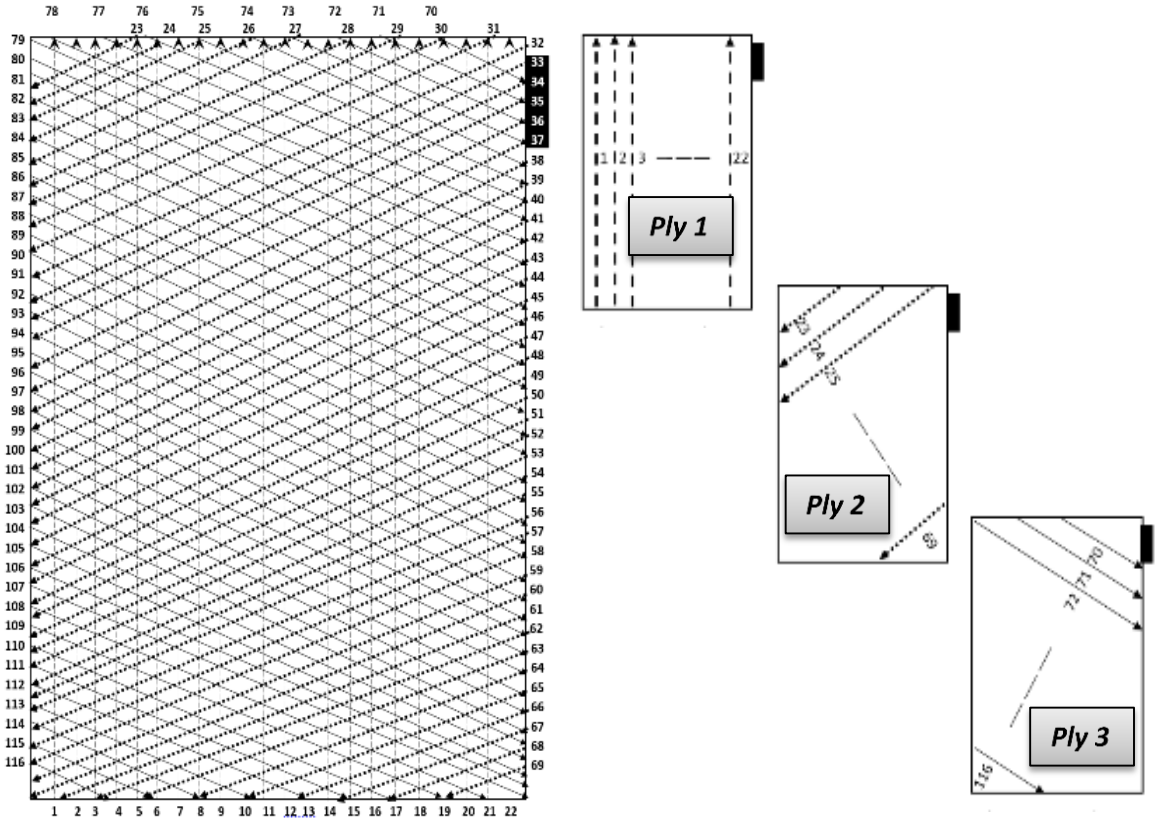


Fig 3.5: Three ply arrangement of FS

ii). **Thresholding** – Thresholding to initialize all sensor values to zero prior any disturbance caused due to the uneven surface carpet. A threshold value of 0.5% is used before starting any experiment and is used as initial threshold value for all sensors.

iii). **Image reconstruction** – Image reconstruction is performed using Landweber algorithm. As compared to other image reconstruction algorithms, the Landweber algorithm is found to converge faster than other techniques [106] and has more control through relaxation factor factor/gain **α** and number of iterations. Landweber is an iterative solver designed for an ill-posed set of linear equations. The error decreases dramatically in the start however after some iterations it requires a good stopping value of **α** to achieve optimum results. Landweber algorithm's output is given by,

$$G_{k+1} = G_k + \alpha S^T (C - SG_k) \tag{3.1}$$

where G_k is the output, α is the gain, the term $(C - SG_k)$ is used to calculate the error in which S^T is the transpose of the sensitivity matrix for area of walking on FS. Implementation of landweber algorithm is mentioned in appendix B.1.3.

iv). **Convolution** – Convolution filtering is implemented using a mask 'M' (a vector of ones, size 3x3) to filter out the pixels having more than the seven noisy pixels from every image as shown in figure 3.6. This procedure is adapted to remove the artifacts caused due to POF based sensor arrangement in three plies for FS. Left figure in figure 3.6 illustrates a re-constructed image before convolution operation while the right figure illustrates the same image after-convolution filtering (code is provided in Appendix B.1.4).

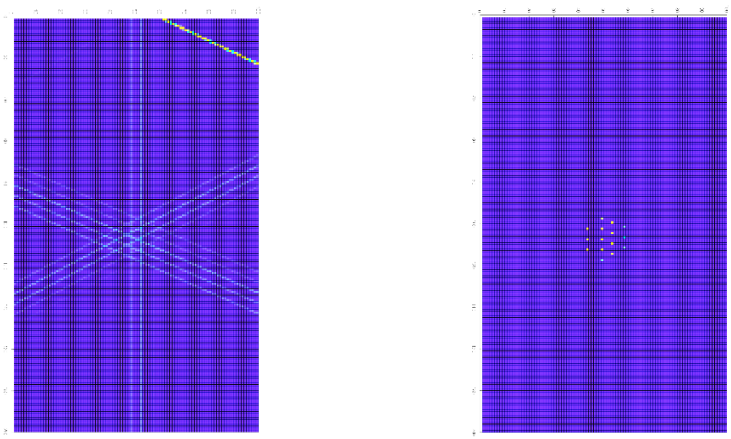


Fig 3.6: Left: Original reconstructed image, Right: Image after convolution filtering

v). **Image segmentation** - ‘Closing’, an image segmentation technique to fill out the closed area contouring the intact areas on carpet. Effects of image segmentation followed by convolution operation, can be seen in figure 3.7.

vi). **Smooth filtering** – Further, Gaussian filter of size 2x2 is used to smooth the boundaries of segmented image and boost the central uniform density values such that more pressurized areas on carpet will appear as high peak signals. This would also help in measuring the difference between peaks of different signals. The effect of smoothing filter after image segmentation is shown in figure 3.7.

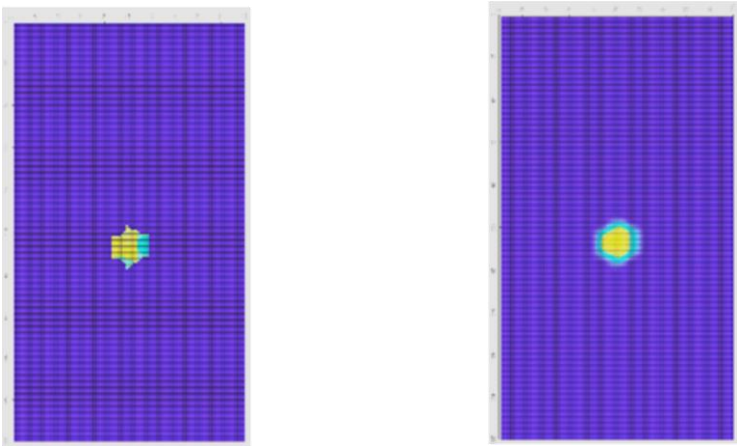


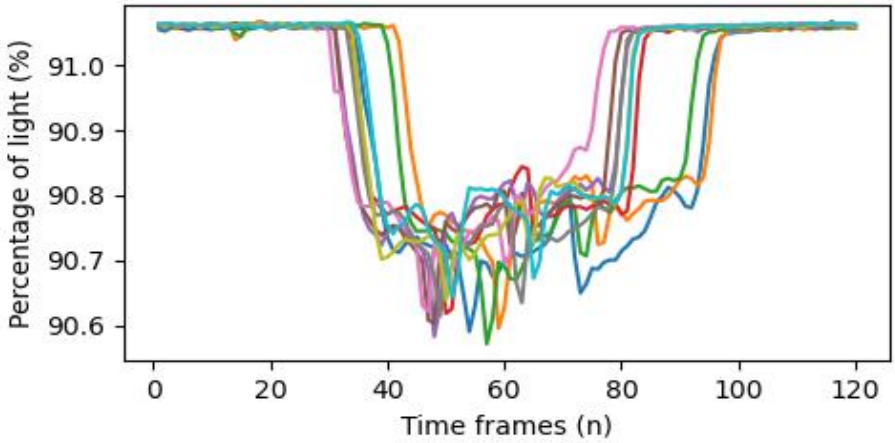
Fig. 3.7 Application of left: image segmentation, Right: smooth filtering

3.1.4 Data acquisition and pre-processing

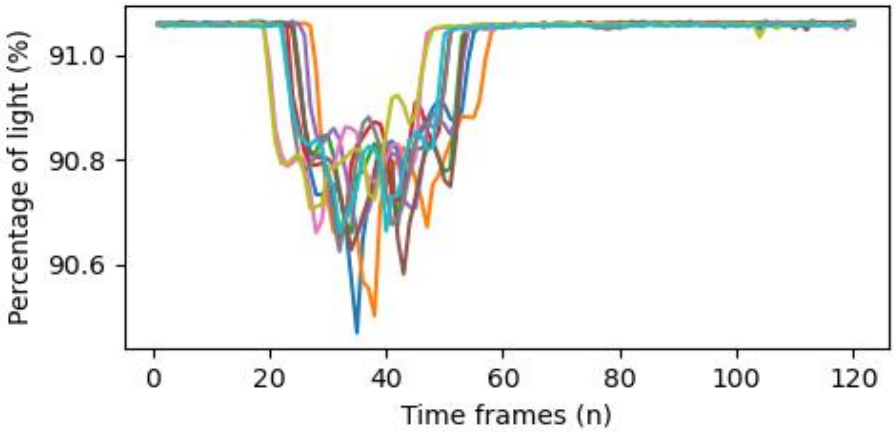
Data obtained from FS is a string of values output from 12bit ADC converter at every timestamp. These strings of information are processed and converted into transmitted light percentages. FS is synchronized at the same frequency of 20Hz used with AIS. The spatial average (SA) of the spatio-temporal information obtained from FS from different gait activities is given by,

$$SA[t] = \frac{1}{n}\sum_{k=1}^{n}(F_k[t]) \tag{3.2}$$

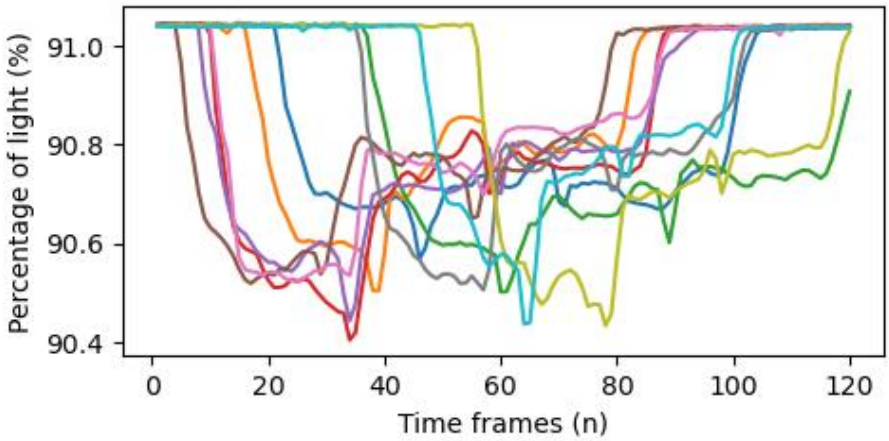
where n is the total number of sensor inputs (n=116) and F_k is the input signal at every time instance t. A set of 10-SA signals for each gait activity are shown in figure 3.8 (a)-(g). It is noticeable that some of the SA signal (in same gait activity) have a delay which represent the time delay before initializing the gait activity by user. However, each user is allocated 6 seconds window to complete one gait activity.


Percentage of light (%)
91.0
90.9
90.8
90.7
90.6
0
20
40
60
80
100
120
Time frames (n)

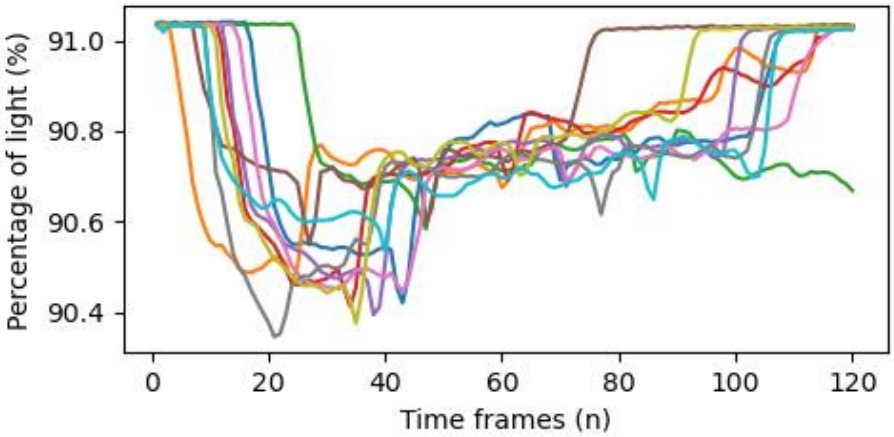
(a)


Percentage of light (%)
91.0
90.8
90.6
0
20
40
60
80
100
120
Time frames (n)

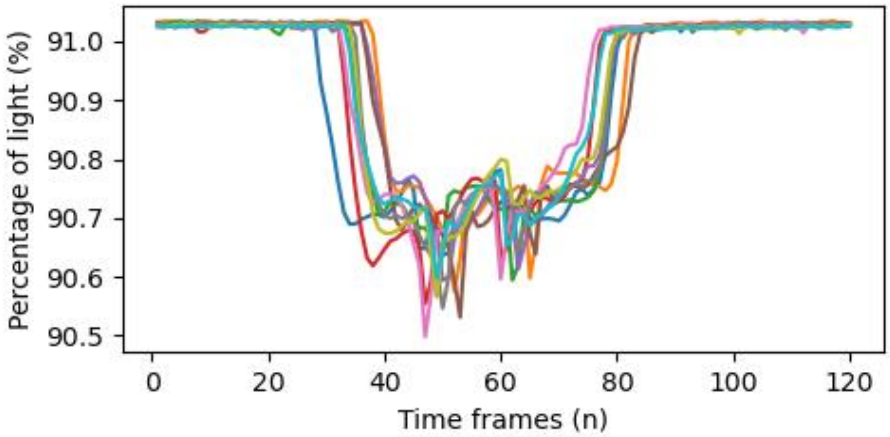
(b)


Percentage of light (%)
91.0
90.8
90.6
90.4
0
20
40
60
80
100
120
Time frames (n)

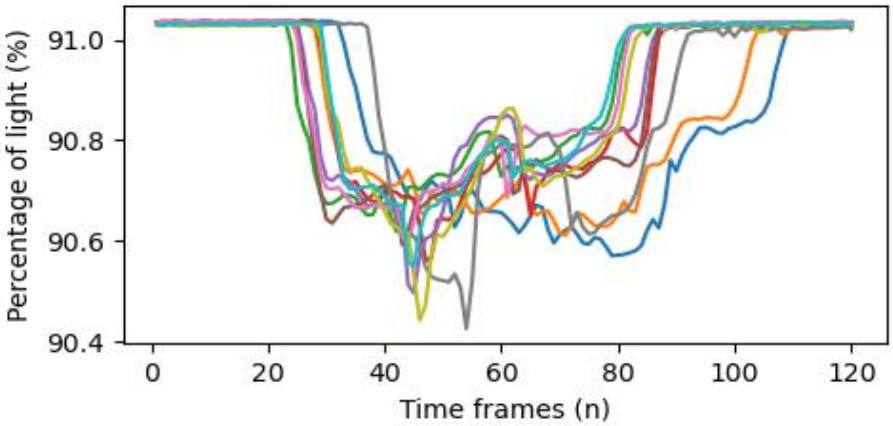
(c)


Percentage of light (%)
91.0
90.8
90.6
90.4
0
20
40
60
80
100
120
Time frames (n)

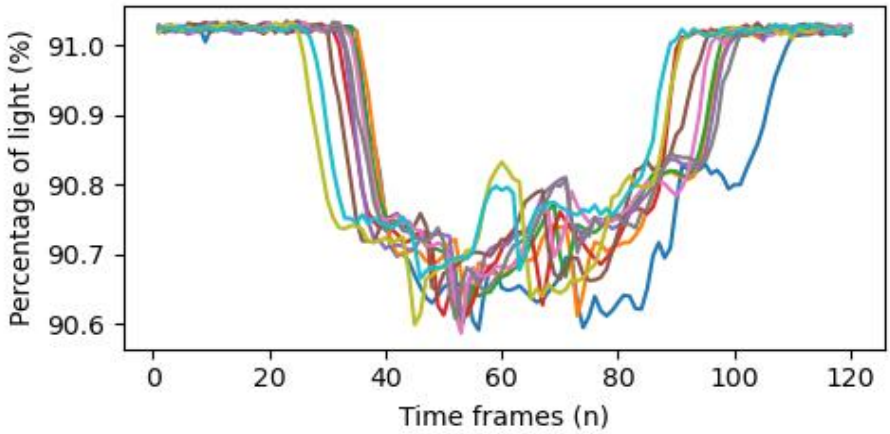
(d)



(e)



(f)



(g)

Fig 3.8: FS results of a user performing 10 gait cycles for 7 gait activities i.e., (a) normal walk, (b) fast walk, walking whilst (c) subtracting number 3, (d) subtracting number 7, (e) listening, (f) typing on mobile (g) talking whilst walking

3.1.5 Relationship between FS and gait activities

To observe the response from 116 sensors during each gait activity, the spatial average is calculated for the data related to seven gait activities as shown in figure 3.8. The activities including comparatively higher cognitive load such as subtracting-3, subtracting-7 and walking while typing on the mobile are spread across 120 times frame (representing one gait activity experiment) as compared to the activities involving lower cognitive loads such as normal walk, walking while listening, walking while talking and fast walk which consume lesser time. It was observed that higher cognitive load activities slow down the walking process and tend to have more foot intact than the lower cognitive ones which can be seen in the form of signal peeks (representing heel strikes on the FS carpet area). The second peak signal represents the heel strike of the second foot on FS and signal drops its intensity as the weight of the user gets distributed on the other foot during gait cycle.

3.2 Ambulatory inertial sensors (AIS)

The utility of inertial sensors to monitor and classify human activities, and gait in particular, is well established [26]. Inertial sensors require attachment on body to capture information from different body locations during human gait activity. A portable AIS system has been developed and deployed to study the effect of gait on the movements in the lower half of human body. The AIS system comprises: (i) a Raspberry-PI (R-Pi) (ii) Sense-HAT board, (iii) two 9DoF Razor IMUs.

3.2.1 Hardware components of AIS

Initially, Arduino microcontroller[107] is used in this research which does not need a separate operating system like Raspberry-Pi (R-Pi). However, Arduino needs an extra connection of Wi-Fi to transfer real-time data to the processing machine which would result in additional hardware attached to the user. Therefore R-Pi is selected which is a credit card sized computer, with a quad core 1.4GHz processor, 1GB RAM, Bluetooth and built-in Wi-Fi[108]. Keyboard, Mouse, plug and play USB devices, HDMI display, power supply and additional hardware can be attached with raspberry pi with ease. Third version and B+ model of R-Pi is used which can be seen in figure 3.9 (the latest versions among all models or R-Pi at the time of proposed study).

Sense-HAT[109] with a built-in 3D accelerometer (+/-16g) and a gyroscope (+/-2000dps), is an add-on board which fits on the top of R-Pi. Sense-HAT is selected due to its smart design and suitability to attach with user pelvis during experiments.

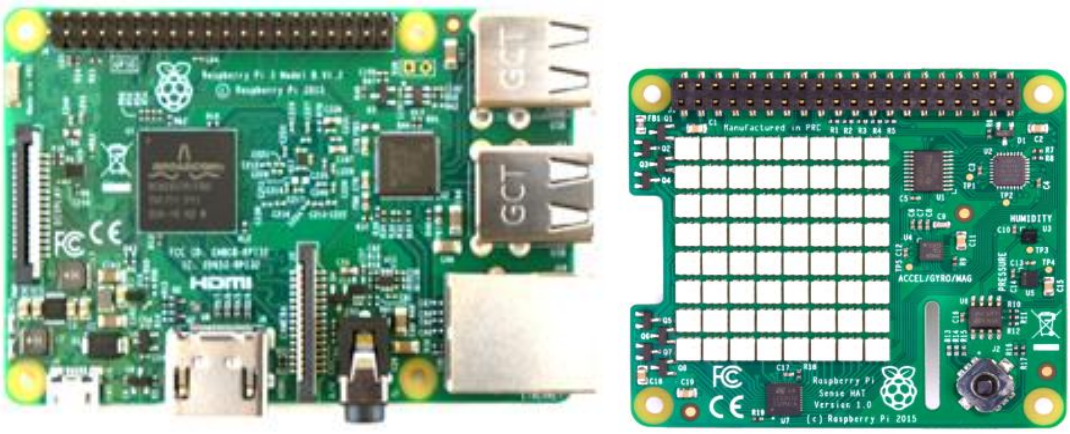


Fig 3.9: Raspberry-Pi (ver: III, model: B+) (left), Sense-HAT Board for R-Pi (right)

Two 9DoF Razor IMU M0[110] sensors (as shown in figure 3.10) are used in this work. These sensors are designed to work either with a USB power cable or a Lithium polymer battery (3.7-4.2V). These sensors have been selected due to their compact size (3x3x0.1cm) and suitability to attach with user heels with/without wires during experiments. Each sensor has the following specifications:

- 3 axis accelerometer (+/-16g) and a gyroscope (+/-2000dps)
- Integrated Atmel SAMD21cortex-M0+ microprocessor (32 bit) to filter and pre-process data during acquisition
- Programmability through a dedicated USB connection
- LiPo battery charger connection for wireless charging
- MicroSD card socket for data recording whilst attached on the user

Fig. 3.10 9DoF Razor IMU M0 front view (left) and back view (right)

3.2.2 Design and build

The R-Pi with the Sense-HAT board (attached at the top) with a 2000-mAh portable battery bank (attached at the bottom) is called 'Sensor 1' which is connected through USB cables to both 9DoF Razor IMUs called 'Sensors 2 & 3' as shown in figure 3.11. Further, AIS is connected to a workstation for data transfer and control through a Wi-Fi connection. Concerning the AIS system, sensors 1-3 (highlighted yellow) are attached to the pelvis and both heels of the user, to capture pelvic and foot motions during gait activities. The feet, being furthest from the centre of mass, are the obvious fastest movers during gate phases; the pelvic motion engages in body weight shifting, as part of natural gait patterns.

Different number of sensors have been reported to capture gait activities in literature [48]. However, deploying the minimum number of sensors may result in performance bottle necks whilst recording the complex gait activities [111]. The sensor positioning and number of sensors attached to the human body are also important factors whilst judging the quality of extracted data. Panebianco et al. [51], reported accuracies using 17 algorithms on 5 IMUs placed on back (1 IMU), shanks (2 IMUs) and feet (2 IMUs). To estimate the stance time, results obtained from the acceleration values of shank and foot performed better than the lower trunk. However, angular velocity estimation performed better in the detection of toe off and heel-strike events, with noticeable dependencies on sensor position.

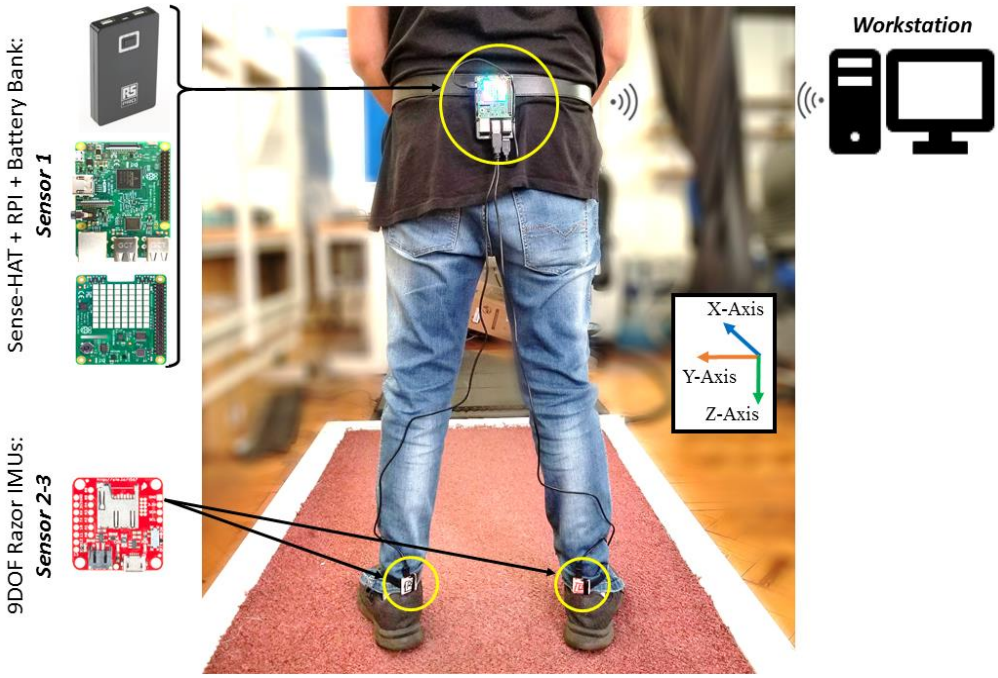


Fig 3.11 AIS placed on the user, comprising the Sense-HAT board attached to the R-Pi powered by a portable battery bank (sensor 1) and 9DOF Razor IMUs (sensors 2&3) connected through USB cables to RPI.

3.2.3 Data acquisition and pre-processing

From AIS, raw data on acceleration and angular velocity values is obtained from sensors 1-3, a normal walking gait activity of a subject using 3 sensors is shown in figure 3.12.

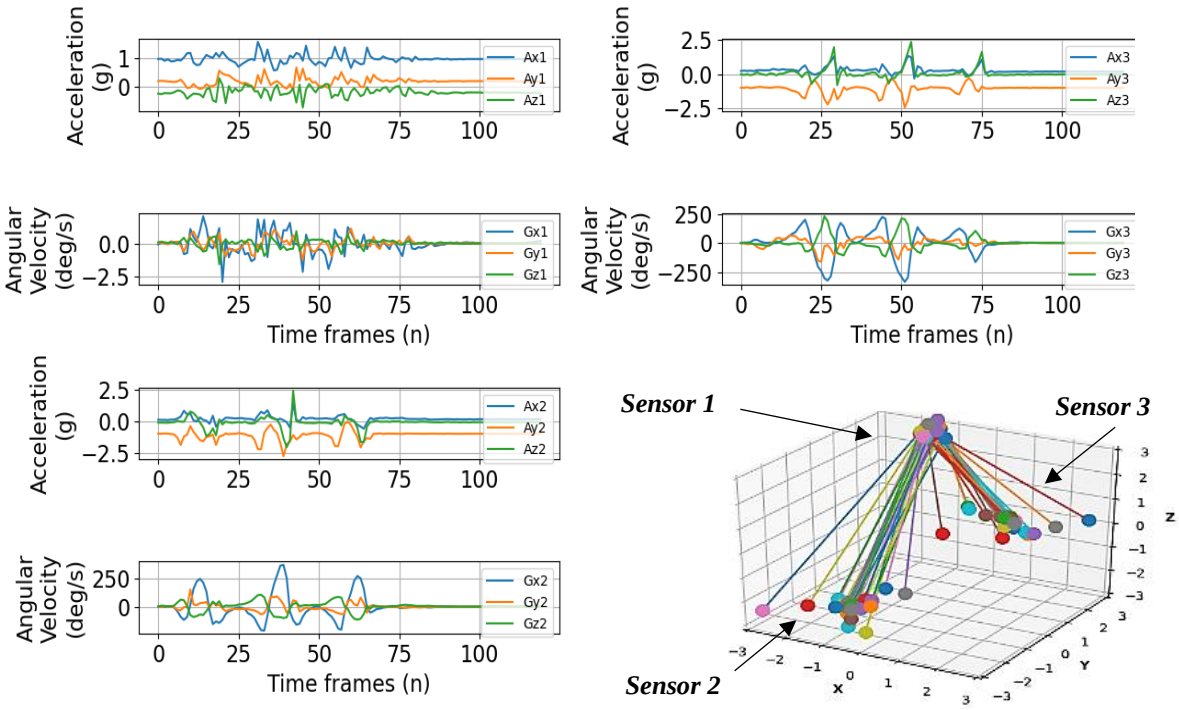


Fig. 3.12 Accleration and angular velocity values from three sensors attached to a subject performing normal gait pattern; Bottom right: Test program.

