Technische Universiteit Delft

Human Activity Recognition using a Deep Learning Algorithm for Patient Monitoring

Esther Fridriksdottir

TU Delft

Human Activity Recognition using a Deep Learning Algorithm for Patient Monitoring

by

Esther Fridriksdottir

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday August 27th, 2019 at 10:00 AM.

Student number:	4746384	
Project duration:	December 17, 2018 – August 27th, 2019	
Thesis committee:	Prof. dr. P. J. French,	TU Delft (supervisor)
	Dr. ir. A. Bonomi,	Philips Research (supervisor)
	Dr. ir. M. A. P. Pertijs,	TU Delft
	Dr. B. Hunyadi,	TU Delft
	Dr. ir. J. F. L. Goosen,	TU Delft

This thesis is confidential and cannot be made public until August 27, 2024.

TU Delft

The work in this thesis was carried out in collaboration with Philips Research laboratories (Eindhoven, NL).
Their cooperation is hereby greatly acknowledged.

Abstract

Physical activity and mobility are important indicators of the recovery process of patients in the general ward of the hospital. Currently, monitoring mobility of hospitalized patients relies largely on direct observation from the caregivers. Accelerometers have the potential to quantify physical activity of patients objectively and without obstructing their daily routines. Human Activity Recognition (HAR) is a technique used to assess the type of activity an individual subject is carrying out based on sensor readings and has been extensively studied. However, the literature shows that HAR methodologies have been largely developed to recognize activities typical for healthy subjects. This means that activities performed at a slow and irregular pace, such as by a symptomatic patient or an elder, are scarcely considered to design HAR methods. Using HAR for patient monitoring would allow clinically meaningful metrics, such as time spent ambulating or in sedentary behaviour each day, to be obtained automatically. This may offer a convenient solution to enable caregivers automatically monitoring the recovery process of patients.

The aim of this work was to develop an accurate HAR model to recognize activities typical for hospitalized patients. A data collection study was conducted with 20 healthy volunteers in a simulated hospital environment. A single triaxial accelerometer placed on the trunk was used to measure body movement and recognize seven activity types: Lying in bed, upright posture, walking, walking with walking aid, wheelchair transport, stair ascent and stair descent. Features from both time and frequency domain were extracted and used to train three machine learning (ML) classifiers (Naïve Bayes, Random Forest, Support Vector Machine). Additionally, a deep neural network (DNN) consisting of a three convolutional layers and a Long Short-Term Memory layer was developed.

The performance of the DNN model was evaluated on holdout data and compared to the performance of the feature-based ML classifiers. The DNN model reached a higher classification accuracy than the latter approaches (F1-score= 0.902 vs. 0.821). All the models showed a large number of misclassification between the walking with or without walking aid class. By combining these two classes the DNN model reached an F1-score 0.946, compared to F1-score 0.856 of the best feature-based ML approach represented by a support vector machine classifier.

This work shows for the first time the value of applying deep-learning techniques to improve the accuracy of feature-based ML classifiers for addressing the problem of HAR using a single tri-axial accelerometer in simulated hospital conditions.

Acknowledgements

First of all I would like to thank my supervisor Alberto Bonomi, without him I would not have had the opportunity to take on this project. I highly value the experience I've gained these last several months at Philips. Together with Lieke and Laura we completed the data collection study and thanks to your flexibility and positivity I never had any problems with planning the sessions. Thus, I wanted to thank you and the rest of the team for the support. It been a pleasure working with all of you.

Secondly, I would like to thank my supervisor, Paddy French, for providing me guidance throughout the graduation process. In addition, I am thankful for the valuable input at the beginning of my data analysis from Borbala Hunyadi, who together with Michiel Pertijs and Hans Goosen, became members of my graduation committee, therefore, I would like to thank them all.

I'm grateful for all the friends I've made both in Delft and in Eindhoven. Even though these last two years were tough at times, I have so many good memories to look back on thanks to my Delft family and fellow Philips interns. Sigga Bogga, I am so happy that we came to Delft together. We've made such a great team and I don't want to imagine what these part two years would've been like if you weren't here. Þúsund þakkir! Alberto, Pedro, Jens, Armin, Quinten, Ferry and Irene, if it weren't for our vigorous study sessions in the first year I don't think I would've even made it through the first quarter. Amal, thank you for always being ready to help me when I need during the second year. Maarten, thank you for being there for me every step of the way during the thesis.

To my parents and sisters, living away from home these past two years hasn't always been easy but knowing that I have your support no matter what means the world to me. For that I am forever grateful.