The default sampling frequency of sensor 1 is 30Hz while sensor 2 and sensor 3 are sampled at 100Hz. After filtering and re-sampling, the spatio-temporal information from all three sensors of AIS is synchronized at 20Hz. Raw acceleration values are in two's compliment format; therefore, these values are converted into values between +16g and -16g (where 1g = 9.8m/s^2). Synchronized data acquired from AIS is shown in figure 3.13. ACL/X, ACL/Y and ACL/Z are the accelerations in x, y and z-axis from sensor 1 (prefix: HAT), sensor 2 (prefix: IMU1) and sensor3 (prefix: IMU2) respectively. Similarly, GYR/X, GYR/Y and GYR/Z are the angles in x, y and z-axis from sensor 1 (prefix: HAT), sensor 2 (prefix: IMU1) and sensor3 (prefix: IMU2) respectively. To calculate the angle (θ) from raw angular velocity (ω) values, the following formula is used:

$$\theta = \omega \,.\, \Delta t + \theta \tag{3.3}$$

TIME	HAT-ACL/X	HAT-ACL/Y	HAT-ACL/Z	HAT-GYR/X	HAT-GYR/Y	HAT-GYR/Z	IMU1-ACL/X	IMU1-ACL/Y	IMU1-ACL/Z	IMU1-GYR/X	IMU1-GYR/Y	IMU1-GYR/Z	IMU2-ACL/X	IMU2-ACL/Y	IMU2-ACL/Z	IMU2-GYR/X	IMU2-GYR/Y	IMU2-GYR/Z
1530722364096.00	0.03	0.18	1.26	1	-0.25	-0.13	-0.02	-0.03	1.02	-2.26	1.28	0.18	0.15	-0.33	0.92	0.55	3.23	0.49
1530722364148.00	-0.03	0.15	1.35	0.62	-0.34	-0.32	-0.05	-0.11	1.08	-2.2	1.16	0.18	2	1.39	1.16	-9.15	17.87	4.57
1530722364200.00	-0.07	0.05	1.13	0.22	0.13	-0.18	-0.13	0.41	0.96	-0.55	2.26	1.65	1.74	0.95	1.28	-2.01	70.61	185.73
1530722364252.00	0.1	0.16	0.81	0.91	0.19	0.51	-0.09	0.24	1.09	10.37	6.28	44.57	0.48	0.5	1.44	31.65	80.61	491.46
1530722364304.00	0.21	0.33	0.46	1.15	0.31	0.49	-2	-1.37	1.59	12.5	-2.01	112.32	-0.95	-1.13	1.96	-4.94	67.5	574.7
1530722364356.00	0.18	0.48	0.32	-0.15	-0.73	0.18	-2	1.28	0.99	-64.45	22.93	-54.45	-1.57	-1.01	0.58	-67.68	8.78	489.88
1530722364408.00	-0.06	0.57	0.82	-0.16	-0.79	-0.39	-0.4	0.49	-0.71	9.82	20.49	-90.49	-1.47	-1.95	1.93	-139.7	-13.11	352.62
1530722364460.00	0.19	0.36	1.25	1.03	-0.72	-0.04	0.76	0.89	0.03	48.05	14.45	15.43	-0.37	-0.61	-0.69	-155.49	54.94	88.78
1530722364512.00	0.1	0.42	1.44	0.22	0.07	-0.62	0.69	0.71	2	-30.85	12.99	94.27	-1.34	-1.95	0.09	12.74	48.78	-54.15
1530722364564.00	0.2	0.17	1.29	0.66	0.08	-0.18	2	0.26	-0.42	-82.8	2.01	205.98	0.21	-0.52	-2	44.02	110.61	-185.61
1530722364616.00	0.17	0.16	0.96	0.54	0.13	0.17	-0.31	-0.77	2	-99.7	-17.93	227.2	-0.55	-1.16	-0.62	-69.76	107.99	-85.79
1530722364668.00	0.27	0.17	0.59	0.2	0.54	0.27	-0.03	-1.66	-1.14	-158.48	-25.12	167.38	-0.49	1.77	2	-39.33	37.62	-194.02
1530722364720.00	0.17	0.35	0.34	0.1	0.15	0.68	-0.01	-0.12	0.9	-10.06	-35.73	131.95	-0.21	-0.48	1.35	-60.98	-16.46	18.23
1530722364772.00	0.16	0.55	0.44	-0.93	-0.79	0.27	0.67	0.18	1.03	12.07	-43.66	151.65	0.83	0.11	1.4	-55.43	-36.65	135.61
1530722364824.00	0.08	0.62	0.92	-1.1	-0.49	-0.45	1.33	0.07	0.48	-0.06	-33.11	180.73	0.79	-0.42	0.3	-38.84	-35.98	148.41
1530722364876.00	0.07	0.66	1.13	0.36	-0.21	-0.64	0.34	-0.16	0.43	-16.77	-7.74	222.93	0.23	-0.39	1.37	-1.46	15.06	175.06
1530722364928.00	0.19	0.45	1.34	0.43	-0.12	-0.36	0.27	-0.99	-0.14	-91.59	81.59	245.06	-0.33	-0.96	0.15	59.45	148.54	173.66
1530722364980.00	0.17	0.35	1.32	0.43	0.11	-0.36	-0.81	-1.56	2	-153.11	219.94	153.66	-0.11	-0.94	-2	-12.68	300.43	95.43
1530722365032.00	0.29	0.09	1.07	0.37	0.05	-0.07	-1.61	0.68	0.83	285.79	276.59	55.98	-0.56	0.32	0.22	-152.99	341.52	4.27

Fig. 3.13 Data-frame comprising 18 outputs obtained from AIS

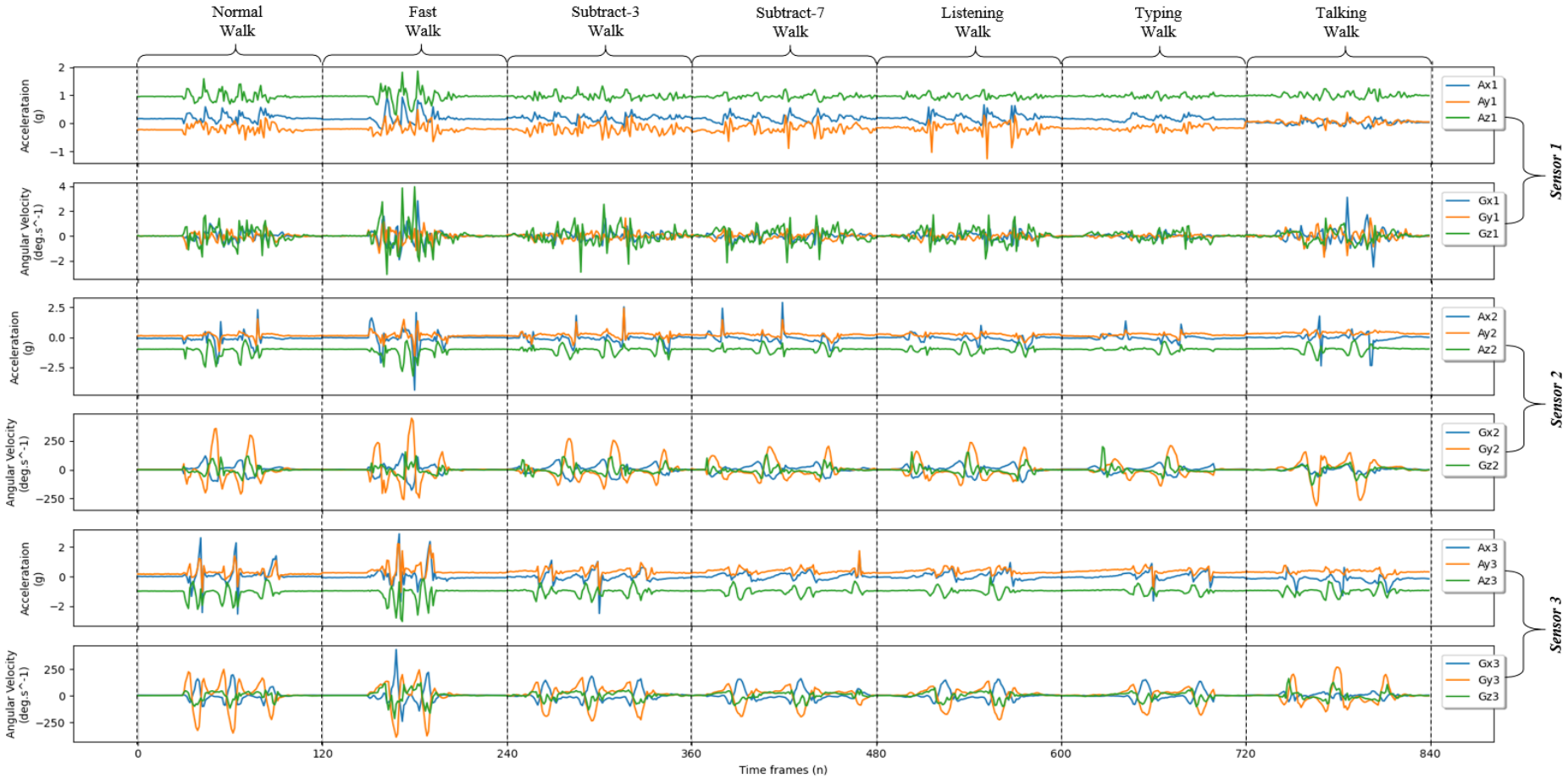
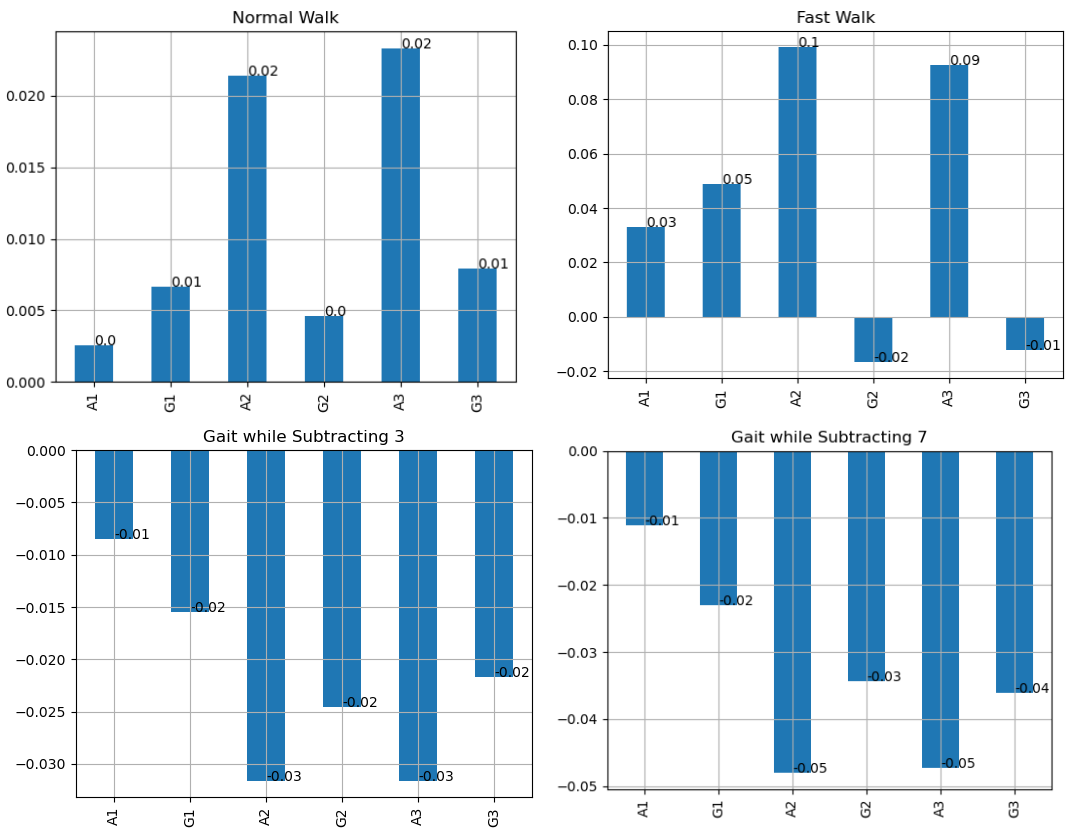


Fig 3.14: Acceleration and angular velocity values from AIS for 7 gait activities i.e., Normal walk, fast walk, dual tasks: subtracting number 3, number 7, listening, typing on mobile and talking whilst walking.

where Δt is the time step. Data acquisition of acceleration and angular velocity values from three AIS for seven gait activities is shown in figure 3.14. The nature of each experiment requires subjects to start walking from one end of the FS to the other in forward direction whilst wearing the AIS. Therefore, all IMUs are aligned so that the highest acceleration (in forward direction) is represented by X-axis; weaker acceleration (vertical, in up/down direction between heel strike and toe off events of each foot) is represented by Z-axis; the weakest acceleration (lateral, in left/right direction) is represented by Y axis.

3.2.4 Relationship between AIS and gait activities

A suitable approach to compare all gait activities across the acceleration and angular velocity values is the correlation bars which can be seen in figure 3.15. A1-A3 and G1-G3 on horizontal axis represent acceleration and angular velocity values obtained from sensors1-3, vertical axis represents correlation factor with maximum varience of ±1 (+1 representing maximum positive, 0 means nill and -1 means maximum negative correlation between the selected value and the related gait activity). Root of squared acceleration and angular velocity values in all three direction is used to calculate a combined response from each



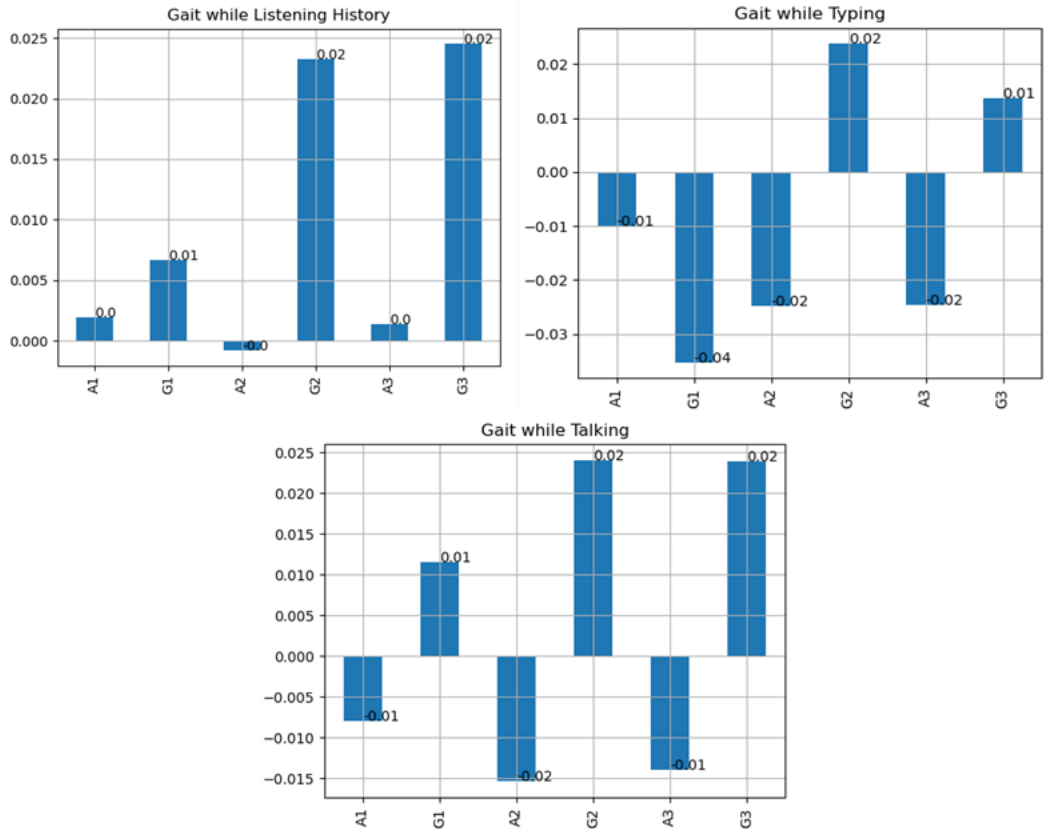


Fig 3.15: Correlation bar plot using maximum acceleration and angular velocity values of all gait activities from AIS dataset.

sensor of AIS. It can be seen that gait activities with higher cognitive load have an overall negative correlation as compared to lower cognitive load have an overall positive correlation with the input values. Higher cognitive load tends to negatively correlate with the acceleration and the angular velocity values which is more noticeable in case of 'Fast Walk' with maximum positive correlation of 0.1. However, a mixed response can be seen for 'Gait while Talking activiy'.

3.3 Visual gait analysis

The full gait cycle can be represented (see figure 1 in [9]) as 5 events in the stance phase, starting with a heel strike (HS) and finishing with the HS of the opposite foot. Some of the gait events are possible to identify by visual inspection of data obtained separately from both modalities. For illustration purposes, in figure 3.16 representing a normal gait activity where only HS is indicated on the FS signals (as the mean of all 116 sensor outputs) and the AIS signals (as the root of squared maximum accelerations in all 3 directions) given

by,

$$A_{max} = \sqrt[2]{A_x^2 + A_y^2 + A_z^2} \tag{3.4}$$

The notable HS dip at the dashed lines is alternating between the two legs: HS1 and HS3 from sensor 3 and HS2 and HS4 from sensor 2. Sensors 1 is not sensitive to HS as expected because of its position close to centre of mass and not on the limbs.

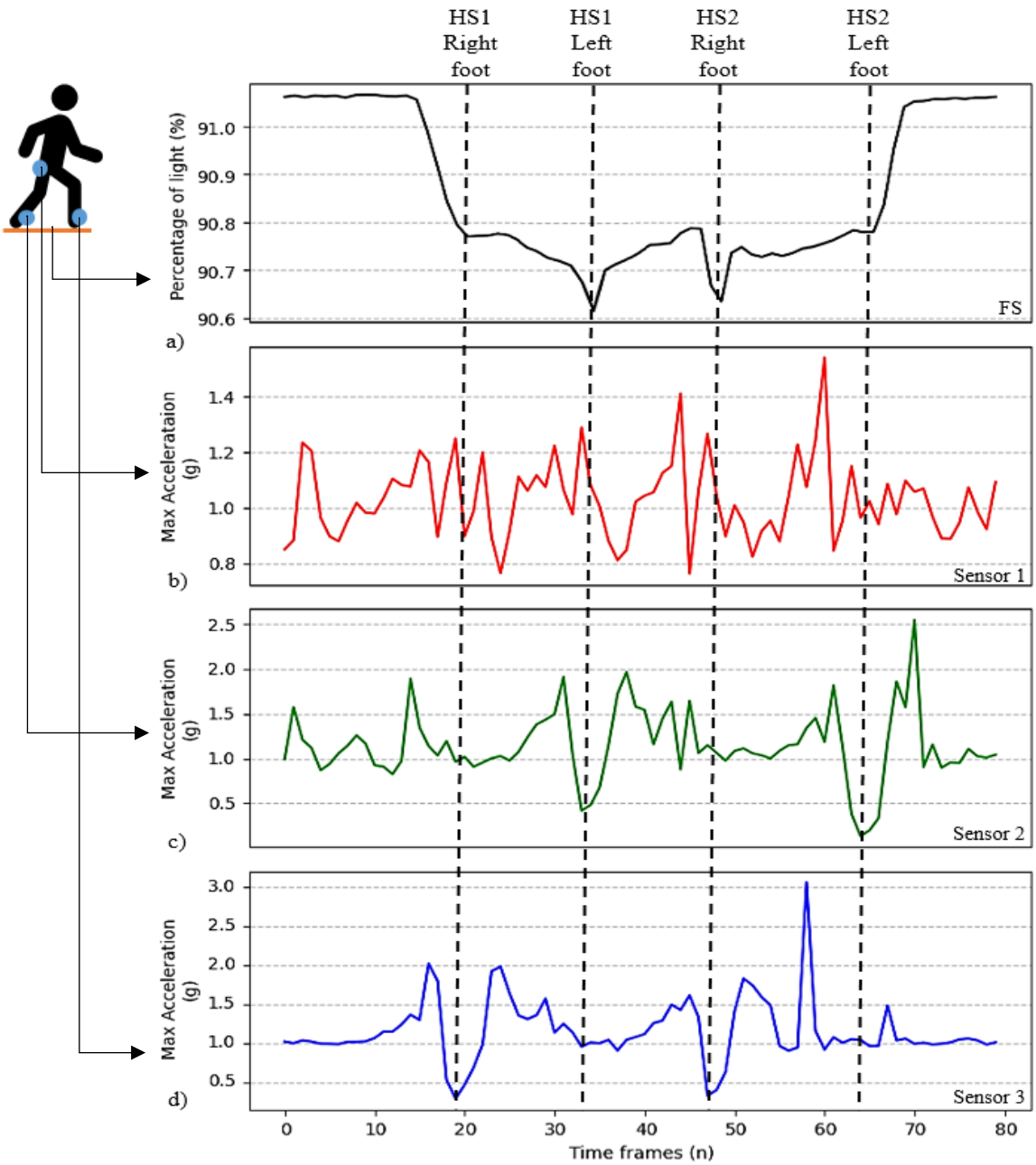


Fig 3.16: (a) Mean of 116 values from FS; (b),(c),(d) Root sum of maximum accelerations from AIS sensors1/2/3(placed on pelvis/left-foot/right-foot respectively) vs time frames for a normal walk gait pattern.

Note: Mean of 116 FS values represents the overall disturbance in light channel from 100% and maximum acceleration represents maximum movement for every sensor measured in g where 1g = 9.8m/s²

3.4 Experiments and data acquisition

An ethical request for the proposed research on healthy subjects using FS and AIS has been approved from Manchester University Research Ethics Committee (MUREC). Prior to experiments, written consent from each volunteer's was obtained prior to all experiments and research was conducted in accordance with the general guidelines of ethics board. However, the data acquisition programme has been substantially affected by the pandemic, as elaborated in COVID-19 statement.

Data was collected from 11 healthy volunteers; volunteer profile is shown in table 3.1. Volunteers, wearing AIS while walking on FS, performed all the activities. Serial subtraction of number 7 from a random starting number, listening to a story, texting on a mobile device and talking to the operator, are the cognitive load-based activities involved in our research and each activity is recorded 10 times for every volunteer. From each of the 3 AIS sensors 3 axis acceleration and 3 angle values were collected yielding a set of 18 values.

From FS, 116 values were used after calibration. These values from both modalities are further extracted for the unique gait features as described in Chapter 4 and 5.

User	Weight(kg)	Height(cm)	Gender	Age(year)
1	71	160	Female	31
2	68	175	Female	29
3	57	162	Male	48
4	80	186	Male	24
5	69	170	Female	40
6	75	185	Male	31
7	79	176	Female	21
8	65	171	Male	23
9	70	173	Male	29
10	96	177	Male	33
11	53	168	Female	23

Table 3.1: Subject Profile

3.5 Data pre-processing

Python is used as programming languages which provides complete support through its libraries for data pre-processing, feature extraction, feature selection, feature fusion, and final classification. Data pre-processing is used to make data ready before feeding it to any ML and DL model for classification and involves the following steps:

- **Importing libraries** – Libraries are the predefined functions that tell the IDE to perform certain functions. For example, in python, the NumPy library helps to use mathematical tools; matplotlib library helps to draw charts and pandas library helps to import datasets and manage datasets. We have utilized python libraries such as Keras, TensorFlow, Pandas, NumPy and Matplotlib in this research work.

- **Importing dataset** – Data needs to be imported in the form of matrices. Data could also be divided in sub-matrices containing independent values (inputs) and dependent values (outputs).

- **Categorical dataset** – Data containing values as categories like names of gait activities is required to convert into numerical values. Categories could not be fed directly into strong mathematical equations used by ML algorithms. Therefore, these categorical values need to be decoded into numeric values. As we are dealing with seven different gait activities therefore these categories are converted into numerical values for supervised machine learning procedures involved in our research.

- **Splitting dataset into the training set and test set** – Any dataset could be split into training set and test set. ML algorithms make use of correlation between values of dataset called the training set and build a model which is tested on slightly different dataset called the test set. Performance of ML model on test set is not different than the performance obtained from the training set which means that the model is quite flexible; it has understood the correlation between the values and not learned the training set values by heart. We have implemented 80%-20% split [112] for training-testing on our original dataset.

- **Feature scaling** – Some time data values in different column or rows need scaling otherwise it would cause issues in machine learning models. Euclidean distance is a common measure used for feature scaling. Euclidian distance two points $P(x_1, y_1)$ & $P(x_2, y_2)$ is given by,

$$d = \sqrt[2]{(x_2 - x_1)^2 - (y_2 - y_1)^2} \tag{3.5}$$

Other methods of scaling are listed as:

Standardization: In this technique the values are centred around the mean with standard deviation. So, the mean of attribute turns zero and results in a unit standard deviation.

$$x_{stand} = \frac{x - mean(x)}{standard\ deviation(x)} \tag{3.6}$$

Normalisation*:* The values are scaled between 0 and 1 without destroying differences in the ranges of values or loosing information.

$$x_{norm} = \frac{x - min(x)}{max(x) - \min(x)} \tag{3.7}$$

Any method could be used for scaling, however no variable should be dominated by any other variable. We have implemented standardization in our research before handing our data to any ML model. Another main advantage of feature scaling is that ML like DT algorithm converge much faster if the data is scaled properly and take longer time otherwise.

In this chapter we have described the hardware and software implementation of FS and the designed AIS system used in our research. We have also reported the data acquisition and pre-processing methods used for both modalities. Further, we have explored the relationship between gait sensor data and the related gait activities.

Chapter 4

Feature level based multi-modality sensor fusion

It has been suggested by various studies that the fusion of sensory information can take place at a) data level b) feature level and c) decision level [113]. In data level fusion, the raw data obtained for a measured object using sensors is combined directly. Fusion of information at data level contains maximum information therefore it is expected to produce better results. However, data level fusion aims to combine similar nature sources of sensory data. Data level fusion has limited actual implications as various physical quantities can be calculated from a more comprehensive analysis.