Contents

List of Figures

List of Tables

1 Introduction

Physical activity and mobility are important indicators of the recovery process of patients in the general ward of the hospital. A decline in physical activity of patients can possibly indicate adverse events [1] whereas stable or improved activity levels can serve as a valuable input for assessing patient discharge readiness [2]. Discharge readiness refers to whether a patient is fit enough to return home or to a lower level of care. Mobility has been identified as a meaningful criterion when assessing patient discharge readiness, among other factors such as being clinically well and able to perform necessary activities of daily living (ADLs) [3]. Wearable accelerometers have the potential to act as powerful tools in evaluating the health status of patients during recovery in an objective way and enabling evaluation of rehabilitation and other medical interventions [4]. Time spent in sedentary behaviour is an example of an activity which can be measured with accelerometers and has been shown to have a relationship with performance of ADLs [5]. By incorporating posture detection algorithms, changes in posture can be detected which is important information for preventing pressure ulcer formation [6]. Several studies have reported a relationship between amount of physical activity measured by wearable sensors and length of hospital stay of post-operative patients. These measures were estimation of energy expenditure [7], amount of time spent in an upright position [8] or daily step count, either recorded in early post-operative period [9, 10] or in the last 24 hours before discharge [2]. A study by Sallis et al. [2] which continuously monitored over 700 patients of three medical-surgical units found that all patients, regardless of their length of stay (LOS), achieved at least 1100 steps in their last 24 hours before discharge. A similar finding was made by Cook et al. [10] who reported a significant relationship between step count in early post-operative period and LOS. These studies suggest that step count is one measure of mobility that could serve as an indicator for patient discharge readiness.

The clinical staff is expected to discharge patients efficiently such that readmission rates are minimized and patients don't remain hospitalised longer than necessary. As mobility is an important indicator of health status [11], it is relevant for the medical staff to know about the physical activities undertaken by their patients and the changes in their behaviour during the recovery period. Currently, monitoring mobility of hospitalized patients relies largely on direct observation from the caregivers. There are multiple tools available to assess the mobility and functional ability of patients. The choice of which assessment tool to use depends on feasibility and the clinician's preference. These tools are mainly divided in two categories; self-report and performance based measures [12]. Self-report involves asking the patient a series of questions regarding their functional abilities while performance based measures involve a patient performing certain tasks and a clinician evaluating the outcome. Self-report questionnaires are easy to use, less time consuming than performance based measures and are therefore often preferred over performance based measures [12]. However, self-report is based on the patient's perception of their mobility rather than actual performance which can lead to misleading results due to recall bias and under-reporting [13]. On the other hand, performance based measures provide objective evidence about the capabilities of the patient. The downside of using performance based measures is that setting up a test course can be time consuming or cumbersome for the clinician.

Assessment of patient mobility and performance of ADLs can be achieved by using human activity recognition (HAR). Wearable sensors such as accelerometers, gyroscopes, magnetometers and barometric pres-

sure sensors (BPS) can be used to provide data for HAR. Accelerometers measure linear acceleration and have been found to provide informative and accurate data for HAR [14]. Gyroscopes measure angular acceleration and are often used in combination with accelerometers for HAR. Gyroscopes are substantially more energy demanding than accelerometers and might not always provide useful information in addition to the acceleration data [15]. Magnetometers measure the strength of earth's magnetic field and have been found to provide informative data for HAR as well [16, 17]. A disadvantage of magnetometers is that they are not always a reliable source due to the fact that any nearby ferromagnetic material will distort the sensor output. BPS can be used to enhance detection of sit-to-stand and stand-to-sit transitions [18] and recognize stair walking [15]. As an alternative to wearable sensors, camera-based systems can be used for HAR. However, these systems have a limited working range and entail privacy issues. Processing signals from wearable sensors, such as accelerometers requires considerably less computational power compared to the camera-based approach and imposes a less invasion of privacy.

The HAR process is summed up in Figure 1.1. Firstly, acceleration data along the x-, y- and z- axes is collected from a device. The data is often preprocessed, which can include removing outliers, filling gaps in measurements, filtering and normalization. Next, the signals are segmented and labelled to allow feature extraction. These features can be statistical features such as mean and standard deviation, or frequency domain features such as Fourier coefficients. Lastly, the extracted features are used to develop a classification algorithm which can predict labels based on the features of a segment.

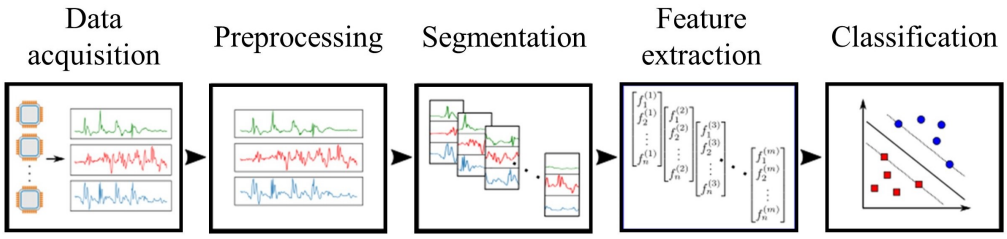


Figure 1.1: The Human Activity Recognition process from data acquisition to classification.

The classification part is commonly achieved using supervised machine learning (ML) algorithms. ML is a branch of artificial intelligence based on the idea that systems can identify patterns in data and learn from experience. Supervised ML refers to a group of algorithms which learn to associate labels with input data. Recently, an alternative approach called deep learning (DL) has become popular for HAR. DL is a subset of ML and is highly successful in areas such as object recognition and speech recognition. The main advantage of using DL approaches is that the feature extraction step is achieved automatically, as opposed to using hand-crafted features with supervised ML. HAR using accelerometers embedded in smartwatches and smartphones as fitness trackers has recently become widely accepted in the consumer industry. However, step detection shows high error rates during slow walking [19] and when using walking aids [20], which remains as one of the challenges for applying this technology in clinical settings, such as for patient monitoring.

The purpose of this thesis is to develop a classification model which can recognize typical activities of patients during hospitalization using a single accelerometer worn on the trunk. In this thesis, two different approaches will be explored and compared; feature-based ML and a DL approach. The aim is to answer the following research question: "How accurately can a deep learning algorithm recognize low-intensity activities, as performed by symptomatic patients, using a single trunk-worn accelerometer?"

Chapter 2 presents a literature review in accelerometer based HAR approaches, after which the objectives of the thesis are further elaborated. Chapter 3 provides information about the data acquisition process and data preprocessing. Chapter 4 presents the performance of three supervised ML algorithms (Naïve Bayes, Random Forest, Support Vector Machine) on the dataset to use as benchmark for the DL approach. Chapter 5 explains the model architecture of the Deep Neural Network (DNN) classifier. Chapter 6 contains the classification results and evaluation of the HAR models. Lastly, Chapter 7 includes conclusions and future recommendations.

2

Related Work

Micro electromechanical systems (MEMS) accelerometers are widely used in smartphones and other devices. These structures are essentially made up of a mass suspended by cantilevers. When the MEMS is subjected to acceleration, the mass exerts a force on the cantilevers, causing them to bend. Piezoresistive MEMS accelerometers measure the stress caused by the deflection of the cantilevers using piezoresistors. Alternatively, capacitive MEMS accelerometers measure change in capacitance as the mass moves away from or towards a reference plate (see Figure 2.1).

An extensive literature review was conducted to identify different HAR approaches using wearable accelerometers. The first section covers studies that compare performance of HAR based on sensor placement. In the second section, a variety of feature-based HAR approaches are discussed. Feature-based HAR approaches require domain knowledge, meaning that the researcher needs to know what information from the data is relevant for the model and what information is irrelevant. The following section covers DL approaches, which on the other hand do not require domain knowledge. Raw data can be fed directly into the DL model and it automatically learns a relationship between the raw data and its output label. Deep learning has only recently been applied to accelerometer based HAR problems, with promising results. The fourth and last section covers studies that have developed HAR models specifically for patient monitoring.

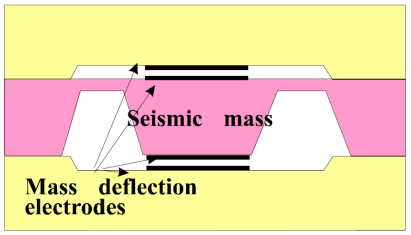


Figure 2.1: A capacitive micro electromechanical systems accelerometer. When the system is subjected to acceleration, the seismic mass moves, causing the cantilevers to bend. As the mass moves with respect to a reference electrode, a change in capacitance arises.

2.1. Optimal Sensor Placement for HAR

Acceleration signals differ significantly depending on the sensor placement on the body [21]. The optimal sensor placement for HAR is still a matter of debate and depends on the activities to be recognized. Several studies have investigated the optimal sensor placement for HAR [15, 17, 21–25]. Moncada-Torres et al. [15] found that using sensors placed on the wrists performed better than a combination of five sensors placed on chest, both wrists and ankles. The activities included in this study were 16 ADLs including whole body activities and also finer grain activities such as brushing teeth, cutting food and writing. Khatun et al. [22] found that the chest location provided the best specificity for 12 whole-body movements and 12 transitions compared to sensors located on the ankle and lower arm. Gao et al. [23] found that the waist sensor outperformed sensors placed on lower-arm, chest or thigh in detecting four whole-body activities and four postural transitions. Meanwhile, sensors placed on the ankle provide good information on ambulation, both for slow walking [10] and for sports activities [17].

Combining data from accelerometers placed on multiple parts of the body can provide more informative

data, however it has been shown that in some cases adding more sensors does not improve classification accuracy [15, 25]. Additionally, wearing multiple sensors can be cumbersome for the wearer and obstructive for daily activities. Thus, the amount of sensors to be worn on the body should be kept to a minimum. Awais et al. [24] compared classification accuracy of sensors placed on chest, wrist, lower back and thigh. The authors found that the best results for a single sensor classification were obtained using the sensor at the lower back, achieving an F1-score 0.81. The sensors at the chest and thigh performed slightly worse. Combining lower-back and thigh resulted in improving classification performance to 0.87. Cleland et al. [21] compared performance of accelerometers placed on the chest, wrist, lower back, hip, thigh and foot. The authors found that the sensor placed on the side of the hip reached the highest classification accuracy, 97.81%, followed by the chest sensor reaching 96.91% accuracy. A combination of chest and thigh sensor performed best out of two-sensor combinations with a classification accuracy 97.84%, only slightly higher than for a single hip sensor. Using a combination of three or more sensors did not improve the classification rate significantly. Ching et al. [25] demonstrated that using more than two sensors does not necessarily mean improving the classification rate significantly. Sensors placed on the wrist and thigh obtained a classification rate of 99.48% in recognizing nine activities, compared to the classification rate of 99.79% obtained using additional sensors on the arm, chest, waist and thigh.