In feature level fusion, features are extracted from the data collected using multiple sensors which are combined and utilized in a special classification model for final decision making. In human gait classification, extracted features include metrics such as mean, variance, standard deviation, energy, entropy, signal amplitude, root mean square, percentiles, as well as characteristics in the frequency domain available through Fourier transform, discrete cosine transform, spectral entropy and energy [114]. Typically, these undergo a further feature selection process involving e.g. the windowing method, kernel discriminant analysis , minimum redundancy, maximum relevance and correlation analysis [115].

Decision level fusion involves the data processing techniques such as feature extraction and pattern recognition implemented on the data followed by the production of a decision vectors used in decision level techniques such as Bayesian theory [116], Dempster-Shafer evidence theory [117] and behavior knowledge space [118].

4.1 Feature extraction

Feature extraction is used for dimension reduction mainly used in case of larger data sets in which initial set of raw data is broken into more manageable groups for processing. Larger datasets require more computational resources, feature extraction helps to effectively reduce the number of inputs into features whilst keeping the accuracy and the complete description of the original dataset. However, in most cases throwing away of data is not recommended as we could lose part of some meaningful information.

Analysis using large number of variables also requires more memory and computational power. Therefore, feature extraction is very effective in removing the redundant and irrelevant data values whilst improving learning accuracies and increasing result comprehensions [119]. A number of features based on statistical or model-based approaches are available; some are listed as follows:

- Minimum
- Sum of squares (SOS)
- Variance
- Root Mean Square (RMS)
- Principal Component Analysis (PCA)
- Maximum
- Mean
- Standard Deviation
- Fast Fourier Transform (FFT)
- Linear Discriminant Analysis (LDA)
- Sum
- Median
- Entropy
- Discrete Wavelet Transform (DWT)
- Canonical Correlation Analysis (CCA)

These features are extracted from the original dataset and serve as inputs to the classification algorithms. These features can also be scaled or normalized depending on the user requirements. In this research, Principal Component Analysis (PCA) and Canonical Correlation Analysis (CCA) are proposed as feature extraction methods for the data obtained from FS and AIS, as explained below:

4.1.1 PCA methodology

Larger datasets suffer from multi-collinearity problem among the independent variables of a dataset which could result in increasing standard errors of the parameter estimate [120]. PCA has been used extensively to extract the uncorrelated linear composites of independent variables into $\boldsymbol{p}$ feature vectors, where $\boldsymbol{p}$ equals the number of original input

variables in a dataset. The first feature vector accounts for the maximum variance among input variables of a given dataset and each subsequent feature vector accounts for the maximum variance which has not been accounted by the previous vector. All new feature vectors are uncorrelated to each other. The original set of p independent variables $\boldsymbol{X} = [X_1, X_2, \ldots, X_p]$ can be described in terms of set of p feature vectors $\boldsymbol{\xi} = [\xi_1, \xi_2, \ldots, \xi_p]$ as follows:

$$\xi_1 = w_{11}X_1 + w_{12}X_2 + \ldots + w_{1p}X_p$$

$$\xi_2 = w_{21}X_1 + w_{22}X_2 + \ldots + w_{2p}X_p \tag{4.1}$$

$$\ldots$$

$$\xi_p = w_{p1}X_1 + w_{p2}X_2 + \ldots + w_{pp}X_p$$

The weights w_{ij} are estimated such that ξ_1 accounts for the maximum variance in $\boldsymbol{X}$, ξ_2 accounts for the maximum variance not accounted by ξ_1 in $\boldsymbol{X}$, and so on. The orthogonality among feature vectors is ensured by,

$$w_{i1}w_{j1} + w_{i2}w_{j2} + \ldots + w_{ip}w_{jp} = 0, \quad i \neq j \tag{4.2}$$

There is a possibility of increase in the variance of linear combinations, while adjusting the scale of weights; therefore (37) is used to fix the scale of feature vectors,

$$w_{i1}^2 + w_{i2}^2 + \ldots + w_{ip}^2 = 1, \qquad i = 1,2,\ldots,p \tag{4.3}$$

Furthermore, the correlation between the original independent variables and the feature vectors is called *loading*. Loadings can be obtained using the formula:

$$l_{ij} = \frac{w_{ij}}{s_j}\sqrt{v_i} \tag{4.4}$$

where l_{ij} and w_{ij} are the respective loading and weight of the *j*th variable of the *i*th feature vector, v_i is the variance of the *i*th feature vector and s_j is the standard deviation of the *j*th variable.

4.1.2 PCA based feature selection

For feature selection, statistically the first few feature vectors should be sufficient to capture most variance and result in substantially reduced representations in any modalities' datasets, as shown in table 4.1. However, the exact amount of unaccounted variance also

affects the data interpretations and further analysis. Therefore, the following strategies have been proposed for the choice of the number of feature vectors:

i). Retain only those feature vectors for which the eigenvalue is greater than one [121].

ii). Plot variance proportion across the feature vectors and determine an "elbow", marking the threshold of retaining the significant number of feature vectors.

iii). Retain only those feature vectors which are statistically significant.

Strategy (i) is the default in most of the statistical analysis methods [120]. In our case, it results in retaining 19 feature vectors out of 116 values for FS (table 4.1, column 2) and 7 feature vectors out of 18 values for AIS (column 7). On the other hand, strategy (ii) is widely used in scree plots (figure 4.1: plotting table 4.1 column:4 for FS and AIS feature vectors respectively). Scree plot shows that no elbow is clearly visible for FS feature vectors, while an elbow with a weak inflection point can be seen at 3rd feature vector for AIS feature vectors as shown in figure 4.1. From significant contribution of feature vectors and keeping at least 2 out of 3 strategies satisfied, a set of <u>significant 19 feature vectors from FS</u> and another set of <u>significant 7 feature vectors from AIS</u> are selected for multi-modality fusion.

In our work, loadings point out the extent to which the original independent variables are influential in forming feature vectors. Loadings of each feature vector can be defined as the contribution in total variance (listed in column:4 of table 4.1). In case of FS, the first independent variable is responsible for 44.35% of the total variance in forming the first feature vector, the second variable is responsible for 5.41% and the last 116th variable is responsible for 0.00% of the total variance. However, in case of AIS, the first variable is responsible for 13.68%, the second variable is responsible for 12.06% and the last 18th variable is responsible for 1.42% of the total variance.

FS				
Feature Vectors	Eigenvalue	Difference	Variance Proportion (%)	Cumulative Proportion (%)
1	**47.01**	41.27	44.35	44.35
2	**5.74**	1.79	5.41	49.76
3	**3.95**	0.53	3.73	53.49
4	**3.42**	0.39	3.23	56.72
5	**3.03**	0.48	2.86	15.23
6	**2.56**	0.12	2.41	61.99
7	**2.43**	0.51	2.29	64.28
8	**1.93**	0.05	1.82	66.10
9	**1.88**	0.07	1.77	67.87
10	**1.81**	0.17	1.70	69.57
11	**1.63**	0.18	1.54	71.11
12	**1.46**	0.07	1.37	72.48
13	**1.38**	0.05	1.31	73.79
14	**1.33**	0.07	1.26	75.05
15	**1.27**	0.05	1.20	76.25
16	**1.22**	0.06	1.15	77.40
17	**1.16**	0.08	1.10	78.50
18	**1.08**	0.05	1.02	79.52
19	**1.03**	0.07	0.97	80.49
20	0.96	0.03	0.91	81.40
21	0.93	0.07	0.88	82.28
⋮	⋮	⋮	⋮	⋮
115	0.00	0.00	0.00	100.00
116	0.00	-	0.00	100.00
AIS				
Feature Vectors	Eigenvalue	Difference	Variance Proportion (%)	Cumulative Proportion (%)
1	**2.46**	0.29	13.7	13.7
2	**2.17**	0.48	12.06	25.76
3	**1.69**	0.05	9.38	35.14
4	**1.64**	0.33	9.1	44.24
5	**1.30**	0.12	7.25	51.49
6	**1.18**	0.06	6.57	58.06
7	**1.12**	0.18	6.23	64.29
8	0.94	0.09	5.21	69.5
9	0.85	0.08	4.71	74.21
10	0.77	0.04	4.26	78.47
11	0.73	0.09	4.03	82.5
12	0.63	0.08	3.51	86.01
13	0.55	0.04	3.04	89.05
14	0.50	0.03	2.8	91.85
15	0.48	0.08	2.65	94.5
16	0.39	0.05	2.19	96.69
17	0.34	0.08	1.89	98.58
18	0.26	-	1.42	100

Table 4.1: Eigenvalues of the feature vectors using PCA

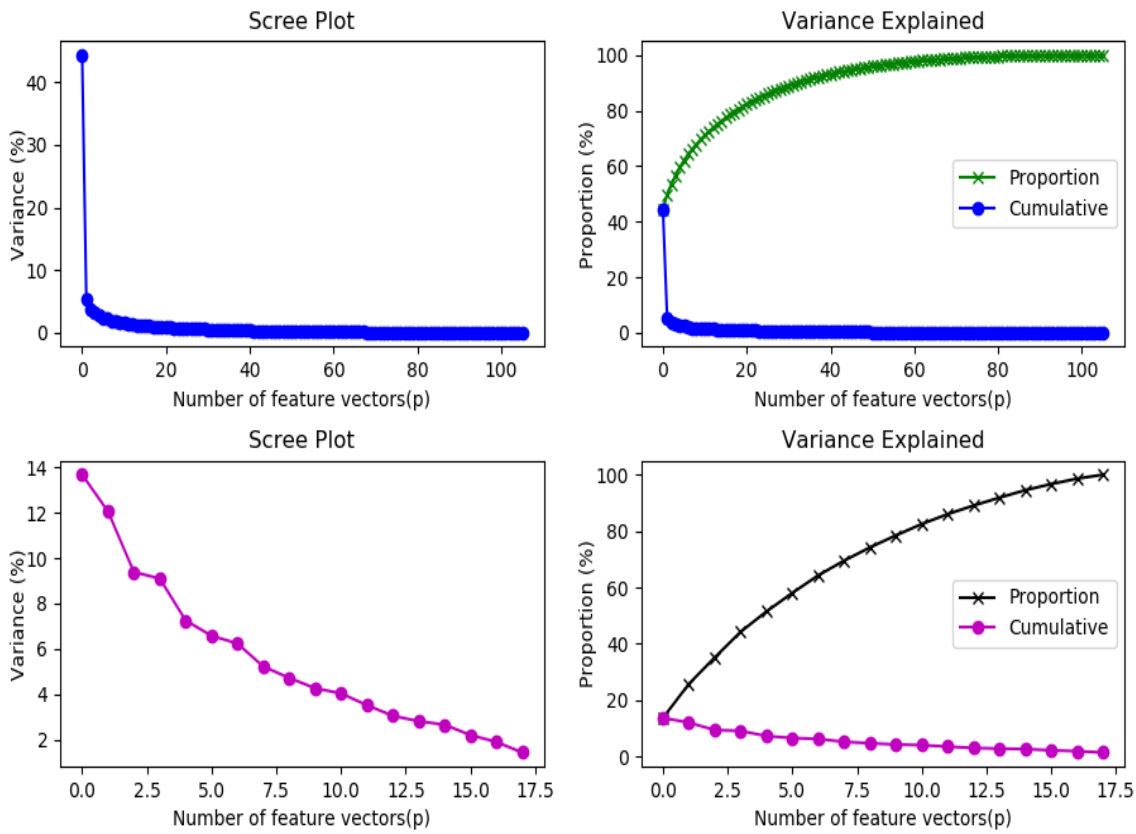


Fig 4.1: Top: Variance contributions of 116 feature vectors of FS; Bottom: Variance contributions of 18 feature vectors of AIS using PCA

4.1.3 CCA methodology

CCA is a multivariate statistical model that facilitates to find the relationship between two set of variables. CCA is very close to PCA in terms of its application on data however difference is the criteria of forming new feature vectors. In CCA, feature vector are formed in pairs with maximum correlation between them and uncorrelated with other feature vector pairs. Implementation of CCA can be summarized as follows [122]:

- Consider two sets of variables, $\boldsymbol{X} = [X_1, X_2, \ldots, X_p]$ and $\boldsymbol{Y} = [Y_1, Y_2, \ldots, Y_q]$
- Calculate the first feature vector pair as,

$$U_1 = a_{11}X_1 + a_{12}X_2 + \ldots + a_{1p}X_p \tag{4.5}$$

$$V_1 = b_{11}Y_1 + b_{12}Y_2 + \ldots + b_{1q}Y_q \tag{4.6}$$

where U_1 and V_1 are the first canonical variates pair (CVP) with canonical correlation C_1 between them. The objective of CCA is to estimate a_{11}, a_{12}, …, a_{1p} and b_{11}, b_{12}, …, b_{1p}, such that C_1 is maximum.

- Calculate the second feature vector pair as,

$$U_2 = a_{21}X_1 + a_{22}X_2 + \ldots + a_{2p}X_p \tag{4.7}$$

$$V_2 = b_{21}Y_1 + b_{22}Y_2 + \ldots + b_{2q}Y_q \quad , \tag{4.8}$$

where U_2 and V_2 are the second CVP, uncorrelated with U_1 and V_1 and maximum canonical correlation C_2 between them. C_2 is the maximum correlation not accounted by C_1. This process continues until the mth CVP,

$$U_m = a_{m1}X_1 + a_{m2}X_2 + \cdots + a_{mp}X_p \tag{4.9}$$

$$V_m = b_{m1}Y_1 + b_{m2}Y_2 + \ldots + b_{mq}Y_q \quad , \tag{4.10}$$

where $m \leq \min(p,q)$. U_m and V_m are the last canonical variates with maximum canonical correlation C_m between them. The following conditions of canonical correlation must be met by the set of m CVPs:

$$C_m(V_j, V_k) = 0 \quad j \neq k \tag{4.11}$$

$$C_m(U_j, U_k) = 0 \quad j \neq k \tag{4.12}$$

$$C_m(U_j, V_k) = 0 \quad j \neq k \tag{4.13}$$

4.1.4 CCA based feature selection

For feature selection, CCA has been applied on FS and AIS datasets to generate the CVPs. The maximum correlation between the initial CVPs, as compared to further ones, is shown in figure 4.2. Therefore, it is expected to have fewer CVPs with reduced dimensions and higher quality.

Table 4.2 describes the information related to each CVP, where 2nd column shows the correlation between canonical pairs. It is evident that the first pair has the maximum correlation of 52.24% and the last pair has the minimum correlation of 4.28%. Wilks' Lambda provides the significance of canonical correlations and tests the variance between

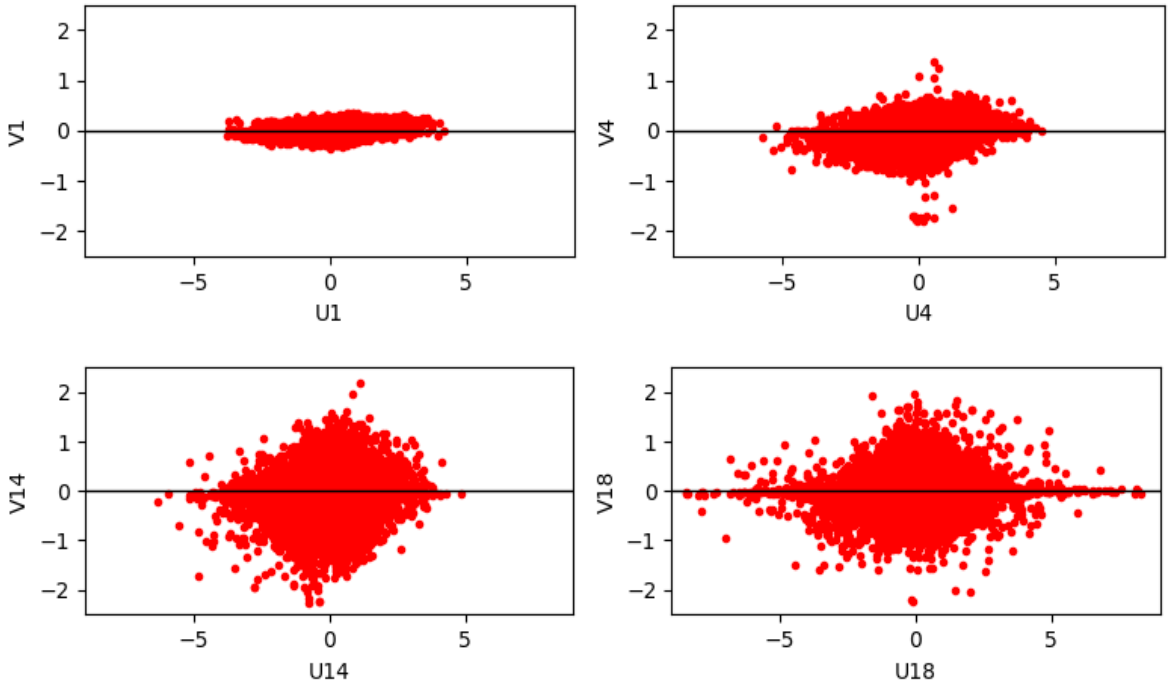


Fig. 4.2: 1st (U_1,V_1), 4th (U_4,V_4), 14th (U_{14},V_{14}) and the 18th (U_{18},V_{18}) CVPs obtained with p=116 and q=18

two datasets with variable number of inputs. Table 4.2, column: 3 shows Wilks' Lambda values as a cumulative contribution between 0 and 1 given by,

$$\Lambda_i = \prod_{k=i}^{m}(1 - C_k^2)\ , \tag{4.14}$$

where C_k^2 is the shared variance between V_i and U_i. F-distribution and probability values [123] are used to find the most significant CVPs (see 4th and 5th column). Normally significance values are thresholder at 0.05 or 0.01 [124]. The latter in used our research which results in 16 significant CVPs out of a total 18.

In the case of larger datasets, even smaller canonical correlation values could be statistically significant. On the other hand, a larger canonical correlation may not result from a stronger correlation between the two datasets. This is because CCA targets the maximum correlation among the linear combinations of variables in set ***X*** and set ***Y,*** and not the amount of variance in one of the sets, accounted for by the other. To find the total amount of variance among the two sets of variables, the redundancy measure is calculated as,

$$R_{V_i|U_i} = A_{Y|V_i} \times {C_i}^2 \tag{4.15}$$

CVP	Canonical Correlation	Wilks' Lambda (Λ)	F-Value (F)	Probability	Redundancy Measure $R_{V_i\|U_i}$	Total Redundancy $T_{R_{AIS\|FS}}$	Redundancy Measure $R_{U_i\|V_i}$	Total Redundancy $T_{R_{FS\|AIS}}$
1	0.5224	0.325	24.25	**<.01**	0.0122	0.0122	**0.03**	0.0299
2	0.4886	0.447	18.43	**<.01**	**0.025**	0.0369	0.0034	0.0333
3	0.3733	0.5872	12.98	**<.01**	0.0077	0.0446	0.0068	0.0401
4	0.3191	0.6823	10	**<.01**	0.0051	0.0496	0.0027	0.0428
5	0.2211	0.7597	7.75	**<.01**	0.0029	0.0525	0.0007	0.0434
6	0.2048	0.7987	6.89	**<.01**	0.0021	0.0547	0.0004	0.0439
7	0.1821	0.8337	6.09	**<.01**	0.0016	0.0562	0.0014	0.0453
8	0.1739	0.8623	5.47	**<.01**	0.0025	0.0588	0.0002	0.0455
9	0.1412	0.8891	4.81	**<.01**	0.0012	0.06	0.0002	0.0457
10	0.1397	0.9072	4.47	**<.01**	0.0015	0.0615	0.0004	0.0461
11	0.1299	0.9253	4.05	**<.01**	0.0005	0.062	0.0002	0.0463
12	0.1212	0.9412	3.65	**<.01**	0.0009	0.0628	0.0005	0.0467
13	0.1103	0.9552	3.25	**<.01**	0.0004	0.0632	0.0003	0.0471
14	0.1058	0.967	2.89	**<.01**	0.0005	0.0638	0.0001	0.0472
15	0.0936	0.9779	2.43	**<.01**	0.0004	0.0642	0.0001	0.0473
16	0.084	0.9865	1.98	**<.01**	0.0004	0.0645	0.0001	0.0473
17	0.0679	0.9936	1.43	0.01	0.0002	0.0647	0	0.0473
18	0.0428	0.9982	0.82	0.89	0.0001	**0.065**	0	**0.047**

Table 4.2: Canonical correlation analysis results

where $R_{V_i|U_i}$ is the redundancy measure, or the amount of variance in set $\boldsymbol{Y}$ that is accounted for by set $\boldsymbol{X}$ for the *i*th canonical correlation C_i^2. $A_{Y|V_i}$is the average variance in set Y accounted for by the canonical variate V_i expressed as,

$$A_{Y|V_i} = \frac{\sum_{j=1}^{q} L_{Y_{ij}^2}}{q} \tag{4.16}$$

Here, $L_{Y_{ij}^2}$ is the loading of the *j*th variable in set Y on the *i*th canonical covariate.

The total redundancy is the total variance accounted in one set of variables by the other set of variables, given by

$$T_{R_{Y|X}} = \sum_{i=1}^{m} R_{V_i|U_i} \tag{4.17}$$

Table 4.2 shows the redundancy measures of variance in AIS input variables accounted by FS input variables and vice versa in two sets of two colums, with the maximum variance and the total variance shown in bold. In the *AIS/FS* case, the maximum variance of 0.0247 (see 6th Column) is manifested by the 2nd CVP and total variance is 0.0648 (see 7th column). Likewise, in the *FS/AIS* case the maximum variance of 0.0299 (see 8th column) is attributed to the 1st CVP and total variance is 0.0474 (see 9th column).

4.2 Feature-level based sensor fusion approach

Feature-level fusion of different modalities involves extracting features from multiple sensors and generating an information pool of new representations which can be different from those acquired [9]. Feature-level fusion is helpful in situations where low computational cost is a key challenge. The focus on feature-level fusion is essential, since the efficiency of gait analysis depends on employing the maximum variability of data by automatically extracted features, rather than using hand-crafted features based on observational practice.

The proposed multi-modality sensor fusion system can be seen in figure 4.3. Raw data is obtained from two modalities i.e., FS and AIS is pre-processed and prepared for feature extraction. Feature are extracted using two techniques i.e., PCA and CCA. In case of PCA, 19 and 7 feature vectors are selected a FS and AIS respectively. These two feature sets have no relationship between them. However, in case of CCA the 16 CVPs are selected

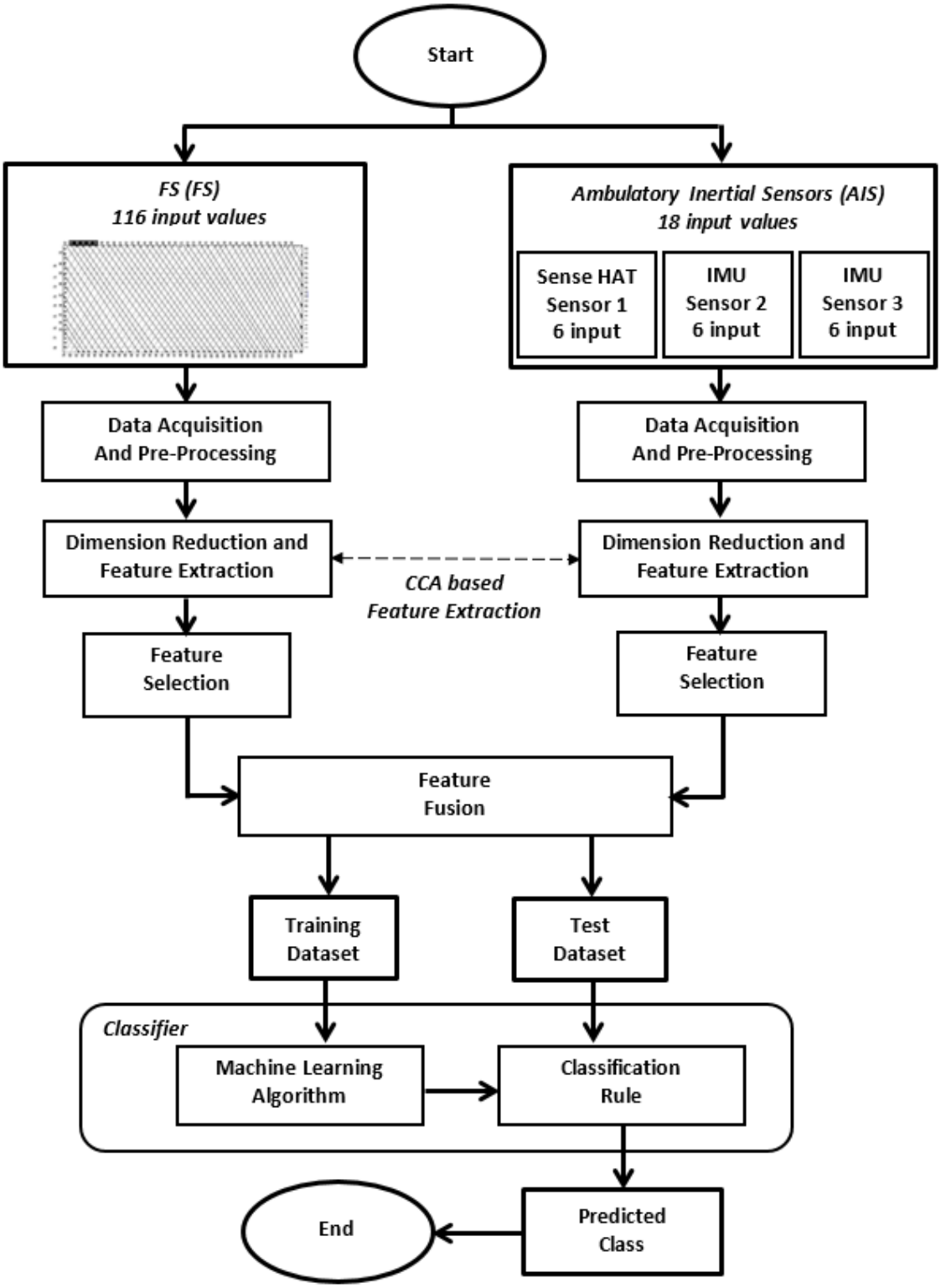


Fig. 4.3 Data flow diagram of proposed feature-level multi-modality sensor fusion system [133]

among FS and AIS such that each pair is mutually correlated and totally un-correlated with the other pairs. This mutual correlation between CVPs of FS and AIS is represented by dotted line in figure 4.3.

Selection of gait activity features using PCA/CCA when fused combine the discriminatory information obtained from two modalities and allow to capture more of the gait dynamics as it involves information gathered from measurements of diverse physical quantities. Concatenation is the most common and straightforward method used for feature-level sensor fusion [125]. In this work, having identified two sets of selected features; in FS feature space $S_1 \in R^{FS}$ and in AIS feature space $S_2 \in R^{AIS}$, the fused samples can be written as $S = (S_1 , S_2)$.

In this research, four cognitive based gait activities: walking while subtracting 7, walking while listening to a story, walking while texting on a mobile and walking while talking to operator are used to acquire spatio-temporal gait signals from 11 people using two modalities i.e., FS and AIS. One gait activy comprises a fixed 120 time frames window to complete, this procedure is repeated 10 times which increases the number of samples to 1200 for one person. Table 4.3 describes the spatio-temporal samples included in raw and fused modallity datasets used for classification purposes. Complete datasets including all 4 gait activities including single modality and multi-modality data are further split into 80% training and 20% test set ratio [112] before feeding to the ML algorithm. All experiments in this research are 10-fold cross validated in order to get the best learning outcomes [126]. 10-fold cross validation creates 10 trained test-folds of the training set (80% of data). ML model is trained on each test-fold and at the same time tested seperately on the test set (20% of data). Therefore, the model is evaluated on different test sets which increases the chances of accurate model predictions.

Spyder is an open source development environment which helps in data exploration, inspection, executions and visualizations using python libraries. Python libraries such as pandas, numpy, sklearn, seaborn, matplotlib etc are used in our work different task: data acquisition, feature extraction, concatenation, classification and visualisation of the spatio-temporal information obtained from both modalities. ML algorithms: LR, SVM, NB, K-NN, K-SVM, DT, RF (discussed in chapter 2: literature review) are used to find the best approximations that correctly map the sensors' data to the gait activities in supervised manner. The challenge for the ML models is to achieve the highest classification accuracies, as observed through the classification scores with optimized hyper-parameters.

Experiment	Total Samples per person	Raw Samples using FS (116 Inputs)	Raw Samples using AIS (18 Inputs)	Fused Samples using PCA (26 Comp)	Fused Samples using CCA (16 CVPs)
1. Walking while subtracting 7 **2.** Walking while listening to a story **3.** Walking while texting on a mobile **4.** Walking while talking to operator	1,200	1,200 samples x 11 person = 13,200	1,200 x 11 = 13,200	13,200 FS samples + 13,200 AIS samples = 26,400	13,200 + 13,200 = 26,400
Total Samples	1,200 x 4 activities = 4,800	13,200 x 4 = 52,800	13,200 x 4 = 52,800	26,400 x 4 = 105,600	26,400 x 4 = 105,600

Table 4.3: Samples for feature-level based classification

Chapter 5

Deep Learning based multi-modality sensor fusion

Multi-modality sensor fusion results in producing new data representations which are unique to the collection of individual sensors and modalities. Several modalities have demonstrated their capabilities to capture gait attributes and anomalies; however, most of these methods rely on handcrafted features. In such approaches, feature engineering might lose the salient features involved in problems. In our work, DL achieves the learning and extracting of highly statistically significant features from the gait activity data recorded from two different modalities. DL models implemented and used to extract gait features from both modalities, are discussed as follows:

5.1 FFNN for single and multi-modality cases

The neural network in which output from one layer is fed to the next layer in forward direction without any loops in the network is called a feed-forward neural network [93]. The basic architecture of a FFNN model consists of an input layer, few hidden layers and an output layer of neurons. In the input layer, each input value from the respective datasets is represented by an input node. The spatio-temporal data from both modalities, after pre-processing is passed to the fully connected input layers of sizes 64 (116 input FS) and 16 (18 input AIS) respectively. In FFNN, the neurons in one layer are fully connected to the next layer through synapses or assigned weights to learn the complex representations of data. The weights are initialized with a value close to zero and bias is added to breakup linearity during model training.

In our work, for AIS, the training set is a 2D vector (73920x18) in which each row represents the spatial data at a single time instance. 18 Input values are passed to the fully

connected (FC) layers of sizes 16, 12, 10, 8 and an output layer of size 7 (representing 7 gait activities). The first layer size (16 being a multiple of 2 and closer to the average of input (18), output (1)) is selected, however, any number between number of input and output can be selected. Also, higher accuracy is observed using first layer of size 16 than 8. The effect of every weight at FFNN layers is determined by the activation function which allows the model to achieve a desired output. To introduce the non-linearity of the spatio-temporal gait patterns in our dataset a Rectified Linear Unit (ReLU) activation function [127] is implemented at all the hidden layers. The weight of every neuron is multiplied by the input and passed through the activation function. Propagation continues until a prediction is achieved. At the output layer of size 7, a linear classifier *SoftMax* is used to transform results into probabilities [128].

For FS, the training set is also a 2D vector (73920 x 116). 116 input values are passed to the FC layers of sizes 64, 32, 10, 8 and an output layer of size 7. For the multi-modality case, in order to create a balance between the number of features, the FC layers of size 10 from each modality are merged as shown in figure 5.1. The outputs from each layer are passed in forward direction to the next layer. For multi-modality case, forward propagation takes places over the merge layer.

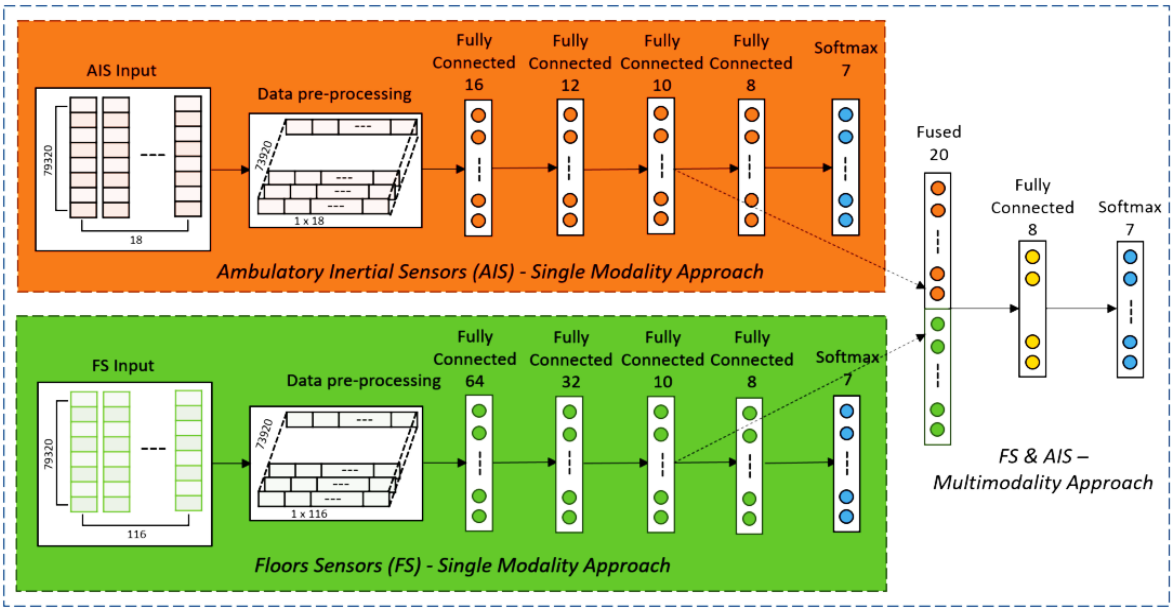


Fig 5.1: Layer-wise distribution of AIS (orange), FS (green) and Proposed FS & AIS multi-modality approach (white) using FFNN

Likewise forward propagation, the FC layers are responsible for the propagation of error in back ward direction. The predicted results are compared with the actual results and the error is quantified with the help of a cost function [94], [95]. We have used cross-entropy, based on a logarithmic function to handle very small errors. The error is back propagated in the form of updated weights send to the neurons layer-wise in backward direction. Among the gradient based algorithms such as stochastic gradient descent [96], conjugate gradient [97] and Adam [98], which are the commonly used methods for error optimization, the latter is used to determine the learning rate of new weights and biases in our research.

The above procedure is repeated, and weights are updated after each batch of observations from the training set for each modality. Batch size of 120 observations is selected to update the weights which is equal to one activity. One epoch is completed when one whole training set passes through the FFNN. We have trained all experiments through 100 epochs for all cases. Results are further discussed in chapter 6.

5.2 1D-CNN for single and multi-modality cases

A basic CNN consists of an input layer, convolution layers, down-sampling or pooling layers, flattening layers, FC layers and an output layer [129]. In this work, the implementation of 1D-CNN for single and multi-modality cases can be seen in figure 5.2. For AIS, the training set 73920x18 is converted into 73920 arrays of size 1x18, where a single array determines the spatio information at a single time instance. Each array is passed to a 1D-Convolution layer (Conv1: 32 filters, kernel size 3, stride 1) to automatically extract the unique variability features from the training dataset. Max-pooling layer (MP1: kernel size 2) is used to down-sample the large volume of data after convolution. Results obtained from MP1 are feed to another 1D-Convolution layer (Conv2: 16 filters, kernel size 3, stride 1) and a Max-pooling layer (MP2: kernel size 2). Extracted features from max-pooling layers are in 2D format and therefore required to get aligned in a 1D feature vector of inputs for FC layers using the flattening function. FC layer of size 16 is used to connect flattening output. ReLU is the activation function used to handle the non-linearity at the convolution layers and the FC layers. SoftMax function is used at the output layer (size 7) as discussed earlier.

For FS, the training set 73920x116 is converted into 73920 arrays of size 1x116, where a single array contains the spatial information at a single time instance. Each array is passed to a 1D-Convolution layer (Conv1: 64 filters, kernel size 3), Max-pooling layer

(MP1: kernel size 2), another 1D-Convolution layer (Conv2: 16 filters, kernel size 3), Max-pooling layer (MP2: kernel size 2), flattening layer and a FC layer of size 16 followed by output layer of size 7. For the multi-modality case, to create a balanced number feature set, the FC layers of size 10 from each modality are merged.

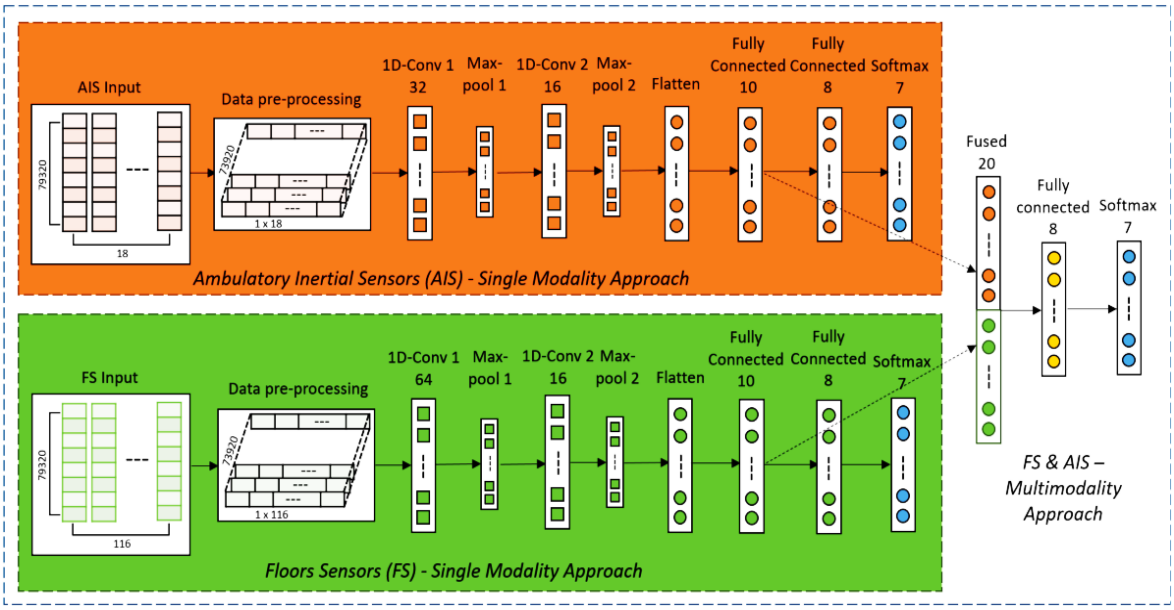


Fig. 5.2 Layer-wise distribution of AIS (orange), FS (green) and Proposed FS & AIS multi-modality approach (white) using 1D-CNN

5.3 2D-CNN for single and multi-modality cases

The implementation of 2D-CNN for single and multi-modality cases can be seen in figure 5.4. The same number of layers and filters at each layer are used for the 1D and 2D approach. However, the dimensions of inputs and size of convolutional and max-pooling layer are different. Since CNN are most applied to analyze visual images, therefore we have utilized their ability by transforming the 18 inputs of AIS into a 5x5 image and 116 inputs of FS into a 7x7 image with zero padded columns each as shown in figure 5.3. The filters kernel size for each convolutional layer is 3x3 and for max-pooling layer is 2x2. Results obtained for all cases are discussed in chapter 6.

It is important to mention here that for both CNN-1D &2D, the number of filters is only different in the first convolutional layer however all the following filtered and hidden layers are identical for FS and AIS cases when compared. First convolutional layer of size 32 and 64 are reported for AIS and FS respectively due to the best performance achieved in terms of higher accuracy and lesser execution time for both cases.

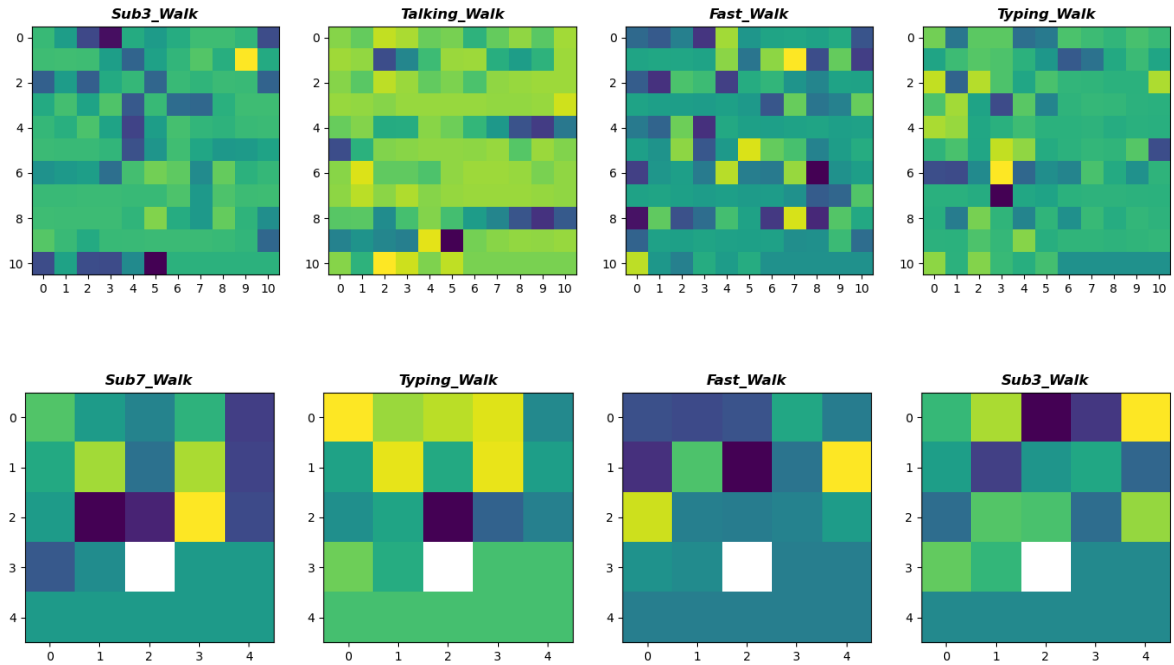


Fig. 5.3 Random spatio-temporal samples from Top: FS and Bottom: AIS training datasets where each input is an image of size 11 x 11 and 5 x 5 respectively

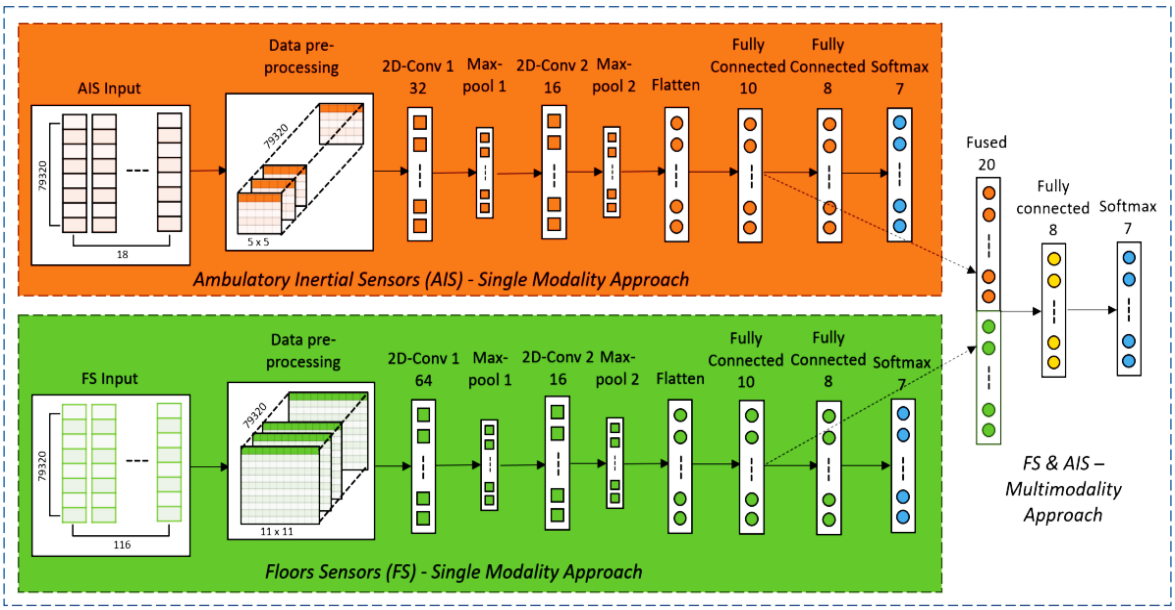


Fig. 5.4: Layer-wise distribution of AIS (orange), FS (green) and Proposed FS & AIS multi-modality approach (white) using 2D-CNN

5.4 LSTM for single and multi-modality cases

LSTM models work on time-processed data and are capable of learning time dependencies in sequence prediction problems. Since timestamps are equal in number for both modalities, the first layer of operation has been implemented with 16 blocks for both cases. Stacked layer LSTM models have been used to deeply exploit the dependencies between time-stamps [130]. The two stacked layered LSTMs, reported in many cases have been adopted in our approach to implement the individual [33] and multi-modality cases [131].

For AIS, the training data set is a 2D vector (73920 timestamps x 18 inputs) which is converted into a 3D vector (73824 time-stamps x 120 window samples x 18 inputs), with 120 window samples out of 119 serve as memory for the associated timestamp. Training data is fed to the successive LSTM model in the form of batches of size 120 for different epoch values. The LSTM stack of two layers is implemented with 16 LSTM units which are selected due the best performance achieved in terms of higher accuracy and lesser execution time for FS and AIS cases. Higher sizes of LSTM layers such as 32 and 64 causes system hang up problems for the available computational resources mentioned on page 97. Each LSTM layer is followed by a dropout layer (DO) which utilizes percentage probability of data to prevent any overfitting. In our research, dropout more and less than 20% had no improvement in accuracies therefore we have reported the optimized value of 20% dropout

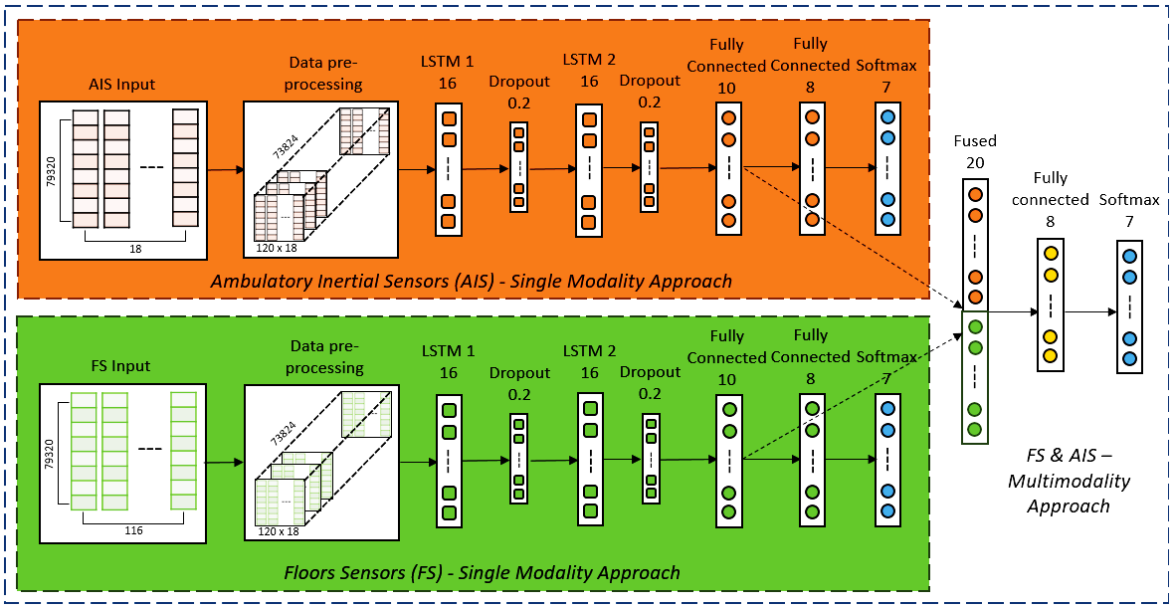


Fig. 5.5 Layer-wise distribution of AIS (orange), FS (green) and Proposed FS & AIS multi-modality approach (white) using LSTM

in our case. In case of FS, we have training data as a 3D vector (73824 time-stamps x 120 window samples x 116 inputs) following a similar LSTM model like AIS. After two layers a similar layered approach has been utilized as in case of FFNN, 1D-CNN, 2D-CNN for single and multi-modality cases as shown in figure 5.5.

5.5 Overview of DL based multi-modality sensor fusion

In this work, a DL based fusion of lower human body joint angle trajectories (obtained from an AIS modality) and ground reaction forces generated by feet (obtained using POF based FS modality) is presented.

AIS and FS, each modality records their datasets on their own RPI. The two respective RPIs are programmed to synchronize and record readings at 20Hz. Both modalities are checked, and tested before starting experiments. Seven gait activities such as: Normal Walk, Fast Walk, Subtracting 3 walk, Subtracting 7 walk, Listening story & walk, Typing on mobile & walk and Talking & walk are performed to generate and acquire spatio-temporal gait signals from 11 people during each gait activity (recorded 10 times) using both modalities.

120 samples are obtained from a single person during a single gait activity (recorded 10 times increases the sample number to 1,200) in a single modality. 13,200 samples are collected for one activity and 92,400 samples are collected for 7 activities using a single

Gait Activities	**Samples per person**	**Single Modality Samples**	**Multi-Modality Samples**
1. Normal Walk 2. Fast Walk 3. Subtracting 3 Walk 4. Subtracting 7 Walk 5. Listening story & Walk 6. Typing on mobile & Walk 7. Talking & Walk	1,200	1,200 samples x 11 person = 13,200	13,200 FS samples + 13,200 AIS samples = 26,400
Total Samples	1,200 x 7 activities = 8,400	13,200 x 7 = 92,400	26,400 x 7 = 184,800

Table 5.1: Samples for DL based classification

modality. Similarly, 184,800 samples are collected for 7 activities using multi-modality as mentioned in table 5.1. Each case of single and multi-modality is split into 80% training, 10% validation and 10% test sets before feeding to the DL model. Training dataset from two modalities is used to train the DL model which is validated and tested for maximum 100 epochs and 120 batch size of information. All data processing and computational tasks are conducted on Lenovo ThinkPad with Intel® Core™ i7-8560U CPU @ 1.9GHz 2.11GHz and 8GB physical memory.

From the point of view of the fusion task, the data is collected from synchronized RPIs separately. In this research, we have utilized the first two layers from each DL model (as shown in figure 5.6) to automatically extract unique gait activity features from both modalities that mostly contribute towards gait classification whilst dropping the less significant values across the complex network layers. Fusion of this unique information helps to retain most of the gait activity dynamics from individual modalities. Python environment libraries, including TensorFlow and Keras, are utilized to implement and run DL models. DL model layers can process the body orientation, positioning and forces in space and time using AIS. Likewise, these layers are equally useful to process the effect of forces resulted in foot on ground contact captured in FS data. Different techniques have been proposed and adapted for sensor fusion using DL models [132]. Some of the techniques implemented in this work, are as follows:

i). **Add:** Two input vectors of same size are added into a single vector of the same size as individual inputs.

ii). **Multiply:** Multiply two inputs vectors (like Add).

iii). **Average:** Computes the average of two input vectors into a single vector of the same size as individual inputs.

iv). **Maximum:** Computes the maximum of the two input vectors into a single vector of the same size as individual inputs.

v). **Minimum:** Computes the minimum of the two input vectors (like Maximum).

vi). **Concatenate:** Combines two inputs vectors into a single long vector, so that that the second input vector comes below the first.

The listed layers perform arithmetic operations on their input layers and require them to be the same shape for fusion. However, concatenate layer can work with different shape inputs. Results obtained using these layers are discussed in chapter 6.

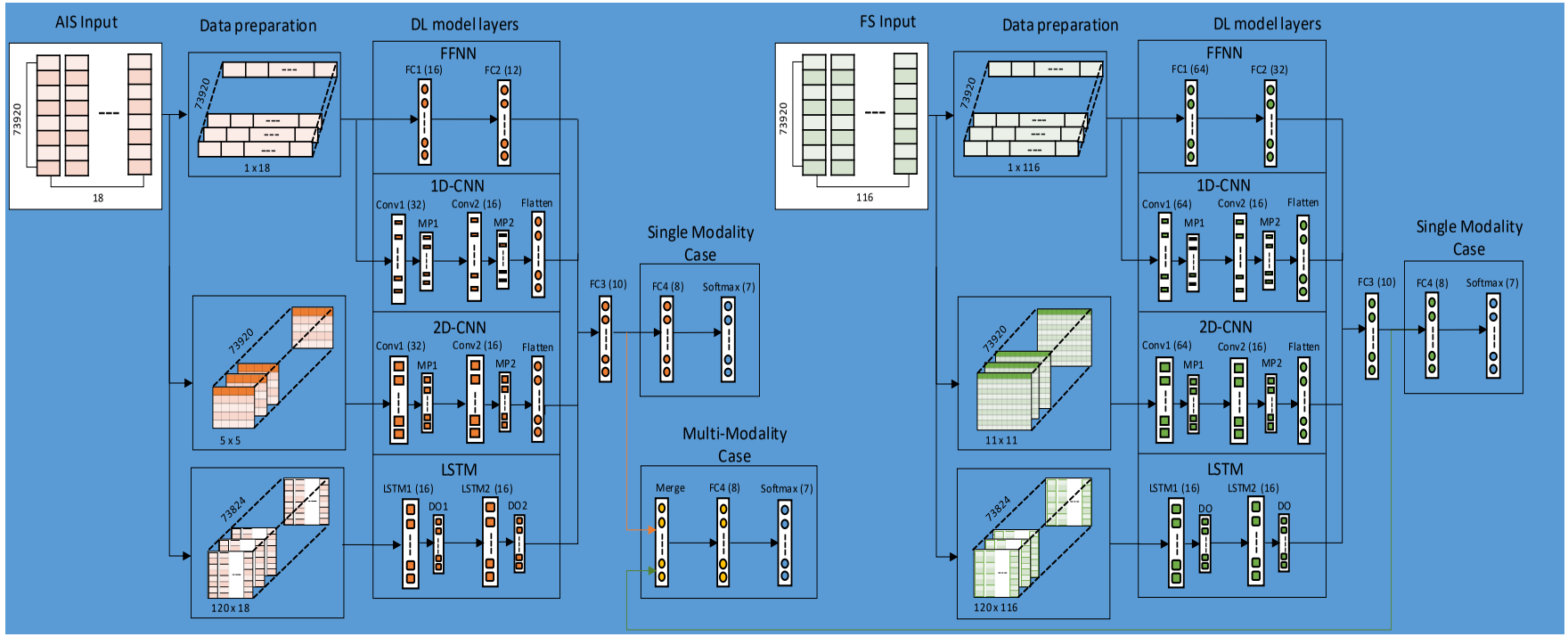


Figure 5.6 Complete diagram of single and multi-modality cases using DL models

Chapter 6

Results and discussions

FS captures information related to feet contact on the floor and it can only measure limited aspect of movement manifested in human gait. Likewise, AIS captures only the acceleration and angles representing the kinematic movements from lower portion of body such as lower back and ankles. It is reasonable to expect that fusion of information from both sources helps to compensate the degraded spatial and temporal accuracy in certain situations, making the overall classification more robust as compared to a single modality approach. Benefits of the deployed complementary modalities with a balanced in cost and acceptability by the individuals can be seen in the comparison of results obtained from single and multi-modality approaches. Meanwhile keeping the focus on fusion of spatio-temporal information at feature level, this works examines significant differences in the performance of multiple classification algorithms using single and multi-modality systems. Results are summarized and discussed as follows:

6.1 Feature level based multi-modal fusion

Feature level fusion of single modality systems has obvious advantages. Features extracted from single modalities reflect different characteristics of data. Combination of these different features not only preserves the essential discriminant information but also eliminates the redundant information to a certain extent. PCA and CCA are used as data reduction methods whilst fusing the extracted features. Using PCA, features are extracted in such a way that initial feature vectors represent maximum variance in the dataset then the latter one. Number of new feature variables formed are equal to the sum of inputs in both datasets (134 feature vectors from input {116,18}). On the other hand, in CCA, the new

feature vectors are formed in pairs and number of pairs is equal to the number of inputs in the minimum input dataset (18 CVPs or 32 feature vectors). Feature vectors in the initial CVPs have more tendency to maintain linearity than the latter ones (as shown in figure 4.2). Therefore, the obvious advantages using CCA on PCA are two folds: First, CCA results in overall lesser number of feature vectors i.e., 32 as compared to 134 feature vectors using PCA to describe the total variance of the two datasets. Second, CVPs help to eliminate the redundant information whilst preserving the essential discriminant information resulting in fewer feature vectors as compared to PCA. This aspect of CCA makes it useable in computational load reduction when dealing with huge datasets.