2.2. Feature-based HAR Approaches

A feature-based HAR model can be rule-based, such that certain features are compared to empirically determined threshold values in order to differentiate between activities such as lying, sitting, standing and walking. Other approaches consist of ML classifiers such as decision trees, RF, SVM and so on. Explanations and derivations of these machine learning algorithms can be found in [26]. Other options for feature-based HAR approaches are hierarchical models which are a combination of multiple classification models.

2.2.1. Rule-based HAR Models

Rule-based detection algorithms for HAR are based on comparing the recorded sensor data to a set of empirically determined threshold values in order to differentiate between activities such as lying, sitting, standing and walking. Posture can be detected by applying a low-pass filter to the acceleration data to identify the gravitational component. If the orientation of the sensor relative to the body is known, the posture of the wearer can be estimated according to the gravitational vector. Predicting whether the wearer is lying down or in an upright position can be achieved by selecting threshold values for tilt angles of the trunk [4]. However, distinguishing between sitting and standing using only an accelerometer placed on the trunk remains a challenge since there is not a significant difference in trunk tilt angle for those two activities [27]. In order to distinguish between sitting and standing, several studies used a sensor placed on the thigh in combination with a chest-worn sensor [4, 28–30]. Others focused on accurately detecting Sit-to-Stand (SiSt) and Stand-to-Sit (StSi) transitions to classify sitting and standing by using Discrete Wavelet Transform (DWT) [31], by estimating vertical velocity [32] or by monitoring the change in trunk tilt angle [33].

To distinguish dynamic activities from static, a feature must be selected to represent activity intensity and a threshold value is used to determine whether an activity is static or dynamic. These features can be based on the frequency of the signal [27, 28, 34] or based on statistical features such as the mean absolute deviation [29], standard deviation [4] or normalized signal magnitude area (SMA) [30, 35]. The threshold value is either chosen based on prior experiments or derived from observing the data. Other features are then chosen to distinguish between different types of dynamic activities. Kim et al. [27] compared the magnitude of the acceleration vector to a threshold to distinguish between walking, running and fast running. Nazarahari et al. [33] used the successive positive peaks in acceleration magnitude to count steps during walking. The step count reached an accuracy of 99% during slow and normal walking and 95% for fast pace walking. Walking up and down stairs was detected using a threshold for the trunk inclination angle and steps were counted using the same method as for non-level walking, reaching accuracies of 97% and 96%, respectively. Liu et al [34] computed the energy of the acceleration signal derived from the Fourier transform of the signal magnitude. This feature was used to classify three activities; squatting down, sitting down and coughing.

HAR using empirically determined rule-based approaches are more transparent and intuitive than super-

vised ML and DL approaches. The downside of using rule-based detection algorithms is that a considerable amount of time goes into choosing meaningful features from the data and determining the set of thresholds for all features to characterize each activity. Thresholds for features are commonly determined based on observation of the recorded data, which is the same data as is used for evaluating the algorithm [4, 28, 30, 35]. This raises doubts about the generalizability of the algorithms. To ensure that algorithms are generalizable, studies should perform cross-validation and/or test their algorithm on an independent dataset.

2.2.2. Supervised ML HAR Models

A different way of approaching classification problems is by using supervised ML algorithms. For these approaches there is no need to empirically determine thresholds for each feature because the classification algorithm takes care of that. However, for most classifiers there are certain parameters that require tuning to obtain optimal results. As an example, for the k-nearest neighbour (k-NN) classifier, the value of k can influence the classification accuracy and should therefore be chosen carefully. These approaches are often able to make use of more features compared to the rule-based detection models. However, that may come at the cost of higher computational complexity. Commonly, vast amounts of features are extracted from the acceleration data and then feature reduction or feature selection algorithms are used to reduce the redundancy in information. Principal Component analysis (PCA) is a feature reduction method which is commonly used to extract linearly uncorrelated features (principal components) from data to provide meaningful information with less dimensionality. Alternatively, there are various feature selection algorithms available which select the optimal subset of features to reduce dimensionality of the classification problem. Making use of feature selection algorithms can not only improve classification performance but also reduce the computational cost of the system, which also means lower power consumption [24].

Altun et al. [36] demonstrated the high classification accuracy and low computational complexity of using a Bayesian Decision Making (BDM) classifier for HAR. A total of 1170 features were extracted per segment from five sensors. PCA was used to remove redundant information and reduced the amount of features to 30 per window. Using 10-fold cross validation the classification accuracy was found to be 99.2% using all five sensors while using only the sensor on the chest results in 96.6% accuracy. Others demonstrated that decision trees provide accurate classification while maintaining a low computational cost [23, 37]. Bagging is a technique which can be used to increase accuracy of decision trees by combining classifications of randomly generated training sets [38]. Khatun et al. [22] used this technique for accurately classifying 12 activities and 12 transitions. Arif et al. [39] used a correlation-based feature selection method and a rotation forest classifier [40] to classify 12 different activities using data from accelerometers on the chest, wrist and ankle. The classifier achieved a classification accuracy of 98% using all three accelerometers, while using only the chest sensor resulted in a classification rate of 93.8%. Reiss et al. [41] presented a modular activity monitoring system using a boosted decision trees. A triaxial chest-worn accelerometer made up the base module and is capable of coarsely estimating the intensity of an activity as light, moderate or vigorous. By adding a heart rate monitor this estimation became more accurate. Two more accelerometers, placed on the arm and foot, could be added to provide recognition of basic activities and postures. The average accuracy of the classification algorithm was 90.65%.

Other studies demonstrated the effectiveness of using support vector machines (SVM) [21, 24, 42] and k-NN [15, 43]. Parera et al. [42] used feature selection based on discriminant power and SVM to classify five different activities, including standing, walking, SiSt, StSi and other transitions. Using acceleration data from a single chest worn sensor the classifier reached an overall accuracy of 91% and could be used for real time activity monitoring. Awais et al. [24] used correlation-based feature selection for the SVM classifier for HAR in a free-living environment. Using a single chest-worn sensor the classifier reached an F1-score 0.784. Cleland et al. [21] reached a 96.91% accuracy using a single chest-worn sensor in classifying seven whole body activities and postures. Attal et al. [43] used Random Forest (RF) as a feature selection algorithm to select the 12 most informative features out of 168 features. Then a k-NN classifier was used to predict the type of activity. Moncada-Torres et al. [15] extracted a total of 131 features from five sensors, each containing a triaxial accelerometer, triaxial gyroscope and a barometric pressure sensor (BPS). The Relief-F [44] algorithm was used to select the most relevant features and a k-NN classifier with k=1 was chosen for classification, resulting in an overall classification accuracy of 92.5% using all five sensors. The authors showed that using a barometric pressure sensor (BPS) can improve classification for stair walking and level walking. In addition, the authors

found that features from the gyroscopes were not selected in any cases and therefore deemed the gyroscope unnecessary.

Sharma et al. [45] extracted a total of 388 features from a chest-worn accelerometer and used PCA to transform the feature set and reduce dimensionality. Using 28 principal components and an artificial neural network classifier a classification accuracy of 94% was obtained for classifying resting, walking and running activities and detecting simple forward falls. In another study, Sharma et al. [46] designed a back propagation neural network classifier to classify resting, walking and running using a single chest-worn accelerometer. The only features used for the classification were the Fast Fourier transform coefficients of the recorded signal. The classifier reached a mean classification rate of 83.96% for the testing dataset.

2.2.3. Hidden Markov Models and Hierarchical HAR Models

Some studies have implemented Hidden Markov Models (HMM) for the classification of activities. The benefit of this method is that it takes into account the previously detected activity and uses probabilistic models to assist in classifying the current activity. This way the algorithm knows that it is impossible to go directly from lying to walking, as an example. For this reason many consider HMM a promising tool for human activity recognition because it takes into account kinematic constraints in predicting sequence of activities. Li et al. [47] reported a daily physical activity classification system using a single chest-worn triaxial accelerometer with outstanding performance. HMM was used in combination with a modified Viterbi algorithm which has previously been used for speech recognition purposes. The algorithm was used to find the most likely state sequence given the observed sequence. The overall classification accuracy of five activities and four transitions was 99.59%. The classification of transitions achieved 92.63% while the activity recognition was near-perfect (>99% for each activity).

Kozina et al. [48] presented a three-layered activity recognition model using data from accelerometers worn on the chest, thigh and ankle. The authors showed that their model performed significantly better than three standard single-layer classifiers (SVM, decision tree and Naïve Bayes). The bottom layer of the HAR model consisted of three parts. Rule-based detection was used for classifying postures based on sensor orientation and RF classifiers returned probabilities on which type of activity the window belonged to. In addition, k-NN was used to reduce influence of badly annotated activities. The middle layer consisted of three hierarchical layers. The bottom layer used a RF classifier to recognize cycling or transitions such as SiSt and StSi. If the bottom layer did not recognize cycling or transitions, the classification continued to the middle layer. The middle layer used rule-based detection to distinguish between six different postures; lying, kneeling, sitting, bending, on all fours and upright. If an upright posture was detected the top layer further classified which upright posture such as standing, walking and running using an RF classifier. Finally, the top layer of the HAR model used HMM for making a final decision of classified activity by taking into account previous activities. By doing so, the model was able to make corrections in case the output of the middle layer did not fit into the previous sequence of activities. The proposed HAR model reached 98.03% accuracy using all three sensors and 79.43% accuracy using only the chest sensor.