Data selection has been implemented to find out the most significant features using PCA and CCA. This resulted in 26 significant feature vectors using PCA and 16 CVPs (32 feature vectors) using CCA. The techniques used to select these significant values using both methods are mentioned in chapter 4. A comparison of classification between single modality f-scores (data without feature extraction) and dual-modality f-scores (data with feature extraction) can be seen in Table 6.1, where the 2nd and 3rd columns represent the results of 'raw' values obtained from a set of 116 inputs from FS and 18 inputs from AIS. A direct comparison between the disproportionate sensed parameters from the 2nd and 3rd columns is futile. However, we have proposed a balanced and successful methodology to fuse the two modalities using PCA and CCA to keep the maximum variant features from the two. This sort of fusion is implemented on reduced inputs without the substantial degradation of the spatio-temporal information occurs in individual modalities.

6.1.1 Results and discussion (Part-I)

Results obtained using ML algorithms are shown in table 6.1. Column 2 & 3 show results of ML algorithms using 116 and 18 raw inputs obtained from single modality FS and AIS respectively. Column 3, 4 and 5 show results of ML algorithms using fused inputs comprising 26 PCA components, 10 and 16 CVPs respectively.

The second column might show higher f-scores 99.92±0.23% using 116 raw inputs using FS in some cases which brings up the question about the need for proposed fusion. In this research, we have proposed a suitable feature level fusion of spatio-temporal information obtained from two modalities which is specific to the scenario in which the user is using both modalities. Results for 'raw' input are listed to highlight the imbalanced number of

inputs from both modalities and their respective f-score for reference. PCA and CCA are implemented as robust and efficient feature extraction techniques which reduce the computational load of inputs on ML algorithms (say, 116 in case of FS to the fused 26 PCA, 10 and 16 CVPs). Further, the resulting data is fused and assessed using various classification algorithms. Classification f-scores results for multi-modality fusion in case of 26 components PCA and 10/16 component CCA are listed in columns 4, 5 and 6 respectively with improvements in later case. Here 26 component PCA has overall high f-score with few exceptions when compared with 10 CVPs (10 +10 components). On the other hand, 26 component PCA has overall lesser high scores when compared with 16 CVPs (16 +16 components). This shows CCA as an overall advantageous method when compared to PCA for most of the ML algorithms based on the selected significant components (method of selection is mentioned in chapter 4).

It can be seen in table 6.1 (highlighted cases) that f-score measure for 26 component PCA using K-SVM (4th column) and for 16 CVPs using NB, K-NN and K-SVM are higher than individual modalities (2nd and 3rd column) simultaneously. It is noticeable that when compared with the individual modalities 16 CVPs has higher f-scores than 26 component PCA for NB, K-NN and K-SVM. However, in terms of computational load 26 components PCA is still efficient as compared to 16 CVPs or 32 component CCA.

Since CCA make use of correlation to describe the unique information between two datasets, this resulting in the option to select fewer components to achieve higher f-scores. Using 10 CVPs or 20 component CCA (5th column), it is evident that higher f-scores are achieved using NB and K-SVM.

6.1.2 Role of ML models

For multi-modality cases, the overall accuracies for linear classifiers: LR and SVM are lower compared to the non-linear classifiers: NB, K-NN, K-SVM, DT and RF. Since we are dealing with two modalities of different nature i.e., FS uses the disturbance of light stopped by the pressure asserted on the POF based sensors whereas AIS captures acceleration and angular velocity values from pelvis and both heels of the individuals.

As shown in table 6.1, K-NN outperforms all ML models and acquires 95.03±0.98% f-score for 16 CVPs using CCA. It is clear that the improvement is marginal over FS (2nd column)

Machine Learning	**SM-FS (116 Inputs)**	**SM-AIS (18 Inputs)**	**MM-PCA (26 Component)**	**MM-CCA (10 Covariate Pairs)**	**MM-CCA (16 Covariate Pairs)**
LR	66.17±1.30	32.73±1.50	43.73±1.41	40.94±1.98	44.25±1.37
SVM	74.30±1.04	32.06±1.84	47.03±1.72	41.14±2.41	47.22±1.99
NB	37.41±0.96	40.37±1.51	36.73±1.41	**44.78**±0.48	**48.12**±0.89
Kernel-NN	94.53±1.64	66.81±1.16	90.14±1.00	92.81±0.97	**95.03**±0.98
Kernel- SVM	90.86±0.33	60.89±1.04	**91.36**±1.15	**92.34**±1.15	**94.33**±1.47
DT	99.28±0.12	61.31±1.52	89.88±1.02	82.75±0.99	83.44±0.89
RF	99.92±0.23	72.49±1.07	94.39±1.11	89.03±1.01	89.46±0.87

Table 6.1: Overall classification f-scores ± standard deviation percentages for single modality and multi-modality systems

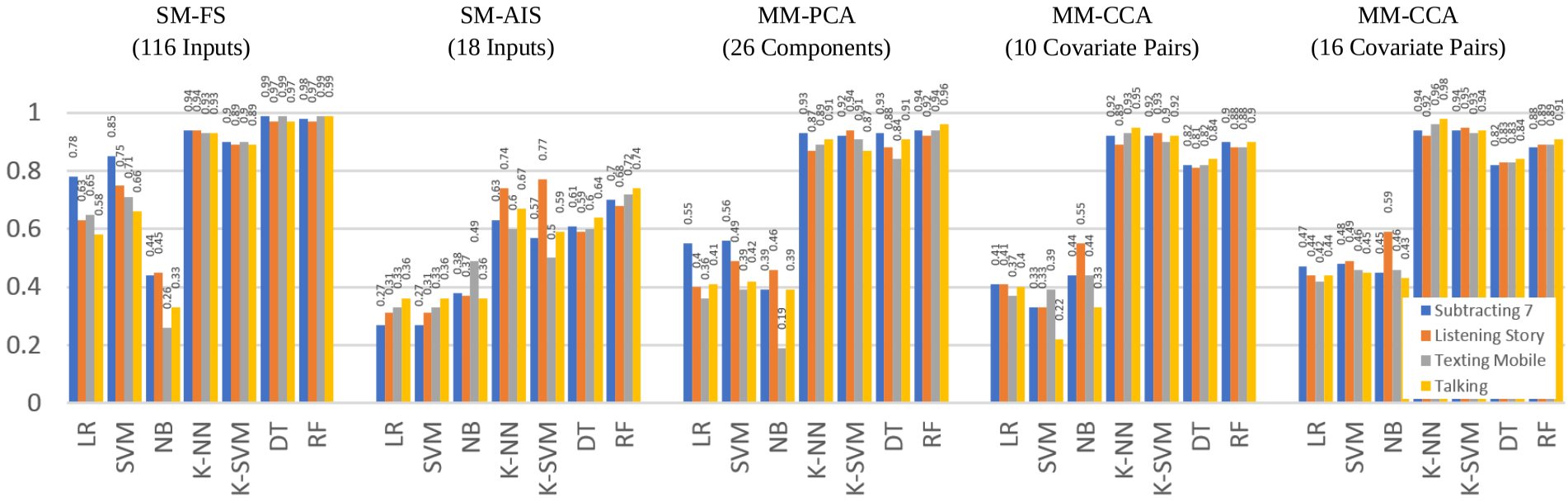


Fig 6.1 Classwise f-score percentages for different classifiers on 4 gait activities

SM: Single Modality; MM: Multi-Modality; LR: Logistic Regression; NB: Naïve Bayes, DT: Decision Tree, RF: Random Forest

but substantial over AIS (3rd column). However, K-SVM has an overall increased f-scores for 26 component PCA and 10/16 CVPs CCA as compared to single modalities FS and AIS. DT and RF manifest higher f-scores in case of individual modalities (especially for FS) but show degraded f-scores for the proposed multi-modality feature fusion.

Figure 6.1 shows the classwise f-scores of individual gait activities. The ‘talking activity’ in 10 CVPs CCA, ‘texting on mobile’ and ‘talking’ activity in 16 CVPs CCA have higher f-score than any other gait activity and these results are obtained using K-NN. Specifically considering K-NN, activity comparison of 26 component PCA with 10 CVP CCA reveals the latter as a better approach for all activities with one exception of ‘Subtracting-7’ activity (93%) for 26 component PCA. However, activity comparison of 26 component PCA with 16 CVP CCA yields later as a superior method for multi-modality sensor fusion.

Similarly, considering K-SVM, activity comparison of 26 component PCA with 10 CVP CCA reveals the former as a better approach for all activities with one exception of ‘Talking’ activity (87%) for 26 component PCA. However, activity comparison of 26 component PCA with 16 CVP CCA yields later as a superior method for multi-modality sensor fusion. For all gait activities and using our proposed feature level multi-modality sensor fusion 10 CVP CCA appeared as 2nd best choice in terms of low computational load and robust classification f-scores.

6.2 Deep learning based multi-modal fusion

In our previous work, we have performed feature extraction using PCA and CCA. However, feature engineering might lose the salient features involved in problems. Therefore, we have employed deep learning methods to learn and extract the highly statistically significant features from gait activity data recorded using two modalities. The results achieved using single and multi-modality systems are used to explore the benefits of complementary modalities in comparison with the cost and acceptability by the user. While retaining our focus on multi-modality fusion, significant differences in the performance of multiple DL models are observed.

6.2.1 Results and discussion (Part-II)

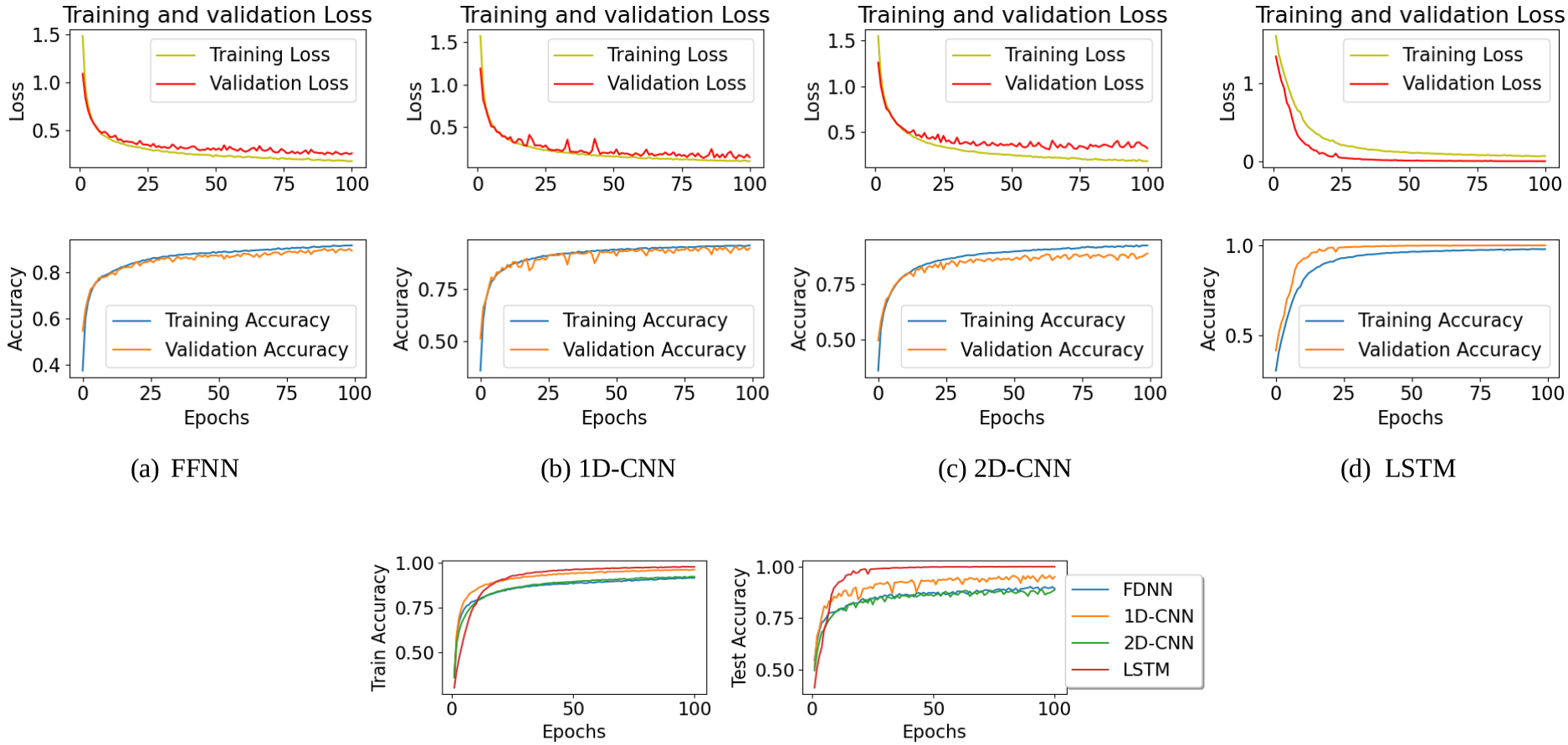
In our work, we have compared single and multi-modality fusion over DL models: FFNN, 1D-CNN, 2D-CNN and LSTM. We have used two processing layers from every DL model to perform the fusion of multi-modality sensor data as shown in figure 5.6. Results are corroborated in table 6.2 for a range of epochs 1-100 and DL models are accessed based on the f-scores and execution times.

It is expected to achieve higher f-scores for FS (116 inputs) as compared to AIS (18 inputs) which can be seen in table 6.2. It is also understandable that the execution time to generate classifications from FS (116 inputs) is much higher than AIS (18 inputs). In our work, we have proposed a fusion strategy to balance the disproportional number of inputs between the two modalities, without substantial degradation of the information content. The classification features obtained using the fused multi-modality data yielded better f-scores as compared to individual modalities using all DL models (see Table 6.2, columns 3, 5 & 7). However, this is achieved at higher execution time than single modalities.

The accuracy and loss graph for training and validation datasets obtained using all DL models is presented in figure 6.2(a)-(d). Further, test accuracy is also mentioned across all DL models in figure 6.2(e). It is found that in case of LSTM, the validation accuracy is minimum (bottom left figure) at 10th epoch as compared to other DL models. However, for 50 and 100 epochs, LSTM has attained highest accuracies over other DL model. A detailed record of tested results for individual and multimodality cases of all DL models has been described in table 6.2, which confirms the behaviour of models for the combined dataset.

DL Models	Epochs	FS	Execution Time (hh:mm:ss)	AIS	Execution Time (hh:mm:ss)	FS & AIS Multi-modality Fusion	Execution Time (hh:mm:ss)
FFNN	1	50.09	00:00:01	18.69	00:00:01	54.64	00:00:02
	5	72.56	00:00:07	30.11	00:00:06	73.84	00:00:09
	10	75.48	00:00:13	31.14	00:00:12	78.27	00:00:18
	50	81.19	00:01:06	36.69	00:01:04	87.35	00:01:27
	100	82.49	00:02:13	38.44	00:02:11	**89.33**	**00:03:06**
1D-CNN	1	50.34	00:00:12	27.19	00:00:04	51.25	00:00:12
	5	73.75	00:00:56	32.4	00:00:21	80.9	00:01:05
	10	79.72	00:02:01	34.73	00:00:43	84.75	00:02:06
	50	89.12	00:10:23	41.2	00:01:44	93.52	00:10:43
	100	94.71	00:20:01	42.47	00:03:30	**94.97**	**00:21:14**
2D-CNN	1	40.15	00:00:18	27.42	00:00:05	49.73	00:00:13
	5	64.7	00:00:59	34.63	00:00:26	73.31	00:01:05
	10	69.1	00:02:01	38.16	00:00:51	78.3	00:02:10
	50	76.89	00:10:14	44.29	00:03:34	85.7	00:10:49
	100	80.4	00:20:25	45.75	00:07:03	**88.73**	**00:21:40**
LSTM	1	34.56	00:02:32	33.2	00:01:08	41.43	00:03:17
	5	54.79	00:11:09	44.05	00:05:46	69.83	00:16:12
	10	72.49	00:23:51	68.26	00:11:15	90.93	01:06:37
	50	99.5	05:22:55	91.23	00:56:15	99.77	12:14:32
	100	99.78	22:51:17	95.19	01:52:31	**99.9**	**23:23:45**

Table 6.2: F-scores percentages for single modality and multi-modality fusion using DL models



(a) FFNN (b) 1D-CNN (c) 2D-CNN (d) LSTM

(e)

Fig 6.2: Training vs validation loss and accuracy for proposed multi-modality approach using (a) FFNN, (b) 1D-CNN, (c) 2D-CNN, (d) LSTM, (e) comparison of training accuracy and test accuracy for all DL models

The effectiveness of this DL based multi-modality fusion has been further tested and verified using different fusion techniques as discussed in section 5.5. The 'add' method appears to deliver the most accurate fused result among all. However worst f-scores are obtained using 'minimum' and 'multiply' methods in case of 2D-CNN, as shown in figure 6.3.

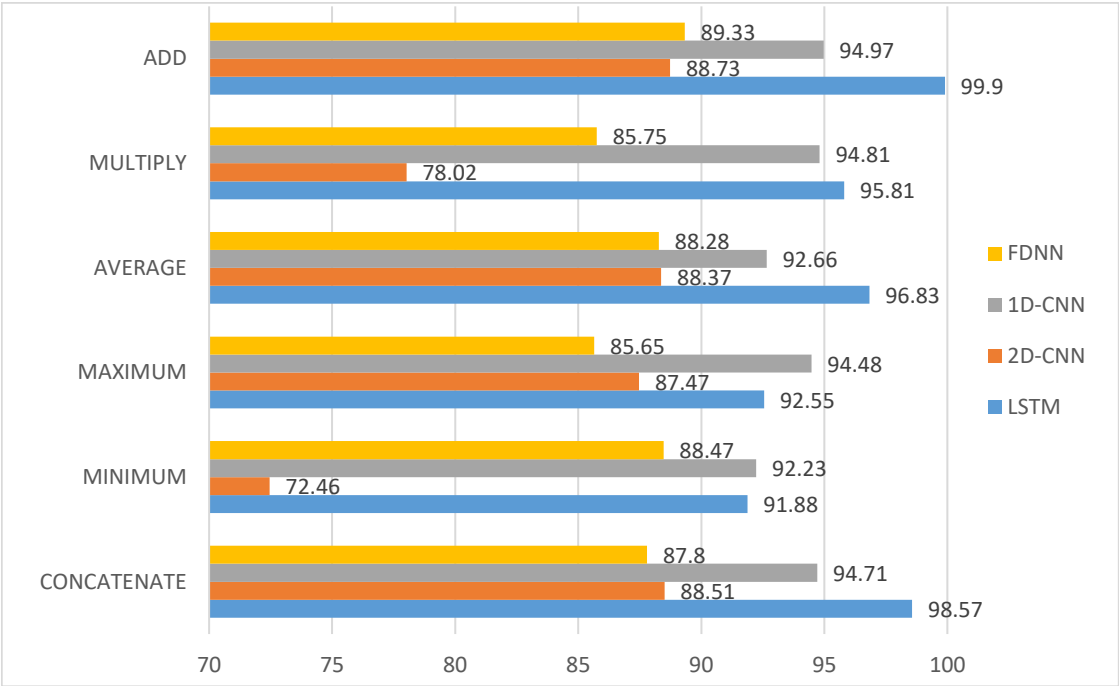


Fig 6.3: F-scores for overall gait classification using fused approaches (epochs: 100, batch size: 120)

6.2.2 Role of deep learning models

The f-scores for DL models: FFNN, 1D-CNN, 2D-CNN and LSTM are higher in multi-modality cases as compared to individual modalities as shown in table 6.2. In case of FS (see column: 3), 1D-CNN shows higher f-scores for all epochs when compared with FFNN and 2D-CNN. Comparison of 1D-CNN with LSTM shows mixed results with higher f-scores for 50 and 100 epochs in the latter case.

LSTM models are capable of learning time dependencies in sequence prediction problems and work on time processed data. Since the timestamps are equal in number for both modalities, the first two layers of operation have been implemented with 16 units for both cases (see figure 5.6). A higher number i.e., 32 or 64, is reportedly beyond the capabilities of the computer system used. LSTM shows higher f-scores for all epochs when

compared with FFNN, 1D-CNN and 2D-CNN in case of AIS (table 6.2: column 5). For the multi-modality fusion case, LSTM has the highest f-scores for 10, 50 and 100 epochs in case of all DL models.

In case of 1D-CNN and 2D-CNN, the first two layers of operation have the same number of filters for single and multi-modality cases (see figure 5.6). FS data, considering a 5-fold larger number of inputs compared to AIS, have shown the maximum f-scores with 64 filters, as compared to AIS with 32 filters. AIS has been checked with 16 filters too manifesting reduced f-scores. The scope of this research is to report the most suitable approach for the fusion task. 1D-CNN proves itself as a second choice when compared with 2D-CNN and a single exception at 1 epoch with FFNN.

The execution time to train 1D-CNN, 2D-CNN and LSTM models is significantly higher for FS than AIS in all epochs (see columns: 4 & 6). Only FFNN has comparatively closer execution times using FS and AIS. In case of multi-modality fusion, FFNN takes much lesser time compared to LSTM which manifests the highest execution time for all epochs (see table 6.2, column 8). Therefore, the execution time is in a trade-off with the overall performance of the system. Best f-score could be achieved using LSTM-based DL model when speed of execution is not of concern and data processing system with higher specifications is utilized.

Since the overall f-scores are significantly higher for multi-modality fusion cases as compared to the single modality cases, therefore the discussion of f-scores obtained using individual modalities for all gait activities is trivial. Hence, the hypothesis built for this research and mentioned in section 1.3 is true. The confusion matrix using DL models for all gait activities are presented in figure 6.4. Similarly, f-scores are mentioned and compared using DL models for all gait activities in figure 6.5.

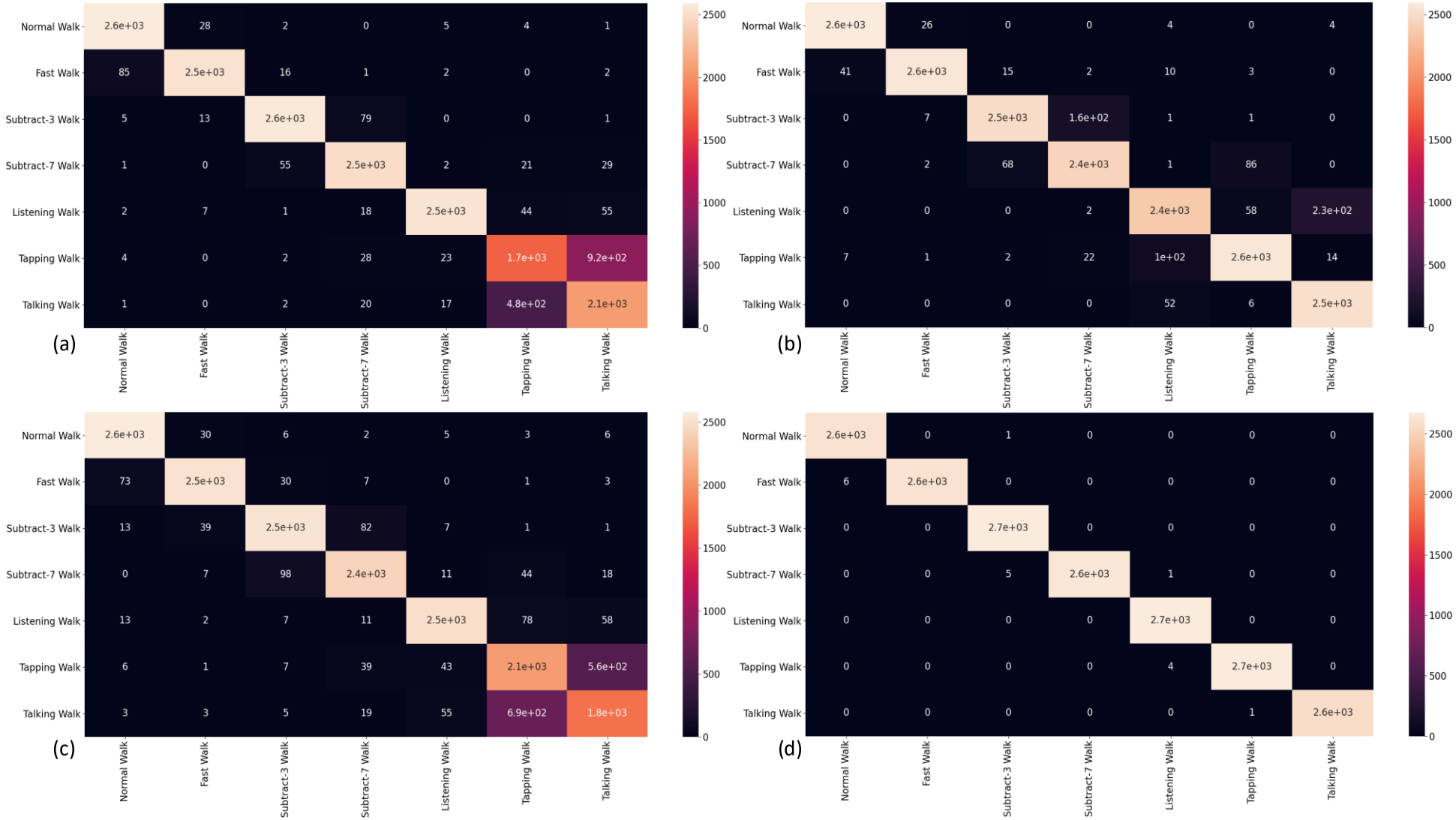


Fig 6.4 Confusion matrix for all gait activities using proposed multi-modality fusion approach and (a) FFNN, (b) 1D-CNN, (c) 2D-CNN, (d) LSTM, (epochs: 100, batch size: 120)

LSTM yielded f-scores superior to all other DL models in case of all gait activities as shown in figure 6.5. The 'typing' and 'talking' gait show worst f-score results among all activities: 64.09% (lowest) and 80.01% in case of FFNN; 75.87% and 70.09% in case of 2D-CNN. 1D-CNN model appears as the second choice due to its second highest f-scores for all gait activities, with some exceptions in 'subtracting-3'and 'listening', as well as 'subtracting-7' gait, showing lesser f-scores than 2D-CNN and FFNN, respectively. It is noticeable that 1D-CNN shows worst f-score for 'listening', which is as high as 89.03% compared to FFNN (64.09% for 'typing' gait) and 2D-CNN (70.09% for 'talking' gait). Standard deviation in model- wise f-scores of multi-modality fusion for all classes is calculated as: LSTM (0.09%), 1D-CNN (3.05%), 2D-CNN (10.18%) and FFNN (11.81%).