Khan et at. [49] developed a hierarchical classification model and showed that a hierarchical approach performs better than single-level models in recognizing 15 activities using a single chest-worn sensor. The lower level of the hierarchical approach used statistical features such as mean, standard deviation, spectral entropy, and correlation to distinguish between three activity states; static, transitions and dynamic. The upper level made a prediction about the activity based on the state. The classification rate was 97.9% and 71.6% for the hierarchical and single layer models, respectively. That same group also presented a HAR system consisting of a single triaxial accelerometer that can be placed in any pocket and still obtain accurate classification results [50]. In order to do this, the authors proposed a two-level approach. The first level differentiated between a resting activity or dynamic activity coming from either the upper or lower body. The second level recognized the dynamic activities depending on where the data is assumed to be coming from. This HAR system allows for more flexibility than other proposed systems that assume a fixed position on the body and is therefore more suitable for long term monitoring.

2.3. Deep learning HAR Approaches

The deep learning approaches discussed in this section are Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks. These models are types of deep neural networks (DNNs) and are able to automatically extract meaningful features from raw data. Therefore, DNNs differ from the previously mentioned approaches in that they do not require domain knowledge and feature engineering.

2.3.1. Convolutional Neural Networks (CNNs) for HAR

CNNs have become extremely effective in recognizing objects in images and recently CNNs have been applied to sensor data for HAR with outstanding performances. Hur et al. [51] proposed an approach to translate inertial sensor recordings into an image and treat the activity recognition as an image processing problem. Images are constructed from 3 second long acceleration signals and fed into a CNN model with six 2D convolutional layers. Ha et al. [52] argues that 2D convolution is a better option to extract distinguishable features from multiple sensors.

Zhou et al. [53] developed a five layer CNN model to recognize activities such as standing still, walking, stairs walking, using an elevator or escalator, for indoor localization. The sensor data came from a smartphone and consisted of an accelerometer, gyroscope, magnetometer and barometer. In an effort to make the optimization easier and accelerate convergence, the sensor data is normalized to have zero mean and unit standard deviation. The segments are presented to the model as a vector and therefore 1D convolution is used. The model reached 98% accuracy in detecting nine activities. Ignatov et al. [54] proposed using a 1D convolutional layer followed by a max-pooling layer. The features remaining after the max-pooling layer are flattened and several statistical features are added to the feature vector, such as mean, variance and sum of absolute values. The segmented data is fed into the model without any normalization, as the authors stated that performing time series centering decreased accuracy for their model. The model was used to recognize six activities with an overall accuracy of 82.76%. The six activities were walking, upstairs, downstairs, sitting and standing. Due to the shallow structure of the recognition model the algorithm is suitable for running on mobile devices in real-time.

Cho et al. [55] developed a two-stage CNN HAR model. The data was sharpened to exaggerate finer details in the acceleration signals. This was done by applying a Gaussian filter to the data to attenuate high frequencies. This denoised signal is then subtracted from the original signal to obtain the detailed information stored in the higher frequencies. That signal is scaled and added to the original signal for a sharper signal. The first stage of the HAR model is a binary classification model which predicts whether the sharpened segments represent static or dynamic activities. The second stage consists of two identical CNNs composed of a convolutional layer, max-pooling layer and another convolutional layer. One of the second stage CNNs is trained to distinguish between sitting, standing and lying, while the other one is trained to distinguish between walking, walking upstairs and downstairs.

Avilez-Cruz et al. [56] developed a three-headed CNN model for recognizing six activities; sitting, standing, lying, walking, walking upstairs and downstairs. The three CNNs work in parallel, all receiving the same input signal coming from a triaxial accelerometer and a triaxial gyroscope (smartphone data-worn in pocket). One CNN extracts fine detailed information using four convolutional layers with max-pooling layers in between. The second CNN extracts less detailed features using two convolutional layers, each followed by max-pooling layers. The third CNN extracts coarse information from the segments using only one convolutional layer followed by a max-pooling layer with a pool size much larger than the other two CNNs (16 compared to 2 and 4). The feature maps of the three CNNs are flattened and concatenated before they are passed into a fully connected layer and at last an output layer with a softmax activation. This model reached an average of 100% accuracy for all six classes.