Furthermore, FFNN shows an overall f-score of 89.33% with minimum execution time (03min:06sec) and LSTM shows a highest f-score of 99.9% with maximum execution time (23hr:23min:45sec) (see table 6.2). From the results achieved using given resources the 1D-CNN approach appears to be the optimal approach even though the performance is 5% lower for our case. However, in the presence of higher computational resources LSTM would be the obvious choice. Therefore, 1D-CNN appears as the best DL model for overall performance for the proposed multi-modality fusion due to its performance f-score (94.97%) and a reasonable execution time (21min:14sec) to train the model (see table 6.2, columns: 7 & 8).

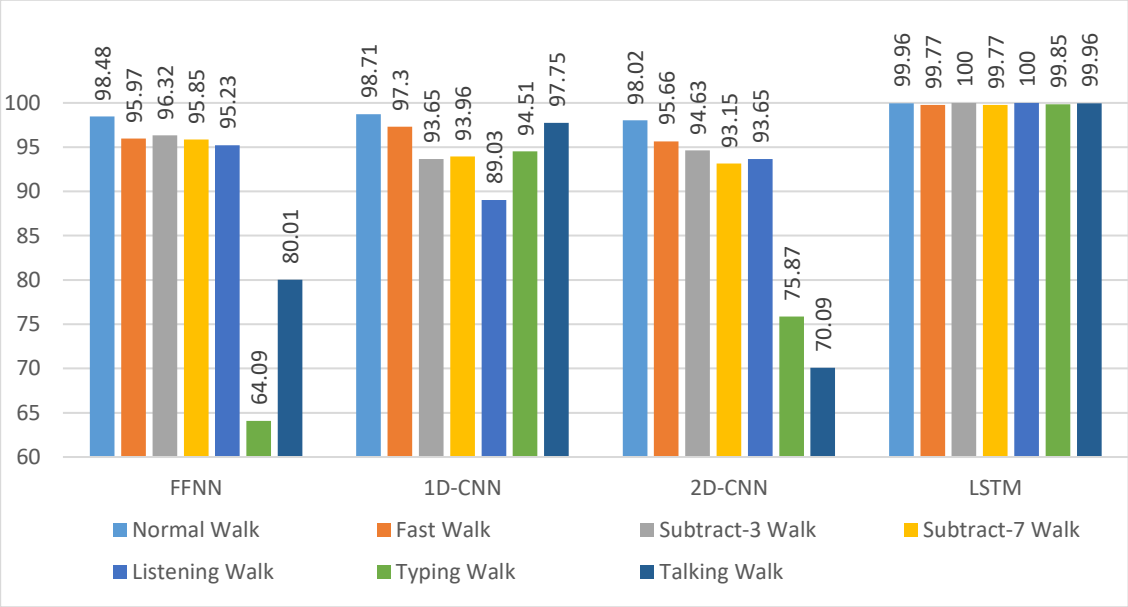


Fig 6.5 Model-wise f-scores of multi-modality fusion for all classes
(epochs: 100, batch size: 120)

Chapter 7

Conclusions

7.1 Summary

The research presented in this PhD thesis is focused on the extraction and fusion of unique gait activity features related to cognitive load and applicable for healthcare scenarios. Our methodology provides a choice of its application using either manual or automatic feature extraction whilst observing the performance of different supervised machine learning and deep learning models. Two modalities i.e., FS and AIS were utilised for independent and combined gait activity analysis.

Initially, we investigated gait activities of 11 volunteers and the plan was to extend the size of datasets to a higher volume. However, our plan had to terminate due to the world-wide outbreak of COVID-19 and UoM decided to close its premises for staff, students, and the visitors. After approximately one year, the campus re-opened for independently working staff and researchers under restricted social distancing measures and guidelines provided by the government. Therefore, it was not feasible to conduct further experiments that involved direct interaction with volunteers. This left some of the results vulnerable to concerns about statistical significance and impacted negatively on the speed of publishing our own results.

In the past, an older version of FS at UoM has been used for offline raw spatio-temporal data analysis with limited gait activities: normal, fast, and dual-task gait. In this research, we have used an upgraded version of FS to analyse gait activities in real-time with remote access. We have included a wider set of gait activities such as: normal walk, fast walk, subtracting-3, subtracting-7, listening to story, typing on the mobile phone, and talking to operator to analyse and investigate the changes in gait related to cognitive load.

During a gait activity, FS are mainly used to capture information from feet contact with the ground and represent only limited aspects of the whole-body movement. Therefore, to capture the complex nature of gait activity information more sensors need to be placed on other body parts such as heels, shanks, and knees. In this research we have used and developed the inertial sensor technology to obtain information from lower body parts during gait activities. A portable AIS system has been developed to study the effect of change in gait activities by placing one centralized inertial sensor around the pelvis and two on the heel of each foot. Inertial sensors are very easy to use and could be used for in/out of lab and long-term monitoring in healthcare scenarios.

The captured movements in the lower parts of the human body, by AIS and of foot falls by FS, in the general case are not independent from each other. Therefore, it is possible to combine the perceived coordination and complementarity of both data sources. We proposed and implemented a multi-modality sensor fusion approach to capture the complex nature of gait information.

This approach uses information from two modalities and provides a more comprehensive description of individual's gait activities. The proposed system is capable to update and synchronize the timestamped information obtained from both systems. We have created a GUI for FS to display the data related to the foot in contact with the floor. The pressure on ground results in different and unique gait patterns corresponding to different gait activities. Another GUI is created to monitor the data capture from the lower parts of human body using inertial sensors. Acceleration and angular velocity values are the parameters used to measure the movement of feet and orientation of sensors placed on the pelvis and both ankles. Further, synchronized data obtained from both modalities is visually analyzed to match for the gait parameters such as Heel strike which is more noticeable as an initial pressure on FS and a peak signal using AIS during different gait activities.

In this research, the results obtained from gait activity classification have the potential to be applied to more complex situations related to detecting changes in human gait patterns as a neurodegenerative progression function.

7.2 Conclusions

7.2.1 Sensor fusion based on feature extraction

Sensor fusion based on feature level presents itself as a desirable option, potentially in a range of applications, where the aspects of data variance are not straightforward to understand and define as features to allow accurate classification. Like their nature, FS has 116 and AIS has 18 inputs, which are disproportionate numbers. Therefore, it is a challenging task to balance and fuse this information and avoid the biasness and unfairness in results. We have proposed PCA and CCA, feature extraction methods which reduce the overall number of inputs without degrading substantially the spatio-temporal information content from the two modalities. Results obtained from PCA and CCA are classified using multiple ML models. When compared CCA appears as first choice and has the following advantages over PCA:

- CCA provides insight of the relationship between CVPs and the adjacent CVPs. PCA provides information of adjacent feature vectors only.
- CCA summarizes the features in CVPs equal to the number of inputs in lower dataset. Features are equal to the sum of inputs in all datasets using PCA.
- CCA is computationally efficient and require lesser extracted feature vectors (CVPs) to represent the original datasets as compared to PCA.
- CCA provides higher f-scores with non-linear classifiers: NB, K-NN and K-SVM with limited CVPs for fused multi-modality cases as compared to PCA.

PCA and CCA are used in conjunction with statistical methods to select the best optimal gait features suitable for the fusion task. However, feature domains containing many features increase the chances of redundancy and irrelevancy in data which is a major challenge for feature extraction methods. Our research in this area has the following contributions:

"*Multi-modality fusion of floor and ambulatory sensors for gait classification"*.
2019 IEEE 28th International Symposium on Industrial Electronics (ISIE). 1st Aug, 2019.
DOI: 10.1109/ISIE.2019.8781127

"Gait Activity Classification from Feature-Level Sensor Fusion of Multi-Modality Systems".
IEEE Sensors Journal, 5th Oct. 2020.
DOI: 10.1109/JSEN.2020.3028697

7.2.2 Sensor fusion based on DL models

Multi-modality sensor fusion based on DL is new and reports of such fusion are few, which should be interpreted in the light of scarcity of suitable datasets. We demonstrate multi-modality sensor fusion for gait activity classification using DL models. DL layers are used to balance and fuse information whilst reserving the categorical content for each gait activity. DL based models such as: FFNN, 1D-CNN, 2D-CNN and LSTM are implemented and used to fuse spatio-temporal gait activity data obtained from FS and AIS.

The automatic extraction of features from data leads to substantially more robust and accurate results as compared to the previously discussed ML techniques. Overall performance is studied in detail and reveals best f-score of 99.9% in case of LSTM and fastest execution time 3 min 06 sec in the case of FFNN for a limited dataset. Our research in this area has the following contributions:

"Multi-modality sensor fusion for gait classification using deep learning"
2020 IEEE Sensors Applications Symposium (SAS), 9-11th Mar. 2020
DOI: 10.1109/SAS48726.2020.9220037

"Gait Activity Classification using Multi-Modality Sensor Fusion: A Deep Learning Approach"

IEEE Sensors Journal, 3rd May. 2021

DOI: 10.1109/JSEN.2021.3077698

7.3 Limitations

The classification obtained using multi-source and multi-modality sensor fusion is expected to produce superior results when compared to that from a single modality. However, the choice of optimal fusion algorithms should also involve the assessment of practicality, design, built and maintenance characteristics of such complex systems.

In this research, multi-modality sensor fusion is used to map maximum spatio-temporal gait parameters involved in cognitive tasks which results in robust results as compared to a single modality approach. Assessment of the multi-modality based results show improvements in case of certain ML and DL models as discussed in chapter 6. However, the computational models may not capture new test patterns of gait activities due to their training on a limited size dataset and hence require more data from more users.

We have adapted model tuning and researched into best hyper-parameters to minimize the error during classification. However, this approach does not guarantee results in terms of interpretability. The model parameters and features despite training do not deliver the insights of the learning procedure during a gait activity. Visualization methodologies in case of large amounts of data will not be feasible. Therefore, new ideas, algorithms and improved model architectures will be required.

7.4 Future work

The desirable further steps for future implementation in the proposed study include:

- Increase in the size of dataset to 20+ users to build an adequate dataset for training and testing classification models.
- Testing of the proposed models for further activities involving external factors such as carrying physical weight, wearing a certain type of shoe or balancing a glass of water while walking to verify the performance of the proposed methodology for a greater range of human gait activities.
- Additional inertial sensors on the upper part of human body to observe the movements of hands, elbows, arms, shoulders and head during the proposed gait activities to prepare a high-volume dataset.
- Current datasets could be used to explore the Kernel based PCA and CCA methods which are the extension of the proposed methods.
- Implantation of CCA could be combined with DL models in which the only CVPs will be used as inputs to the DL model rather raw inputs. This might lead to substantial improvements in classification f-scores when dealing with larger datasets. However, this approach will require powerful computational resources as compared to the proposed study (mentioned in section 5.5).

Bibliography

[1] G. Qian, J. Zhang and A. Kidané, "People Identification Using Floor Pressure Sensing and Analysis," in IEEE Sensors Journal, vol. 10, no. 9, pp. 1447-1460, Sept. 2010, doi: 10.1109/JSEN.2010.2045158.

[2] N. Margiotta, G. Avitabile and G. Coviello, "A wearable wireless system for gait analysis for early diagnosis of Alzheimer and Parkinson disease," 2016 5th International Conference on Electronic Devices, Systems and Applications (ICEDSA), 2016, pp. 1-4, doi: 10.1109/ICEDSA.2016.7818553.

[3] A. K. Jain, K. Nandakumar, and A. Ross, “50 years of biometric research: Accomplishments, challenges, and opportunities,” Pattern Recognit. Lett., vol. 79, pp. 80–105, Aug. 2016, doi: 10.1016/j.patrec.2015.12.013.

[4] H. Lee, S. J. Sullivan, and A. G. Schneiders, “The use of the dual-task paradigm in detecting gait performance deficits following a sports-related concussion: A systematic review and meta-analysis,” Journal of Science and Medicine in Sport, vol. 16, no. 1. pp. 2–7, Jan. 2013, doi: 10.1016/j.jsams.2012.03.013.

[5] A. W. Priest, K. B. Salamon, and J. H. Hollman, “Age-related differences in dual task walking: a cross sectional study,” Journal of neuroengineering and rehabilitation, vol. 5, 29, 14 Nov. 2008, doi:10.1186/1743-0003-5-29

[6] O. Costilla-Reyes, P. Scully, and K. B. Ozanyan, “Age-sensitive differences in single and dual walking tasks from footprint floor sensor data,” in 2017 IEEE SENSORS, pp. 1–3, Oct. 2017, doi: 10.1109/ICSENS.2017.8234299.

[7] J. M. Hausdorff, A. Schweiger, T. Herman, G. Yogev-Seligmann, and N. Giladi, “Dual-task decrements in gait: Contributing factors among healthy older adults,” Journals Gerontol. - Ser. A Biol. Sci. Med. Sci., vol. 63, no. 12, pp. 1335–1343, 2008, doi: 10.1093/gerona/63.12.1335.

[8] G. Yogev-Seligmann, J. M. Hausdorff, and N. Giladi, “The role of executive function and attention in gait,” Movement Disorders, vol. 23, no. 3. Mov Disord, pp. 329–342, Feb. 15, 2008, doi: 10.1002/mds.21720.

[9] A. S. Alharthi, S. U. Yunas, and K. B. Ozanyan, “Deep Learning for Monitoring of Human Gait: A Review,” IEEE Sens. J., pp. 1–1, Jul. 2019, doi: 10.1109/jsen.2019.2928777.

[10] J. A. Cantoral-Ceballos, P. Wright, J. Vaughan, P. Scully, and K. B. Ozanyan, “Real-time reconstruction of footprint positions using an 'intelligent carpet' imaging sensor,” in 2015 IEEE SENSORS, Nov. 2015, pp. 1–4, doi: 10.1109/ICSENS.2015.7370685.

[11] M. W. Whittle and M. W. Whittle, “Normal gait,” Gait Anal., pp. 47–100, Jan. 2007, doi: 10.1016/B978-075068883-3.50007-6.

[12] A. Muro-de-la-Herran, B. García-Zapirain, and A. Méndez-Zorrilla, “Gait analysis methods: An overview of wearable and non-wearable systems, highlighting clinical applications,” Sensors (Switzerland), vol. 14, no. 2. Multidisciplinary Digital Publishing Institute (MDPI), pp. 3362–3394, Feb. 19, 2014, doi: 10.3390/s140203362.

[13] M. W. Whittle and M. W. Whittle, “Normal gait,” Gait Anal., pp. 47–100, Jan. 2007, doi: 10.1016/B978-075068883-3.50007-6.

[14] E. Abdulhay, N. Arunkumar, K. Narasimhan, E. Vellaiappan, and V. Venkatraman, “Gait and tremor investigation using machine learning techniques for the diagnosis of Parkinson disease,” Futur. Gener. Comput. Syst., vol. 83, pp. 366–373, Jun. 2018, doi: 10.1016/j.future.2018.02.009.

[15] S. Chakraborty, A. Jamthe, S. K. Ghosh, and D. P. Agrawal, “From theory to application: Wireless monitoring of patients suffering from neurodegenerative diseases,” in Midwest Symposium on Circuits and Systems, 2013, pp. 944–947, doi: 10.1109/MWSCAS.2013.6674806.

[16] G. Bastas, J. J. Fleck, R. A. Peters, and K. E. Zelik, “IMU-based gait analysis in lower limb prosthesis users: Comparison of step demarcation algorithms,” Gait Posture, vol. 64, pp. 30–37, Jul. 2018, doi: 10.1016/j.gaitpost.2018.05.025.

[17] C. Yan, B. Zhang, and F. Coenen, “Multi-attributes gait identification by

convolutional neural networks," in Proceedings - 2015 8th International Congress on Image and Signal Processing, CISP 2015, Feb. 2016, pp. 642–647, doi: 10.1109/CISP.2015.7407957.

[18] Z. Wu, Y. Huang, L. Wang, X. Wang, and T. Tan, "A Comprehensive Study on Cross-View Gait Based Human Identification with Deep CNNs," IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 2, pp. 209–226, Feb. 2017, doi: 10.1109/TPAMI.2016.2545669.

[19] F. Battistone and A. Petrosino, "TGLSTM: A time based graph deep learning approach to gait recognition," Pattern Recognit. Lett., vol. 126, pp. 132–138, Sep. 2019, doi: 10.1016/j.patrec.2018.05.004.

[20] J. Suutala, S. Pirttikangas, J. Riekki, and J. Röning, "Reject-optional LVQ-based two-level classifier to improve reliability in footstep identification," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 3001, pp. 182–187, 2004, doi: 10.1007/978-3-540-24646-6_12.

[21] D. Gouwanda and S. M. N. A. Senanayake, "Emerging Trends of Body-Mounted Sensors in Sports and Human Gait Analysis," 4th Kuala Lumpur International Conference on Biomedical Engineering 2008, vol. 21, pp. 715–718, Jan 2008, doi: 10.1007/978-3-540-69139-6_178.

[22] M. Yoneyama, Y. Kurihara, K. Watanabe, and H. Mitoma, "Accelerometry-based gait analysis and its application to parkinson's disease assessment-Part 2 : A new measure for quantifying walking behavior," IEEE Trans. Neural Syst. Rehabil. Eng., vol. 21, no. 6, pp. 999–1005, 2013, doi: 10.1109/TNSRE.2013.2268251.

[23] C. Strohrmann, H. Harms, C. Kappeler-Setz, and G. Tröster, "Monitoring kinematic changes with fatigue in running using body-worn sensors," IEEE Trans. Inf. Technol. Biomed., vol. 16, no. 5, pp. 983–990, 2012, doi: 10.1109/TITB.2012.2201950.

[24] M. Destephe, T. Maruyama, M. Zecca, K. Hashimoto, and A. Takanishi, "The influences of emotional intensity for happiness and sadness on walking," in Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS, pp. 7452–7455, 2013, doi:

10.1109/EMBC.2013.6611281.

[25] Y.-C. Liu, Y.-R. Yang, Y.-A. Tsai, R.-Y. Wang, and C.-F. Lu, “Brain Activation and Gait Alteration During Cognitive and Motor Dual Task Walking in Stroke—A Functional Near-Infrared Spectroscopy Study,” IEEE Trans. Neural Syst. Rehabil. Eng., vol. 26, no. 12, pp. 2416–2423, Dec. 2018, doi: 10.1109/TNSRE.2018.2878045.

[26] N. Samudin, W. N. M. Isa, T. H. Maul, and W. K. Lai, “Analysis of Gait Features between Loaded and Normal Gait,” in 2009 Fifth International Conference on Signal Image Technology and Internet Based Systems, pp. 172–179, Nov. 2009, doi: 10.1109/SITIS.2009.37.

[27] M. Woollacott and A. Shumway-Cook, “Attention and the control of posture and gait: A review of an emerging area of research,” Gait and Posture, vol. 16, no. 1. Elsevier, pp. 1–14, Aug. 01, 2002, doi: 10.1016/S0966-6362(01)00156-4.

[28] S. O’Shea, M. E. Morris, and R. Iansek, “Dual Task Interference During Gait in People With Parkinson Disease: Effects of Motor Versus Cognitive Secondary Tasks,” Phys. Ther., vol. 82, no. 9, pp. 888–897, Sep. 2002, doi: 10.1093/ptj/82.9.888.

[29] T. Herman, A. Mirelman, N. Giladi, A. Schweiger, and J. M. Hausdorff, “Executive control deficits as a prodrome to falls in healthy older adults: A prospective study linking thinking, walking, and falling,” Journals Gerontol. - Ser. A Biol. Sci. Med. Sci., vol. 65 A, no. 10, pp. 1086–1092, Oct. 2010, doi: 10.1093/gerona/glq077.

[30] T. IJmker and C. J. C. Lamoth, “Gait and cognition: The relationship between gait stability and variability with executive function in persons with and without dementia,” Gait Posture, vol. 35, no. 1, pp. 126–130, Jan. 2012, doi: 10.1016/j.gaitpost.2011.08.022.

[31] J. A. Cantoral-Ceballos et al., “Intelligent carpet system, based on photonic guided-path tomography, for gait and balance monitoring in home environments,” IEEE Sens. J., vol. 15, no. 1, pp. 279–289, Jan. 2015, doi: 10.1109/JSEN.2014.2341455.

[32] O. Costilla-Reyes, P. Scully, and K. B. Ozanyan, “Deep Neural Networks for Learning Spatio-Temporal Features From Tomography Sensors,” IEEE Trans. Ind. Electron., vol. 65, no. 1, pp. 645–653, Jan. 2018, doi: 10.1109/TIE.2017.2716907.

[33] T. Zebin, M. Sperrin, N. Peek, and A. J. Casson, “Human activity recognition from inertial sensor time-series using batch normalized deep LSTM recurrent networks,” in *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, pp. 1–4, Oct. 2018, doi: 10.1109/EMBC.2018.8513115.

[34] T. Zebin, P. J. Scully, N. Peek, A. J. Casson, and K. B. Ozanyan, “Design and Implementation of a Convolutional Neural Network on an Edge Computing Smartphone for Human Activity Recognition,” *IEEE Access*, vol. 7, pp. 133509–133520, 2019, doi: 10.1109/ACCESS.2019.2941836.

[35] P. Connor and A. Ross, “Biometric recognition by gait: A survey of modalities and features,” *Comput. Vis. Image Underst.*, vol. 167, pp. 1–27, Feb. 2018, doi: 10.1016/j.cviu.2018.01.007.

[36] A. Muro-de-la-Herran, B. García-Zapirain, and A. Méndez-Zorrilla, “Gait analysis methods: An overview of wearable and non-wearable systems, highlighting clinical applications,” *Sensors (Switzerland)*, vol. 14, no. 2. Multidisciplinary Digital Publishing Institute (MDPI), pp. 3362–3394, Feb. 19, 2014, doi: 10.3390/s140203362.

[37] B. Gálai and C. Benedek, "Feature selection for Lidar-based gait recognition," *2015 International Workshop on Computational Intelligence for Multimedia Understanding (IWCIM)*, pp. 1-5, 2015, doi: 10.1109/IWCIM.2015.7347076.

[38] Mohammed Ahmed, Naseer Al-Jawad, and Azhin T. Sabir "Gait recognition based on Kinect sensor", in *Proc. SPIE 9139, Real-Time Image and Video Processing 2014*, vol. 91390, pp. 63-72, 15 May 2014, doi:10.1117/12.2052588

[39] E. Hossain and G. Chetty, “Multimodal feature learning for gait biometric based human identity recognition,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8227 LNCS, no. PART 2, pp. 721–728, 2013, doi: 10.1007/978-3-642-42042-9_89.

[40] W. Kusakunniran, "Recognizing Gaits on Spatio-Temporal Feature Domain," *in IEEE Transactions on Information Forensics and Security*, vol. 9, no. 9, pp. 1416-1423, Sept. 2014, doi: 10.1109/TIFS.2014.2336379.

[41] "The CMU Motion of Body (MoBo) Database - The Robotics Institute Carnegie Mellon University." https://www.ri.cmu.edu/publications/the-cmu-motion-of-body-mobo-database/ (accessed Sep. 18, 2021).

[42] "Center for Biometrics and Security Research." http://www.cbsr.ia.ac.cn/english/Gait Databases.asp (accessed Sep. 18, 2021).

[43] Y. Makihara et al., "The OU-ISIR gait database comprising the treadmill dataset," *IPSJ Trans. Comput. Vis. Appl.*, vol. 4, pp. 53–62, 2012, doi: 10.2197/ipsjtcva.4.53.

[44] H. Iwama, M. Okumura, Y. Makihara, and Y. Yagi, "The OU-ISIR gait database comprising the large population dataset and performance evaluation of gait recognition," *IEEE Trans. Inf. Forensics Secur.*, vol. 7, no. 5, pp. 1511–1521, 2012, doi: 10.1109/TIFS.2012.2204253.

[45] M. Hofmann, J. Geiger, S. Bachmann, B. Schuller, and G. Rigoll, "The TUM Gait from Audio, Image and Depth (GAID) database: Multimodal recognition of subjects and traits," *J. Vis. Commun. Image Represent.*, vol. 25, no. 1, pp. 195–206, Jan. 2014, doi: 10.1016/j.jvcir.2013.02.006.

[46] S. Sarkar, P. J. Phillips, Z. Liu, I. R. Vega, P. Grother, and K. W. Bowyer, "The humanID gait challenge problem: Data sets, performance, and analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 2, pp. 162–177, Feb. 2005, doi: 10.1109/TPAMI.2005.39.

[47] W. Tao, T. Liu, R. Zheng, and H. Feng, "Gait Analysis Using Wearable Sensors," *Sensors (Basel).*, vol. 12, no. 2, p. 2255, 2012, doi: 10.3390/S120202255.

[48] W. Kong, S. Sessa, M. Zecca, and A. Takanishi, "Anatomical Calibration through Post-Processing of Standard Motion Tests Data," *Sensors*, vol. 16, no. 12, p. 2011, Nov. 2016, doi: 10.3390/s16122011.

[49] H. Chan, H. Zheng, H. Wang, and D. Newell, "Assessment of gait patterns of chronic low back pain patients: A smart mobile phone based approach," *in Proceedings - 2015 IEEE International Conference on Bioinformatics and Biomedicine*, BIBM 2015, Dec. 2015, pp. 1016–1023, doi: 10.1109/BIBM.2015.7359823.

[50] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Comput. Surv.*, vol. 46, no. 3, pp. 1–33,

Jan. 2014, doi: 10.1145/2499621.

[51] G. Pacini Panebianco, M. C. Bisi, R. Stagni, and S. Fantozzi, “Analysis of the performance of 17 algorithms from a systematic review: Influence of sensor position, analysed variable and computational approach in gait timing estimation from IMU measurements,” *Gait Posture*, vol. 66, pp. 76–82, Oct. 2018, doi: 10.1016/J.GAITPOST.2018.08.025.

[52] S. R. Hundza et al., “Accurate and reliable gait cycle detection in parkinson’s disease,” *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 22, no. 1, pp. 127–137, Jan. 2014, doi: 10.1109/TNSRE.2013.2282080.