2.3.2. Long Short-Term Memory (LSTM) Networks for HAR

Long short-term memory (LSTM) networks are capable of writing, storing, and updating information in memory cells to learn from past data. These networks perform well in analysing time series data when there are long time lags between events of interest. Chen et al. [57] developed a HAR model with two LSTM layers

as hidden layers. The model reached 92.1% accuracy in recognizing six activities. Yu et al. proposed using a bidirectional LSTM. The benefit of using a bidirectional LSTM network over using the standard LSTM is that the current output of LSTM layers depends on previous information whereas by using bidirectional LSTM layers the output depends on both previous information and subsequent information. The bidirectional LSTM layers had 28 memory cells each and were followed by a fully connected layer. The model reached an accuracy of 93.79% in recognizing six activities, compared to 90.77% accuracy by the standard LSTM model.

Others experimented with combining LSTM layers and CNNs. Ordonez et al. [58] developed a hybrid model composed of five convolutional layers followed by two LSTM layers and compared to a baseline CNN model with the same structure, only the LSTM layers were replaced with fully connected layers. Overall, they found that the hybrid model reached higher F1-score. Li et al. [59] compared the performances of a CNN model, an LSTM model and a hybrid CNN+LSTM model. The hybrid model performed considerably better than the CNN and LSTM models.

2.4. HAR models for Patient Monitoring

The majority of studies which aim to develop HAR models are performed using healthy subjects and classifying high level activities such as walking, jogging and running which is not relevant for patient monitoring. For symptomatic hospitalized patients, the classification of transitions between different postures and time spent in sedentary behaviour can be an indicator of the health status. The main challenge of HAR for these subjects is the low intensity of activities performed. Aminian et al. [29] presented a rule-based HAR system comprising two single axis accelerometers worn on the chest and thigh. The algorithm classified dynamic activities from static postures (lying, sitting and standing) by comparing the mean absolute deviation of each 1 second window to a threshold value. The median value of the two sensors were then used to classify between sitting, standing or lying. The classification rate of the HAR system was 89.3%. The system was tested on three hospitalized patients and compared to patient self-report. There was a significant discrepancy between the sensor outcome and patient self-report, which was explained by subjective bias of patients. The authors concluded that the thresholds used for classification need to be adapted to each patient.

A study by Culhane et al. [4] assessed the accuracy of using accelerometers placed on trunk and thigh to continuously monitor mobility of hospitalized patients. The activities of five patients from the rehabilitation unit of a hospital with varying degrees of mobility were monitored over the course of four days. The standard deviation of acceleration data was compared to a threshold value in order to distinguish between static and dynamic state. This threshold was chosen based on visual inspection of the recorded data. For static activities the mean acceleration of a segment was used to determine inclination of trunk and thigh to classify postures. A midpoint threshold value of +/- 45° for both trunk and thigh inclination was compared against threshold values determined based on observation. Using the threshold values determined from observation, the algorithm reached a classification accuracy of 92% and higher. The mid-point approach performed worse for detecting sitting and lying with mean accuracies of 73% and 77%, respectively. The authors concluded that their method was practical for monitoring sedentary older adults in a clinical setting.

Rauen et al. [30] used normalized SMA for their rule-based HAR algorithm to monitor position changes of 30 immobile patients in early neurorehabilitation using triaxial accelerometers worn on the chest and thigh. Normalized SMA was summed up for each 1 second window and compared to a threshold of 0.2 g ($1g = 9.8m/s^2$) to distinguish rest from activities. Activities were classified as postural transitions if a tilt angle of 45° was exceeded and lasted longer than 15 seconds. The thresholds had been determined from the author's pilot study. The chest-worn accelerometer performed considerably better than the thigh sensor and was able to detect all position changes of the patients, and a few in addition to what was recorded in the standard written care documentation over 24 hours. The authors concluded that their approach was a promising method of monitoring position changes of immobile patients and evaluating their overall health.

Masse et al. [18] used a barometric pressure sensor (BPS) in combination with an accelerometer and a gyroscope to detect SiSt and StSt transitions of 12 mobility impaired stroke patients. The candidate transitions were detected based on tilt angle of the trunk obtained from filtered gyroscope data using DWT. Over 20 features were extracted from the accelerometer, gyroscope and BPS for every candidate transition. In order to

extract features from the BPS signal it was first converted to altitude and the pattern of transition enhanced using a sinusoidal fitting model to compensate for the low signal-to-noise ratio. Using a logistic regression-based classifier, the algorithm was able to correctly classify 90.6% of transitions compared to 75.4% when the barometer was excluded.

Nguyen et al. [60] studied classification of positions and postural transitions that are relevant to hospitalized patients. Accelerometers, gyroscopes and magnetometers were placed on the chest and abdomen of healthy subjects while they performed a set of activities including four different lying postures and three types of sitting postures based on information from medical textbooks. Transitions such as lying-to-sitting, SiSt and StSi were also analyzed using both rule-based detection and artificial neural networks (ANN) for comparison. The rule-based detection used mean and standard deviation of the acceleration vectors to classify postures and transitions. The results from the rule-based detection method was compared to the performance of a feed-forward neural network consisting of six hidden layers using a tan-sigmoid function. PCA was used for pre-processing to provide inputs for the ANN. The rule-based detection performed better in detecting the various lie-to-sit and sit-to-lie transitions while the ANN performed better in detecting SiSt and StSi. The main finding was that by combining transition analysis with orientation (posture) analysis the detection of postures was significantly improved.