[53] A. Saad, F. Guerin, I. Zaarour, M. Ayache, and D. Lefebvre, “Sensoring and features extraction for the detection of Freeze of Gait in Parkinson disease,” *in 2014 IEEE 11th International Multi-Conference on Systems, Signals & Devices (SSD14)*, pp. 1–6, Feb. 2014, doi: 10.1109/SSD.2014.6808786.

[54] T. Zebin, P. J. Scully and K. B. Ozanyan, "Human activity recognition with inertial sensors using a deep learning approach," *2016 IEEE SENSORS*, pp. 1-3, 2016, doi: 10.1109/ICSENS.2016.7808590.

[55] O. Dehzangi, M. Taherisadr, and R. ChangalVala, “IMU-based gait recognition using convolutional neural networks and multi-sensor fusion,” *Sensors (Switzerland)*, vol. 17, no. 12, p. 2735, Dec. 2017, doi: 10.3390/s17122735.

[56] J. A. Cantoral-Ceballos et al., “Intelligent carpet system, based on photonic guided-path tomography, for gait and balance monitoring in home environments,” *IEEE Sens. J.*, vol. 15, no. 1, pp. 279–289, Jan. 2015, doi: 10.1109/JSEN.2014.2341455.

[57] L. Middleton, A. A. Buss, A. Bazin, and M. S. Nixon, “A Floor Sensor System for Gait Recognition,” *Fourth IEEE Workshop on Automatic Identification Advanced Technologies (AutoID’05)*, pp. 171–176, doi: 10.1109/AUTOID.2005.2.

[58] R. Vera-Rodriguez, J. S. D. Mason, J. Fierrez, and J. Ortega-Garcia, “Comparative analysis and fusion of spatiotemporal information for footstep recognition,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 4, pp. 823–834, 2013, doi: 10.1109/TPAMI.2012.164.

[59] A. Bertani, A. Cappello, M. G. Benedetti, L. Simoncini, and F. Catani, “Flat foot functional evaluation using pattern recognition of ground reaction data,” *Clin.*

Biomech., vol. 14, no. 7, pp. 484–493, 1999, doi: 10.1016/S0268-0033(98)90099-7.

[60] P. Leusmann, C. Möllering, L. Klack, K. Kasugai, M. Ziefle, and B. Rumpe, "Your floor knows where you are: Sensing and acquisition of movement data," in *Proceedings - IEEE International Conference on Mobile Data Management*, vol. 2, pp. 61–66, 2011, doi: 10.1109/MDM.2011.29.

[61] J. H. Hollman, F. M. Kovash, J. J. Kubik, and R. A. Linbo, "Age-related differences in spatiotemporal markers of gait stability during dual task walking," *Gait Posture*, vol. 26, no. 1, pp. 113–119, Jun. 2007, doi: 10.1016/j.gaitpost.2006.08.005.

[62] M. S. Singh, V. Pondenkandath, B. Zhou, P. Lukowicz, and M. Liwickit, "Transforming sensor data to the image domain for deep learning - An application to footstep detection," in *Proceedings of the International Joint Conference on Neural Networks*, vol. 2017-May, pp. 2665–2672, Jun. 2017, doi: 10.1109/IJCNN.2017.7966182.

[63] O. Costilla-Reyes, P. Scully, and K. B. Ozanyan, "Temporal Pattern Recognition in Gait Activities Recorded with a Footprint Imaging Sensor System," *IEEE Sens. J.*, vol. 16, no. 24, pp. 8815–8822, Dec. 2016, doi: 10.1109/JSEN.2016.2583260.

[64] D. L. Hall and J. Llinas, "An introduction to multisensor data fusion," in *Proc. IEEE*, vol. 85, no. 1, pp. 6–23, 1997, doi: 10.1109/5.554205.

[65] A. S. Alharthi, S. U. Yunas, and K. B. Ozanyan, "Deep Learning for Monitoring of Human Gait: A Review," *IEEE Sens. J.*, pp. 1–1, Jul. 2019, doi: 10.1109/jsen.2019.2928777.

[66] G. Shakhnarovich, L. Lee, and T. Darrell, "Integrated face and gait recognition from multiple views," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, pp. I-439-I–446, 2001, doi: 10.1109/CVPR.2001.990508.

[67] X. Zhou and B. Bhanu, "Integrating Face and Gait for Human Recognition at a Distance in Video," *IEEE Trans. Syst. Man Cybern. Part B*, vol. 37, no. 5, pp. 1119–1137, Oct. 2007, doi: 10.1109/TSMCB.2006.889612.

[68] E. Vildjiounaite *et al.*, "Unobtrusive Multimodal Biometrics for Ensuring Privacy and Information Security with Personal Devices," in *Proceedings of the 4th*

international conference on Pervasive Computing, pp. 187–201, 2006, doi: 10.1007/11748625_12.

[69] R. Vera-Rodriguez, J. Fierrez, J. S. D. Mason and J. Ortega-Garcia, "A novel approach of gait recognition through fusion with footstep information," *2013 International Conference on Biometrics (ICB)*, pp. 1-6, 2013, doi: 10.1109/ICB.2013.6613014.

[70] E. Ribeiro, A. Uhl, and F. Alonso-Fernandez, "Iris super-resolution using CNNs: Is photo-realism important to iris recognition?," *IET Biometrics*, vol. 8, no. 1, pp. 69–78, Jan. 2019, doi: 10.1049/iet-bmt.2018.5146.

[71] R. Raghavendra, K. B. Raja, S. Venkatesh, and C. Busch, "Transferable Deep-CNN Features for Detecting Digital and Print-Scanned Morphed Face Images," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, Aug. 2017, vol. 2017-July, pp. 1822–1830, doi: 10.1109/CVPRW.2017.228.

[72] F. Wu, J. Zhu, and X. Guo, "Fingerprint pattern identification and classification approach based on convolutional neural networks," *Neural Comput. Appl.*, vol. 32, no. 10, pp. 5725–5734, May 2020, doi: 10.1007/s00521-019-04499-w.

[73] S. F. Chevtchenko, R. F. Vale, and V. Macario, "Multi-objective optimization for hand posture recognition," *Expert Syst. Appl.*, vol. 92, pp. 170–181, Feb. 2018, doi: 10.1016/j.eswa.2017.09.046.

[74] R. Donida Labati, E. Muñoz, V. Piuri, R. Sassi, and F. Scotti, "Deep-ECG: Convolutional Neural Networks for ECG biometric recognition," *Pattern Recognit. Lett.*, vol. 126, pp. 78–85, Sep. 2019, doi: 10.1016/j.patrec.2018.03.028.

[75] E. P. Ijjina and K. M. Chalavadi, "Human action recognition in RGB-D videos using motion sequence information and deep learning," *Pattern Recognit.*, vol. 72, pp. 504–516, Dec. 2017, doi: 10.1016/j.patcog.2017.07.013.

[76] Z. Wu, Y. Huang, L. Wang, X. Wang, and T. Tan, "A Comprehensive Study on Cross-View Gait Based Human Identification with Deep CNNs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 2, pp. 209–226, Feb. 2017, doi: 10.1109/TPAMI.2016.2545669.

[77] O. Mazumder, A. S. Kundu, P. K. Lenka, and S. Bhaumik, "Multi-channel Fusion

Based Adaptive Gait Trajectory Generation Using Wearable Sensors," *J. Intell. Robot. Syst. Theory Appl.*, vol. 86, no. 3–4, pp. 335–351, Jun. 2017, doi: 10.1007/s10846-016-0436-y.

[78] Z. Ding *et al.*, "The Real Time Gait Phase Detection Based on Long Short-Term Memory," in *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, pp. 33–38, Jun. 2018, doi: 10.1109/DSC.2018.00014.

[79] K. R. Mun, G. Song, S. Chun, and J. Kim, "Gait Estimation from Anatomical Foot Parameters Measured by a Foot Feature Measurement System using a Deep Neural Network Model," *Sci. Rep.*, vol. 8, no. 1, Dec. 2018, doi: 10.1038/s41598-018-28222-2.

[80] H. T. T. Vu, F. Gomez, P. Cherelle, D. Lefeber, A. Nowé, and B. Vanderborght, "ED-FNN: A new deep learning algorithm to detect percentage of the gait cycle for powered prostheses," *Sensors (Switzerland)*, vol. 18, no. 7, Jul. 2018, doi: 10.3390/s18072389.

[81] G. Chalvatzaki, P. Koutras, J. Hadfield, X. S. Papageorgiou, C. S. Tzafestas, and P. Maragos, "LSTM-based network for human gait stability prediction in an intelligent robotic rollator," in *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 4225–4232, May 2019, doi: 10.1109/ICRA.2019.8793899.

[82] P. Kumar, S. Mukherjee, R. Saini, P. Kaushik, P. P. Roy, and D. P. Dogra, "Multimodal Gait Recognition With Inertial Sensor Data and Video Using Evolutionary Algorithm," *IEEE Trans. Fuzzy Syst.*, vol. 27, no. 5, pp. 956–965, May 2019, doi: 10.1109/TFUZZ.2018.2870590.

[83] K. Ivanov *et al.*, "Identity Recognition by Walking Outdoors Using Multimodal Sensor Insoles," *IEEE Access*, vol. 8, pp. 150797–150807, 2020, doi: 10.1109/ACCESS.2020.3016970.

[84] A. L. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," *IBM J. Res. Dev.*, vol. 3, no. 3, pp. 210–229, Jul. 1959, doi: 10.1147/RD.33.0210.

[85] A. Ross and A. Jain, "Information fusion in biometrics," *Pattern Recognit. Lett.*, vol. 24, no. 13, pp. 2115–2125, Sep. 2003, doi: 10.1016/S0167-8655(03)00079-5.

[86] "Applied linear regression." http://www.ru.ac.bd/wp-content/uploads/sites/25/2019/03/304_03_Weisberg-Applied-Linear-Regression-

Wiley-2013.pdf (accessed: Dec. 06, 2021).

[87] “Introduction to Logistic Regression | by Ayush Pant | Towards Data Science.” https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148 (accessed Jun. 20, 2021).

[88] “A Quick Introduction to K-Nearest Neighbors Algorithm | by Adi Bronshtein | Noteworthy - The Journal Blog.” https://blog.usejournal.com/a-quick-introduction-to-k-nearest-neighbors-algorithm-62214cea29c7 (accessed Jun. 12, 2021).

[89] “Tutorial on Support Vector Machine (SVM) | Semantic Scholar.” https://www.semanticscholar.org/paper/Tutorial-on-Support-Vector-Machine-(-SVM-)-Jakkula/7cc83e98367721bfb908a8f703ef5379042c4bd9 (accessed Jun. 20, 2021).

[90] “Introduction to Naive Bayes Classifier | by Priyanka Meena | Nov, 2020 | Medium | Towards Data Science.” https://towardsdatascience.com/introduction-to-naive-bayes-classifier-f5c202c97f92 (accessed Jun. 20, 2021).

[91] “Decision Trees: A Complete Introduction | by Alan Jeffares | Towards Data Science.” https://towardsdatascience.com/decision-trees-60707f06e836 (accessed Jun. 20, 2021).

[92] J. Shotton *et al.*, “Real-time human pose recognition in parts from single depth images,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1297–1304, 2011, doi: 10.1109/CVPR.2011.5995316.

[93] “Neural Networks and Deep Learning," http://neuralnetworksanddeeplearning.com (accessed: Jun. 03, 2019).

[94] Z. Wang and A. C. Bovik, “Mean squared error: Lot it or leave it? A new look at signal fidelity measures,” *IEEE Signal Process. Mag.*, vol. 26, no. 1, pp. 98–117, 2009, doi: 10.1109/MSP.2008.930649.

[95] J. E. Shore and R. W. Johnson, “Axiomatic Derivation of the Principle of Maximum Entropy and the Principle of Minimum Cross-Entropy,” *IEEE Trans. Inf. Theory*, vol. 26, no. 1, pp. 26–37, 1980, doi: 10.1109/TIT.1980.1056144.

[96] L. Bottou, “Large-Scale Machine Learning with Stochastic Gradient Descent,”

2010, Accessed: Oct. 19, 2020. [Online]. Available: https://leon.bottou.org/publications/pdf/compstat-2010.pdf.

[97] M. F. Møller, “A scaled conjugate gradient algorithm for fast supervised learning,” *Neural Networks*, vol. 6, no. 4, pp. 525–533, Jan. 1993, doi: 10.1016/S0893-6080(05)80056-5.

[98] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” Dec. 2015, Accessed: Oct. 19, 2020. [Online]. Available: https://arxiv.org/abs/1412.6980v9.

[99] A. Elhassouny and F. Smarandache, “Trends in deep convolutional neural Networks architectures: a review,” *2019 International Conference of Computer Science and Renewable Energies (ICCSRE)*, pp. 1–8, Aug. 2019, doi: 10.1109/iccsre.2019.8807741.

[100] P. J. Werbos, “Generalization of backpropagation with application to a recurrent gas market model,” *Neural Networks*, vol. 1, no. 4, pp. 339–356, Jan. 1988, doi: 10.1016/0893-6080(88)90007-X.

[101] J. L. Elman, “Finding structure in time,” *Cogn. Sci.*, vol. 14, no. 2, pp. 179–211, Apr. 1990, doi: 10.1016/0364-0213(90)90002-E.

[102] R. Pascanu, T. Mikolov, and Y. Bengio, “On the difficulty of training Recurrent Neural Networks,” *30th Int. Conf. Mach. Learn. ICML 2013*, no. PART 3, pp. 2347–2355, Nov. 2012, Accessed: Jul. 01, 2020. [Online]. Available: http://arxiv.org/abs/1211.5063.

[103] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[104] O. Costilla-Reyes, P. Scully, and K. B. Ozanyan, “Age-sensitive differences in single and dual walking tasks from footprint floor sensor data,” *2017 IEEE SENSORS*, pp. 1–3, Oct. 2017, doi: 10.1109/ICSENS.2017.8234299.

[105] L. Middleton, A. A. Buss, A. Bazin and M. S. Nixon, "A floor sensor system for gait recognition," *Fourth IEEE Workshop on Automatic Identification Advanced Technologies (AutoID'05)*, pp. 171-176, 2005, doi: 10.1109/AUTOID.2005.2.

[106] T. Nikazad, “The Use of Landweber Algorithm in Image Reconstruction,” 2007,

Accessed: Oct. 19, 2020. [Online]. Available: https://www.diva-portal.org/smash/get/diva2:16771/FULLTEXT01.pdf.

[107] "Getting Started with Arduino UNO." https://www.arduino.cc/en/Guide/ArduinoUno (accessed Dec. 25, 2021).

[108] "Raspberry Pi - Wikipedia." https://en.wikipedia.org/wiki/Raspberry_Pi (accessed May 21, 2021).

[109] "Sense HAT - Raspberry Pi Documentation." https://www.raspberrypi.org/documentation/hardware/sense-hat/ (accessed Sep. 28, 2019).

[110] "9DoF Razor IMU M0 Hookup Guide - learn.sparkfun.com." https://tinyurl.com/2p8rd2sj (accessed Sep. 28, 2019).

[111] N. A. Capela, E. D. Lemaire, N. Baddour, M. Rudolf, N. Goljar, and H. Burger, "Evaluation of a smartphone human activity recognition application with able-bodied and stroke participants," *J. Neuroeng. Rehabil.*, vol. 13, no. 1, p. 5, Jan. 2016, doi: 10.1186/s12984-016-0114-0.

[112] S. Kour, R. Kumar, and M. Gupta, "Analysis of student performance using Machine learning Algorithms," *Proc. 3rd Int. Conf. Inven. Res. Comput. Appl. ICIRCA 2021*, pp. 1395–1403, Sep. 2021, doi: 10.1109/ICIRCA51532.2021.9544935.

[113] A. Duivenvoorden, K. Lee, M. Raison, and S. Achiche, "Sensor fusion in upper limb area networks: A survey," in *2017 Global Information Infrastructure and Networking Symposium, GIIS 2017*, vol. 2017-Decem, pp. 56–63, Dec. 2017, doi: 10.1109/GIIS.2017.8169802.

[114] I. M. Pires, G. Marques, N. M. Garcia, F. Flórez-Revuelta, M. Canavarro Teixeira, E. Zdravevski, S. Spinsante, and M. Coimbra, "Pattern Recognition Techniques for the Identification of Activities of Daily Living Using a Mobile Device Accelerometer," *Electronics 2020*, vol. 9, no. 3, p. 509, Mar. 2020. doi: 10.3390/electronics9030509.

[115] R. Gravina, P. Alinia, H. Ghasemzadeh, and G. Fortino, "Multi-sensor fusion in body sensor networks: State-of-the-art and research challenges," *Inf. Fusion*, vol. 35, pp. 1339–1351, May 2017, doi: 10.1016/j.inffus.2016.09.005.

[116] H. Zhang, G. Liu, T. W. S. Chow, and W. Liu, "Textual and visual content-based anti-phishing: A Bayesian approach," *IEEE Trans. Neural Networks*, vol. 22, no. 10, pp. 1532–1546, Oct. 2011, doi: 10.1109/TNN.2011.2161999.

[117] S. A. Khamseh and A. Fatehi, "Performance monitoring of heavy duty gas turbines based on Bayesian and Dempster-Shafer theory," in *Proceedings of 2017 International Conference on Electrical and Information Technologies, ICEIT 2017*, vol. 2018-January, pp. 1–7, Jan. 2018, doi: 10.1109/EITech.2017.8255303.

[118] G. Safont, A. Salazar, and L. Vergara, "New applications of late fusion methods for EEG signal processing," in *Proceedings - 6th Annual Conference on Computational Science and Computational Intelligence, CSCI 2019*, pp. 617–621, Dec. 2019, doi: 10.1109/CSCI49370.2019.00116.

[119] S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," in *Proceedings of 2014 Science and Information Conference, SAI 2014*, pp. 372–378, Oct. 2014, doi: 10.1109/SAI.2014.6918213.

[120] S. Sharma, "Applied Multivariate Tecnhiques," *Univ. South Carolina*, pp. 1–493, 1995.

[121] V. H. Patil, S. N. Singh, S. Mishra, and D. Todd Donavan, "Efficient theory development and factor retention criteria: Abandon the 'eigenvalue greater than one' criterion," *J. Bus. Res.*, vol. 61, no. 2, pp. 162–170, Feb. 2008, doi: 10.1016/j.jbusres.2007.05.008.

[122] "Lesson 13: Canonical Correlation Analysis" https://online.stat.psu.edu/stat505/lesson/13 (accessed Apr. 24, 2019).

[123] P. B. Patnaik, "The Non-Central χ 2 - and F-Distribution and their Applications," *Biometrika*, vol. 36, no. 1/2, p. 202, Jun. 1949, doi: 10.2307/2332542.

[124] T. Dahiru, "P-Value, a true test of statistical significance? a cautionary note," *Ann. Ibadan Postgrad. Med.*, vol. 6, no. 1, p. 21, Mar. 2011, doi: 10.4314/aipm.v6i1.64038.

[125] Y. Li, J. Cheng, X. Ji, W. Feng, and D. Tao, "Real-time action recognition by feature-level fusion of depth and inertial sensor," in *2017 IEEE International Conference on Real-Time Computing and Robotics, RCAR 2017*, vol. 2017-July,

pp. 109–114, Mar. 2018, doi: 10.1109/RCAR.2017.8311844.

[126] R. Kohavi, “A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection,” 1995. Accessed: Aug. 19, 2020. [Online]. Available: http://ai.stanford.edu/~ronnyk/accEst.pdf.

[127] V. Nair and G. E. Hinton, “Rectified Linear Units Improve Restricted Boltzmann Machines,” in *Proceedings of the 27th International Conference on International Conference on Machine Learning (ICML'10). Omnipress*, pp. 807-814, 2010, doi: 10.4314/aipm.v6i1.64038.

[128] Y. A. LeCun, L. Bottou, G. B. Orr, and K. R. Müller, “Efficient backprop,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7700, pp. 9–48, 2012, doi: 10.1007/978-3-642-35289-8_3.

[129] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” in *Proc. IEEE*, 1998, doi: 10.1109/5.726791.

[130] I. Yeo and K. Balachandran, "Sentiment Analysis on Time-Series Data Using Weight Priority Method on Deep Learning," *2019 International Conference on Data Science and Communication (IconDSC)*, pp. 1-7, 2019, doi: 10.1109/IconDSC.2019.8816985.

[131] W. Feng, N. Guan, Y. Li, X. Zhang, and Z. Luo, “Audio visual speech recognition with multimodal recurrent neural networks,” in *Proceedings of the International Joint Conference on Neural Networks*, vol. 2017-May, pp. 681–688, Jun. 2017, doi: 10.1109/IJCNN.2017.7965918.

[132] “Merge Layers - Keras Documentation.” https://keras.io/layers/merge/ (accessed Sep. 28, 2019).

[133] S. U. Yunas and K. B. Ozanyan, “Gait Activity Classification from Feature-Level Sensor Fusion of Multi-Modality Systems,” *IEEE Sens. J.*, vol. 21, no. 4, pp. 4801–4810, Feb. 2021, doi: 10.1109/JSEN.2020.3028697.

[134] S. U. Yunas and K. B. Ozanyan, “Gait Activity Classification using Multi-Modality Sensor Fusion: A Deep Learning Approach,” *IEEE Sens. J.*, pp. 1–1, May 2021, doi: 10.1109/jsen.2021.3077698.

[135] J. N. Tomasi, “Development and Evaluation of a Sensor System to Monitor the

Stance-Phase Control Function of the Automatic Stance-Phase Lock (ASPL) Mechanism," 2016. Accessed: Aug. 19, 2020. [Online]. Available: https://tspace.library.utoronto.ca/bitstream/1807/76236/1/Tomasi_Jessica_N_201611_MHSc_thesis.pdf.

[136] "Motion Capture." https://assistiverobotcenter.github.io/projects/2019/06/19/sensors-paper5 (accessed Jun. 19, 2021).

[137] Y. Wang, "UNIVERSITY OF CALIFORNIA Los Angeles Recognition and Classification of the Wolf Motor Function Test Items using Multimode Sensor Fusion," 2012, Accessed: Aug. 19, 2020. [Online]. Available: https://escholarship.org/uc/item/9n21974t.

[138] A. Gijsberts and B. Caputo, "Exploiting accelerometers to improve movement classification for prosthetics," *2013 IEEE 13th International Conference on Rehabilitation Robotics (ICORR)*, pp. 1-5, 2013, doi: 10.1109/ICORR.2013.6650476.

[139] C. Chen, R. Jafari, and N. Kehtarnavaz, "Improving Human Action Recognition Using Fusion of Depth Camera and Inertial Sensors," *IEEE Trans. Human-Machine Syst.*, vol. 45, no. 1, pp. 51–61, Feb. 2015, doi: 10.1109/THMS.2014.2362520.

[140] X. Yang and D. Sun, "Feature-level fusion of palmprint and palm vein base on canonical correlation analysis," in *International Conference on Signal Processing Proceedings, ICSP*, vol. 0, pp. 1353–1356, Jul. 2016, doi: 10.1109/ICSP.2016.7878047.

[141] X. Li, O. W. Samuel, X. Zhang, H. Wang, P. Fang, and G. Li, "A motion-classification strategy based on sEMG-EEG signal combination for upper-limb amputees," *J. Neuroeng. Rehabil.*, vol. 14, no. 1, pp. 1–13, Jan. 2017, doi: 10.1186/s12984-016-0212-z.

[142] L. Tao *et al.*, "Energy expenditure estimation using visual and inertial sensors," *IET Comput. Vis.*, vol. 12, no. 1, pp. 36–47, Feb. 2018, doi: 10.1049/iet-cvi.2017.0112.

[143] A. G. Leal-Junior *et al.*, "POF-IMU sensor system: A fusion between inertial measurement units and POF sensors for low-cost and highly reliable systems," *Opt. Fiber Technol.*, vol. 43, pp. 82–89, Jul. 2018, doi: 10.1016/J.YOFTE.2018.04.012.

[144] J. Beil, I. Ehrenberger, C. Scherer, C. Mandery, and T. Asfour, “Human Motion Classification Based on Multi-Modal Sensor Data for Lower Limb Exoskeletons,” in *IEEE International Conference on Intelligent Robots and Systems*, Dec. 2018, pp. 5431–5436, doi: 10.1109/IROS.2018.8594110.

[145] M. J. Rahman, E. Nemati, M. Rahman, K. Vatanparvar, V. Nathan, and J. Kuang, “Toward Early Severity Assessment of Obstructive Lung Disease Using Multi-Modal Wearable Sensor Data Fusion during Walking,” in *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, Jul. 2020, vol. 2020-July, pp. 5935–5938, doi: 10.1109/EMBC44109.2020.9176559.

Appendix A: Abbreviations

AIS	Ambulatory Inertial Sensors
CCA	Canonical Correlation Analysis
CNN	Convolutional Neural Network
CVP	Canonical Covariate Pair
DoF	Degrees of Freedom
DL	Deep Learning
DNN	Deep Neural Network
DT	Decision Tree
DWT	Discrete Wavelet Transform
EEG	Electroencephalogram
ECG	Electrocardiogram
EMG	Electromyography
FFT	Fast Fourier Transform
FFNN	Feed Forward Neural Networks
FS	Floor Sensors
GUI	Graphical User Interface
IMU	Inertial Measurement Unit
K-NN	Kernel Nearest Neighbour

K-SVM	Kernel Support Vector Machine
LDA	Linear Discriminant Analysis
LR	Linear Regression
LSTM	Long Short-Term Memory
NB	Naïve Bayes
PCA	Principal Component Analysis
POF	Plastic Optical Fiber
RF	Random Forest
RMS	Root Mean Square
RNN	Recurrent Neural Network
R-Pi	Raspberry-Pi
SOS	Sum of squares
SVM	Support Vector Machine

Appendix B: Codes

B.1 Data acquisition and pre-processing codes

B.1.1 Arduino code to acquire data from 9DoF IMU sensors

```
#include <SparkFunMPU9250-DMP.h>
#define SerialPort SerialUSB
MPU9250_DMP imu;

void setup()
{
SerialPort.begin(115200);

if (imu.begin() != INV_SUCCESS)
  {
    while (1)
    {
SerialPort.println("Unable to communicate with MPU-9250");
SerialPort.println("Check connections, and try again.");
SerialPort.println();
delay(5000);
    }
  }

imu.setSensors(INV_XYZ_GYRO | INV_XYZ_ACCEL | INV_XYZ_COMPASS);

// Gyro options are +/- 250, 500, 1000, or 2000 dps
imu.setGyroFSR(2000);
// Accel options are +/- 2, 4, 8, or 16 g
imu.setAccelFSR(2);

imu.setLPF(10); // Set LPF corner frequency to 5Hz
imu.setSampleRate(100); // Set sample rate to 10Hz (changed)
imu.setCompassSampleRate(20); // Set mag rate to 10Hz (changed)
}

void loop()
{
if ( imu.dataReady() )
  {
imu.update(UPDATE_ACCEL | UPDATE_GYRO | UPDATE_COMPASS);
    printIMUData();
  }
}

void printIMUData(void)
{
```

```
43      // convert the raw sensor readings (signed 16-bit values) to their respective units.
44      float accelX = imu.calcAccel(imu.ax);
45      float accelY = imu.calcAccel(imu.ay);
46      float accelZ = imu.calcAccel(imu.az);
47      float gyroX = imu.calcGyro(imu.gx);
48      float gyroY = imu.calcGyro(imu.gy);
49      float gyroZ = imu.calcGyro(imu.gz);
50      float magX = imu.calcMag(imu.mx);
51      float magY = imu.calcMag(imu.my);
52      float magZ = imu.calcMag(imu.mz);
53
54      SerialPort.println(String(accelX) + "," + String(accelY) + "," + String(accelZ) + "," +
55      String(gyroX) + "," + String(gyroY) + "," + String(gyroZ));
56      }
```

B.1.2 Python code to acquire data from FS

```
1       #!/usr/bin/python3
2       import struct
3       import sys
4       from collections import namedtuple
5       from math import sqrt
6       import socket
7
8       #format strings for unpacking data structures
9       headerrecordstruct = '<8sHBxI'
10      headerrecordlen = struct.calcsize(headerrecordstruct)
11      expectedmagic = b'pigmat\x00\xff'
12
13      ledrecordstruct = '<BBHIdd'
14      ledrecordlen = struct.calcsize(ledrecordstruct)
15      ledrecord = namedtuple('ledrecord',['led','channel','oppoint','reserved','slope','offset'])
16
17      adctotalchannels = 128
18      resultrecordstruct = '<HHIQ'
19      resultrecorditemcount = 4
20      resultblockstruct = '<QQQQ' + (resultrecordstruct[1:]) * adctotalchannels;
21      #print(resultblockstruct)
22      resultblocklen = struct.calcsize(resultblockstruct)
23      resultblockinitialitems = 4
24
25      clientheaderstruct = '<8sHHI'
26      clientheaderlen = struct.calcsize(clientheaderstruct);
27
28      address = sys.argv[1];
29
30      s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
31      s.connect((address, 12911))
32
33      magic = b'tamgip\x00\xff'
34      version = 0
35      reserved = 0
36      blocklength = 333;
37      clientheader = struct.pack(clientheaderstruct,magic,version,reserved,blocklength)
38
39      s.sendall(clientheader)
40
41      def receiveall(s,length):
42      buf = b";
43              remain = length;
```

```
while (remain > 0):
remain = length - len(buf)
buf += s.recv(remain)
        return buf

(magic, version, ledcount,blocklength) =
struct.unpack(headerrecordstruct,receiveall(s,headerrecordlen))
if (magic != expectedmagic):
print('bad magic')
sys.exit(1)

print('version: '+str(version))
print('ledcount: '+str(ledcount))
print('blocklength: ' + str(blocklength))
print('ledrecordlen: ' + str(ledrecordlen))

leds = []
ledbytes = receiveall(s,ledrecordlen * ledcount)
#print(repr(ledbytes))
#sys.exit(1)

for ledtuple in struct.iter_unpack(ledrecordstruct,ledbytes):
#print(repr(ledtuple))
leds.append(ledrecord(*ledtuple))

while 1:
resultbytes = receiveall(s,resultblocklen)
resulttuple = struct.unpack(resultblockstruct,resultbytes)
#print(repr(resulttuple))

        resultline = []
for datacol in range(0,ledcount):
lrn = datacol #add ply reordering stuff here
led = leds[lrn].led
channel = leds[lrn].channel
dataoffset = resultblockinitialitems + (channel * resultrecorditemcount)
(minr,maxr,sumr,sosr) = resulttuple[dataoffset:dataoffset+resultrecorditemcount]
meanr = sumr / blocklength
mosr = sosr / blocklength
sdr = sqrt(mosr - (meanr * meanr))
#print('led:'+str(led)+' min: '+str(minr)+' max:'+str(maxr)+' mean:'+str(meanr)+'
sd:'+str(sdr))
slope = leds[lrn].slope
offset = leds[lrn].offset
minp = minr * slope + offset;
maxp = maxr * slope + offset;
meanp = meanr * slope + offset;
sdp = sdr * slope;
#print('led:'+str(led)+' min: '+str(minp)+' max:'+str(maxp)+' mean:'+str(meanp)+'
sd:'+str(sdp))
resultline.append(meanp);
print(repr(resultline))
```

B.1.3 MATLAB code to implement image reconstruction for FS

```
alpha = 0.5;
iterations = 20;
count=0;
for i=1:iterations
   % More surrounding values than gk
```

```
    err4 = sn'*gk;      % 116 x 1 = 116 x 20000 * 20000 x 1
    % More central values and abnormal values (values with less neighbours) got hidden
    err3 = (cn-err4);   % 116 x 1 = 116 x 1 - 116 x 1
    err2 = sn*err3;     % 20000 x 1 = 20000 x 116 * 116 x 1
    err1 = alpha*err2;
    gk2 = gk + err1;
    gk2(gk2<0)=0;
    gk2(gk2>1)=1;
    gk = gk2;
end
```

B.1.4 MATLAB code to implement convolution operation

```
% Convoultion in spatial domain with 3x3 mask of ones
M=[ 1 1 1;
    1 1 1;
    1 1 1];
gk1=gk;
[r,c] = size(gk);
[m,n] = size(M);
h = rot90(M, 2);
center = floor((size(h)+1)/2);
left = center(2) - 1;
right = n - center(2);
top = center(1) - 1;
bottom = m - center(1);
Rep = zeros(r + top + bottom, c + left + right);
% Padding of zeros around the corners
for x = 1 + top : r + top
    for y = 1 + left : c + left
        Rep(x,y) = gk(x - top, y - left);
    end
end
B = zeros(r , c);
% Convolution Algorithm
for x = 1 : r
    for y = 1 : c
        count = 0;
        for i = 1 : m
            for j = 1 : n
                q = x - 1;
                w = y -1;
                B(x, y) = B(x, y) + (Rep(i + q, j + w) * h(i, j));
                if((Rep(i + q, j + w) * h(i, j))~=0)
                    count = count + 1;
                end
                gk1(x,y)=B(x,y);
            end
            if(count<=7)
                gk1(x,y)=0;
            end
        end
    end
end
```

B.1.5 Python code to acquire data from AIS

```
from sense_hat import SenseHat
import serial
```

```
3	import time
4	import csv
5	import os
6	sense = SenseHat()
7
8	import datetime
9	from datetime import datetime
10	import time
11	from time import sleep
12
13	monitoringTime=0
14	fusedData = []
15	count = 0
16	ser0 = serial.Serial('/dev/ttyACM0',115200, timeout = 0.01)
17	ser1 = serial.Serial('/dev/ttyACM1',115200, timeout = 0.01)
18	s1 = float(round(time.time()*1000))
19
20	while monitoringTime < 40:
21	print '%13f' %s1
22
23	acl = sense.get_accelerometer_raw()
24	gyr = sense.get_gyroscope_raw()
25
26	hax = round(acl['x'],2)
27	hay = round(acl['y'],2)
28	haz = round(acl['z'],2)
29	hgx = round(gyr['x'],2)
30	hgy = round(gyr['y'],2)
31	hgz = round(gyr['z'],2)
32
33	line_s0 = ser0.readline()[:-2]
34	line_s1 = ser1.readline()[:-2]
35	line_split0 = line_s0.split(',')
36	line_split1 = line_s1.split(',')
37
38	iax0 = line_split0[0]
39	iay0 = line_split0[1]
40	iaz0 = line_split0[2]
41	igx0 = line_split0[3]
42	igy0 = line_split0[4]
43	igz0 = line_split0[5]
44
45	iax1 = line_split1[0]
46	iay1 = line_split1[1]
47	iaz1 = line_split1[2]
48	igx1 = line_split1[3]
49	igy1 = line_split1[4]
50	igz1 = line_split1[5]
51
52	s2 = float(round(time.time()*1000))
53
54	while (s2 - s1) <= 51:
55	s2 = float(round(time.time()*1000))
56	count = count + 1
57	print '%13f' %s2
58	temp = '%13f' %(s2-s1)
59		print temp
60
61	s1 = s2
62	s3 = "{:13f}".format(s2)
63
```

```
fusedData.append([s3,hax,hay,haz,hgx,hgy,hgz,iax0,iay0,iaz0,igx0,igy0,igz0,iax1,iay1,iaz1,
igx1,igy1,igz1])
        print
monitoringTime = monitoringTime+1

print("Test run completed")

sense.clear()

def open_with_csv(filename, d = ','):
with open(filename, 'w') as csvin:
headernames  =  ['TIME','HAT-ACL/X','HAT-ACL/Y','HAT-ACL/Z','HAT-GYR/X','HAT-
GYR/Y','HAT-GYR/Z','IMU1-ACL/X','IMU1-ACL/Y','IMU1-ACL/Z','IMU1-
GYR/X','IMU1-GYR/Y','IMU1-GYR/Z','IMU2-ACL/X','IMU2-ACL/Y','IMU2-
ACL/Z','IMU2-GYR/X','IMU2-GYR/Y','IMU2-GYR/Z']
writer = csv.writer(csvin, delimiter=d)
writer.writerow(headernames)
for entry in fusedData:
writer.writerow(entry)

# Assign filename to csv file
open_with_csv('fused_wear_sensors.csv')
```

B.1.6 Python code to load and pre-process data from FS and AIS

```
import numpy as np
import pandas as pd
import glob
from pandas import read_csv
from matplotlib import pyplot
import os
os.environ["PATH"] += os.pathsep + 'C:\Program Files\Graphviz 2.44.1\bin'
from keras.utils import plot_model
import seaborn as sns # Statistical data visualization

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()

#------------------------------------------------ SMART CARPET
################################# Normal Walk
#################################
path =r'C:\Users\mchijsy3\Google
Drive\Computer\New_data\smrt_crpt\activities_smrt_crpt\1. Norm_Walk'
allFiles = glob.glob(path + "/*.csv")
list_ = []
for file_ in allFiles:
   df = pd.read_csv(file_,index_col=None, header=None, skiprows = 1,
usecols=range(1,117))
#    df = df.iloc[15:85,:]
   list_.append(df)
s_frame_norm = pd.concat(list_, axis = 0, ignore_index = True)
f1 = s_frame_norm
#f1 = sc.fit_transform(f1)
################################# Fast Walk
#################################
path =r'C:\Users\mchijsy3\Google
Drive\Computer\New_data\smrt_crpt\activities_smrt_crpt\2. Fast_Walk'
allFiles = glob.glob(path + "/*.csv")
list_ = []
```

```
for file_ in allFiles:
    df = pd.read_csv(file_,index_col=None, header=None, skiprows = 1,
usecols=range(1,117))
#    df = df.iloc[15:85,:]
    list_.append(df)
s_frame_norm = pd.concat(list_, axis = 0, ignore_index = True)
f2 = s_frame_norm
#f2 = sc.fit_transform(f2)
################################ Subtracting 3
################################
path =r'C:\Users\mchijsy3\Google
Drive\Computer\New_data\smrt_crpt\activities_smrt_crpt\3. Sub3_Walk'
allFiles = glob.glob(path + "/*.csv")
list_ = []
for file_ in allFiles:
    df = pd.read_csv(file_,index_col=None, header=None, skiprows = 1,
usecols=range(1,117))
#    df = df.iloc[15:85,:]
    list_.append(df)
s_frame_norm = pd.concat(list_, axis = 0, ignore_index = True)
f3 = s_frame_norm
#f3 = sc.fit_transform(f3)
################################ Subtracting 7
################################
path =r'C:\Users\mchijsy3\Google
Drive\Computer\New_data\smrt_crpt\activities_smrt_crpt\4. Sub7_Walk'
allFiles = glob.glob(path + "/*.csv")
list_ = []
for file_ in allFiles:
    df = pd.read_csv(file_,index_col=None, header=None, skiprows = 1,
usecols=range(1,117))
#    df = df.iloc[15:85,:]
    list_.append(df)
s_frame_norm = pd.concat(list_, axis = 0, ignore_index = True)
f4 = s_frame_norm
#f4 = sc.fit_transform(f4)
################################ Walking While Listening
######################
path =r'C:\Users\mchijsy3\Google
Drive\Computer\New_data\smrt_crpt\activities_smrt_crpt\5. List_Walk'
allFiles = glob.glob(path + "/*.csv")
list_ = []
for file_ in allFiles:
    df = pd.read_csv(file_,index_col=None, header=None, skiprows = 1,
usecols=range(1,117))
#    df = df.iloc[15:85,:]
    list_.append(df)
s_frame_norm = pd.concat(list_, axis = 0, ignore_index = True)
f5 = s_frame_norm
#f5 = sc.fit_transform(f5)
################################ Walking While Texting
#######################
path =r'C:\Users\mchijsy3\Google
Drive\Computer\New_data\smrt_crpt\activities_smrt_crpt\6. Tapp_Walk'
allFiles = glob.glob(path + "/*.csv")
list_ = []
for file_ in allFiles:
    df = pd.read_csv(file_,index_col=None, header=None, skiprows = 1,
usecols=range(1,117))
#    df = df.iloc[15:85,:]
    list_.append(df)
```

```
s_frame_norm = pd.concat(list_, axis = 0, ignore_index = True)
f6 = s_frame_norm
#f6 = sc.fit_transform(f6)
################################ Walking While Walking
#######################
path =r'C:\Users\mchijsy3\Google
Drive\Computer\New_data\smrt_crpt\activities_smrt_crpt\7. Talk_Walk'
allFiles = glob.glob(path + "/*.csv")
list_ = []
for file_ in allFiles:
    df = pd.read_csv(file_,index_col=None, header=None, skiprows = 1,
usecols=range(1,117))
#    df = df.iloc[15:85,:]
    list_.append(df)
s_frame_talk = pd.concat(list_, axis = 0, ignore_index = True)
f7 = s_frame_norm
#f7 = sc.fit_transform(f7)
##############################################################################
#####

s_X = np.row_stack((f1,f2,f3,f4,f5,f6,f7))
s_X = sc.fit_transform(s_X)

s_X = np.reshape(s_X, (770,120,116,1))

#------------------------------------------------ Wearable Sensors
################################ Normal Walk
##################################
path =r'C:\Users\mchijsy3\Google
Drive\Computer\New_data\ambl_sens\activities_ambl_sens\1. Norm_Walk'
allFiles = glob.glob(path + "/*.csv")
list_ = []
for file_ in allFiles:
    df = pd.read_csv(file_, header=None)
#    df = df.iloc[15:85,:]
    list_.append(df)
w_frame_norm = pd.concat(list_, axis = 0, ignore_index = True)
f1 = w_frame_norm
#f1 = sc.fit_transform(w_frame_norm)
################################ Fast Walk
##################################
path =r'C:\Users\mchijsy3\Google
Drive\Computer\New_data\ambl_sens\activities_ambl_sens\2. Fast_Walk'
allFiles = glob.glob(path + "/*.csv")
list_ = []
for file_ in allFiles:
    df = pd.read_csv(file_, header=None)
#    df = df.iloc[15:85,:]
    list_.append(df)
w_frame_fast = pd.concat(list_, axis = 0, ignore_index = True)
f2 = w_frame_fast
#f2 = sc.fit_transform(w_frame_fast)
################################ Subtracting 3
################################
path =r'C:\Users\mchijsy3\Google
Drive\Computer\New_data\ambl_sens\activities_ambl_sens\3. Sub3_Walk'
allFiles = glob.glob(path + "/*.csv")
list_ = []
for file_ in allFiles:
    df = pd.read_csv(file_, header=None)
#    df = df.iloc[15:85,:]
```

```
    list_.append(df)
w_frame_sub3 = pd.concat(list_, axis = 0, ignore_index = True)
f3 = w_frame_sub3
#f3 = sc.fit_transform(w_frame_sub3)
################################ Subtracting 7
################################
path =r'C:\Users\mchijsy3\Google
Drive\Computer\New_data\ambl_sens\activities_ambl_sens\4. Sub7_Walk'
allFiles = glob.glob(path + "/*.csv")
list_ = []
for file_ in allFiles:
    df = pd.read_csv(file_, header=None)
#    df = df.iloc[15:85,:]
    list_.append(df)
w_frame_sub7 = pd.concat(list_, axis = 0, ignore_index = True)
f4 = w_frame_sub7
#f4 = sc.fit_transform(w_frame_sub7)
################################ Walking While Listening
######################
path =r'C:\Users\mchijsy3\Google
Drive\Computer\New_data\ambl_sens\activities_ambl_sens\5. List_Walk'
allFiles = glob.glob(path + "/*.csv")
list_ = []
for file_ in allFiles:
    df = pd.read_csv(file_, header=None)
#    df = df.iloc[15:85,:]
    list_.append(df)
w_frame_list = pd.concat(list_, axis = 0, ignore_index = True)
f5 = w_frame_list
#f5 = sc.fit_transform(w_frame_list)
################################ Walking While Texting
#######################
path =r'C:\Users\mchijsy3\Google
Drive\Computer\New_data\ambl_sens\activities_ambl_sens\6. Tapp_Walk'
allFiles = glob.glob(path + "/*.csv")
list_ = []
for file_ in allFiles:
    df = pd.read_csv(file_, header=None)
#    df = df.iloc[15:85,:]
    list_.append(df)
w_frame_text = pd.concat(list_, axis = 0, ignore_index = True)
f6 = w_frame_text
#f6 = sc.fit_transform(w_frame_text)
################################ Walking While Talking
#######################
path =r'C:\Users\mchijsy3\Google
Drive\Computer\New_data\ambl_sens\activities_ambl_sens\7. Talk_Walk'
allFiles = glob.glob(path + "/*.csv")
list_ = []
for file_ in allFiles:
    df = pd.read_csv(file_, header=None)
#    df = df.iloc[15:85,:]
    list_.append(df)
w_frame_talk = pd.concat(list_, axis = 0, ignore_index = True)
f7 = w_frame_talk
#f7 = sc.fit_transform(w_frame_talk)
##########################################################################
#####

w_X = np.row_stack((f1,f2,f3,f4,f5,f6,f7))
w_X = sc.fit_transform(w_X)
```

```

w_X = np.reshape(w_X, (770,120,18,1))

from keras.utils import to_categorical

y1 = np.full((110, 1), 0)
y2 = np.full((110, 1), 1)
y3 = np.full((110, 1), 2)
y4 = np.full((110, 1), 3)
y5 = np.full((110, 1), 4)
y6 = np.full((110, 1), 5)
y7 = np.full((110, 1), 6)

# y1 = np.full((13200, 1), 0)
# y2 = np.full((13200, 1), 1)
# y3 = np.full((13200, 1), 2)
# y4 = np.full((13200, 1), 3)
# y5 = np.full((13200, 1), 4)
# y6 = np.full((13200, 1), 5)

y = np.row_stack((y1,y2,y3,y4,y5,y6,y7))
y = to_categorical(y)

# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
trainsX, testsX, trainY, testY = train_test_split(s_X,y, test_size=0.20, random_state=42)
        trainwX, testwX, trainY, testY = train_test_split(w_X,y, test_size=0.20,
        random_state=42)
```

B.2 Python codes for multi-modality sensor fusion

B.2.1 Implementation of PCA and CCA

```
### Applying PCA
from sklearn.decomposition import PCA
pca = PCA(n_components = 12)
X = np.column_stack((s_X,w_X))
X = pca.fit_transform(X)
explained_variance = pca.explained_variance_ratio_
summer = np.sum(explained_variance[0:18])

# Applying CCA
from sklearn.cross_decomposition import CCA
cca = CCA(n_components = 19)
cca.fit(s_X,w_X)
s_X,w_X = cca.transform(s_X,w_X)
X = np.column_stack((s_X,w_X))
```

B.2.2 Implementation of FFNN

```
from keras.layers import Dense
from keras.models import Model
from keras.layers import Input
from keras.layers.merge import concatenate
from keras.utils import plot_model
from keras.layers import BatchNormalization

```

```
def evaluate_model(trainsX, trainwX, trainY, testsX, testwX, testY):
    # first input model
    num_classes = 7
    visible1 = Input(shape=(116,))
    hs11 = Dense(64, activation = 'relu')(visible1)
    hs12 = Dense(32, activation = 'relu')(hs11)
    hs13 = Dense(10, activation = 'relu')(hs12)
    # bn1 = BatchNormalization()(hs13)
    # second input model
    visible2 = Input(shape=(18,))
    hs21 = Dense(16, activation = 'relu')(visible2)
    hs22 = Dense(12, activation = 'relu')(hs21)
    hs23 = Dense(10, activation = 'relu')(hs22)
    # bn2 = BatchNormalization()(hs23)
    # merge input models
    merge = concatenate([hs13, hs23])
    hs31 = Dense(8, activation = 'relu')(merge)
    # output
    output = Dense(num_classes, activation='softmax')(hs31)
    model = Model(inputs=[visible1, visible2], outputs=output)
    model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
        # fit network
    history = model.fit([trainsX, trainwX], trainY,validation_data=([testsX, testwX], testY),
epochs=50, batch_size=120)

    # scores = model.evaluate([trainsX, trainwX],trainY)
    # print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
    # print(model.summary())
    plot_model(model, to_file=r'C:\Users\mchijsy3\Google
Drive\Computer\3_Third_Year\Coding(3rd_Year)\plot_ANN_complete.jpg',
show_shapes=True, show_layer_names=True)

    y_pred = model.predict([testsX,testwX])
    return y_pred, history
```

B.2.3 Implementation of 1D-CNN

```
# Multiple Inputs
from keras.models import Model
from keras.layers import Input
from keras.layers import Dense
from keras.layers import Flatten
from keras.layers import BatchNormalization
from keras.layers.convolutional import Conv1D
from keras.layers.pooling import MaxPooling1D
from keras.layers.merge import concatenate

def evaluate_model(trainsX, trainwX, trainY, testsX, testwX, testY):
    # first input model
    visible1 = Input(shape=(116,1))
    conv11 = Conv1D(64, kernel_size = 3, activation = 'relu', padding = 'same' )(visible1)
    pool11 = MaxPooling1D(pool_size = 2)(conv11)
    conv12 = Conv1D(16, kernel_size = 3, activation = 'relu', padding = 'same')(pool11)
    pool12 = MaxPooling1D(pool_size = 2)(conv12)
    flat1 = Flatten()(pool12)
    hs1 = Dense(10, activation = 'relu')(flat1)

    # second input model
    visible2 = Input(shape=(18,1))
    conv21 = Conv1D(32, kernel_size = 3, activation = 'relu', padding = 'same')(visible2)
```

```
    pool21 = MaxPooling1D(pool_size = 2)(conv21)
    conv22 = Conv1D(16, kernel_size = 3, activation = 'relu', padding = 'same')(pool21)
    pool22 = MaxPooling1D(pool_size = 2)(conv22)
    flat2 = Flatten()(pool22)
    hs2 = Dense(10, activation = 'relu')(flat2)
    # merge input models
    merge = concatenate([hs1, hs2])
    hs3 = Dense(8, activation = 'relu')(merge)
    # output
    output = Dense(7, activation='softmax')(hs3)
    model = Model(inputs=[visible1, visible2], outputs=output)
    model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
        # fit network
    history = model.fit([trainsX, trainwX], trainY,validation_data=([testsX, testwX], testY),
epochs=50, batch_size=120)

    plot_model(model, to_file=r'C:\Users\mchijsy3\Google
Drive\Computer\3_Third_Year\Coding(3rd_Year)\plot_CNN1d_complete.jpg',
show_shapes=True, show_layer_names=True)

    y_pred = model.predict([testsX,testwX])
    return y_pred,history
```

B.2.4 Implementation of 2D-CNN

```
# Multiple Inputs
from keras.models import Model
from keras.models import Sequential
from keras.layers import Input
from keras.layers import Dense
from keras.layers import Flatten
from keras.layers.convolutional import Conv2D
from keras.layers.pooling import MaxPooling2D
from keras.layers.merge import concatenate

def evaluate_model(trainsX, trainwX, trainY, testsX, testwX, testY):
    # first input model
    visible1 = Input(shape=(120,116,1))
    conv11 = Conv2D(64, kernel_size = 3, activation = 'relu', padding = 'same')(visible1)
   # print(conv11)
    pool11 = MaxPooling2D(pool_size = 2)(conv11)
    conv12 = Conv2D(8, kernel_size = 3, activation = 'relu', padding = 'same')(pool11)
    pool12 = MaxPooling2D(pool_size = 2)(conv12)
    flat1 = Flatten()(pool12)
    # second input model
    visible2 = Input(shape=(120,18,1))
    conv21 = Conv2D(12, kernel_size = 3, activation = 'relu', padding = 'same')(visible2)
    pool21 = MaxPooling2D(pool_size = 2)(conv21)
    conv22 = Conv2D(8, kernel_size = 3, activation = 'relu', padding = 'same')(pool21)
    pool22 = MaxPooling2D(pool_size = 2)(conv22)
    flat2 = Flatten()(pool22)
    # merge input models
    merge = concatenate([flat1, flat2])
    hs3 = Dense(32, activation = 'relu')(merge)
    output = Dense(7, activation='softmax')(hs3)
    model = Model(inputs=[visible1, visible2], outputs=output)
    model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
        # fit network
    history = model.fit([trainsX, trainwX], trainY,validation_data=([testsX, testwX], testY),
epochs=50, batch_size=60)
```

```

    plot_model(model, to_file='plot_CNN2d_complete.jpg', show_shapes=True,
show_layer_names=True)

    y_pred = model.predict([testsX,testwX])
    return y_pred, history
```

B.2.5 Implementation of LSTM

```
from keras.models import Model
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import LSTM
from keras.layers import Input
from keras.layers import BatchNormalization
from keras.layers.merge import add

def evaluate_model(trainsX, trainwX, trainY, testsX, testwX, testY):
    # first input model
    visible1 = Input(shape=(120,116))
    LSTM11 = LSTM(16, return_sequences = True)(visible1)
    drop11 = Dropout(0.2)(LSTM11)
    LSTM12 = LSTM(16)(drop11)
    drop12 = Dropout(0.2)(LSTM12)
    hs1 = Dense(10, activation = 'relu')(drop12)
    # second input model
    visible2 = Input(shape=(120,18))
    LSTM21 = LSTM(16, return_sequences = True)(visible2)
    drop21 = Dropout(0.2)(LSTM21)
    LSTM22 = LSTM(16)(drop21)
    drop22 = Dropout(0.2)(LSTM22)
    hs2 = Dense(10, activation = 'relu')(drop22)
    # merge input models
    merge = add([hs1, hs2])
    hs3 = Dense(8, activation = 'relu')(merge)
    output = Dense(7, kernel_initializer = 'uniform', activation='softmax')(hs3)
    model = Model(inputs=[visible1, visible2], outputs=output)
    model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
    history = model.fit([trainsX, trainwX], trainY,validation_data=([testsX, testwX], testY),
epochs=1, batch_size=120)

    # scores = model.evaluate([trainsX, trainwX],trainY)[]
    # print("\n%s: %.2f%%" % (model.metrics_names[1], scores[1]*100))
    plot_model(model, to_file='plot_LSTM_Combined.jpg', show_shapes=True,
show_layer_names=True)

    y_pred = model.predict([testsX,testwX])
    return y_pred, history
```

B.2.6 Display for loss and accuracy

```
import time
import matplotlib.pyplot as plt

start = time.time()
y_pred, hist = evaluate_model(trainsX, trainwX, trainY, testsX, testwX, testY)
end = time.time()
hours, rem = divmod(end-start, 3600)
minutes, seconds = divmod(rem, 60)
```

```
epochs = range(1,len(hist.history['loss'])+1)
plt.figure()
plt.plot(epochs, hist.history['loss'], 'y', label='Training Loss')
plt.plot(epochs, hist.history['val_loss'], 'r', label='Validation Loss')
plt.legend()
plt.title('Training and validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.show()

plt.figure()
plt.plot(hist.history['acc'],label='train')
plt.plot(hist.history['val_acc'],label='test')
plt.legend()
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.show()
```