2.5. Discussion

It has been shown in previous studies that a single accelerometer placed on the trunk can provide sufficient information about whole-body activities. Single accelerometers placed on wrists or ankles may perform better than the sensors placed on the trunk, depending on the types of activities to be recognized. The results from several studies show that by adding a second sensor, classification accuracy can be improved by 1-3%. Using three or more accelerometers does not necessarily improve performance.

A plethora of HAR models have been developed to recognize activities typical for healthy subjects. These models range from transparent rule-based approaches to more complex black-box ML approaches. Recently the latter approach has become more common with the rise of DL. Activities performed at very slow and irregular pace, such as by a symptomatic patient or an elder, are scarcely considered to design HAR models. With less intensity in movement, changes in the acceleration signal becomes smaller and more difficult to detect. Recognizing slow and irregular walking remains a challenge to be explored.

The few studies that developed HAR models for patient monitoring reported that their methods can provide additional information to patient self-report and standard care documentation. These studies either used empirically determined rule-based detection for classifying activities or supervised ML algorithms for recognizing postural transitions.

The objective of this thesis is to develop a deep learning classification model which can be used for monitoring mobility of hospitalised patients. The research question is the following: "How accurately can a deep learning algorithm recognize low-intensity activities, as performed by symptomatic patients, using a single trunk-worn accelerometer?"

The following sub-questions will also be considered regarding activity recognition using a single trunk-worn accelerometer:

- Is there a substantial difference between classification performance of deep learning and feature-based machine learning approaches?
- Can the sensor provide meaningful information for differentiating between walking with and without walking aid?
- How accurately can walking at slow paces be detected?

3

Methodology

The Simulated Hospital Study (SHS) is a data collection study that was conducted by Philips Research in 2019. The aim of the study was to collect measurements of human subjects in a simulated hospital environment. The protocol consisted of various typical daily activities of hospitalised patients. The study was approved by the Internal Committee for Biomedical Experiments at Philips Research. The SHS dataset contains around 40 hours of acceleration data, of which almost 20 hours are labelled, which was used for developing an activity recognition model aimed for patient monitoring. This chapter covers the data acquisition process, preprocessing of the data, as well as an introduction to the classification models described in later chapters.

3.1. Data Acquisition

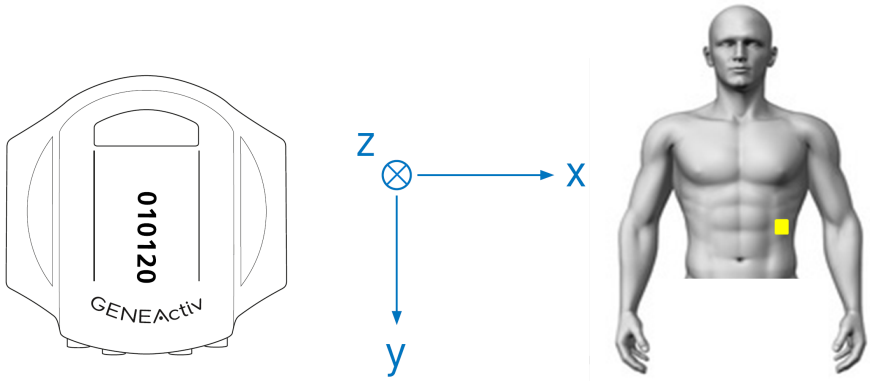
Twenty healthy subjects, ten males and ten females, aged in the range 23-63 years (M = 43.4 years, SD = 13.4 years) were recruited for the Simulated Hospital Study (SHS). Inclusion criteria for volunteers was age in range 18-65 years. Exclusion criteria were pregnancy, movement disorders, hypersensitivity to stainless steel and allergy to medical grade adhesives. Informed consent was obtained from all participants.

A GENEActiv (Activinsights Ltd., UK) sensor was attached to the trunk of a subject by using a medical grade double adhesive. The sensor contains a triaxial MEMS accelerometer with 12 bit resolution in the range of $+/-8g$ ($1g = 9.8m/s^2$). In addition, the sensor contains a rechargeable lithium polymer battery and 0.5 Gb non-volatile memory, enabling up to 7 days of continuous recording at 100 Hz sampling frequency. The accelerometer was placed on the left side of the trunk, just above the lowest rib. Figure 3.1 shows what the GeneActiv accelerometer looks like and the approximate placement of the sensor during the study. All sessions were recorded with a video camera for ground truth.

The protocol consisted of various ADLs as a patient in the hospital such as eating and drinking, lying in bed, performing physiotherapy exercises and walking with and without walking aids. A description of the protocol can be found in table 3.1. The subjects were asked to act as a patient in the hospital for all tasks except for the Ebbeling test, 6-Minute Walk Test (6-MWT) and the first Timed Up and Go (TUG) test. That is because these tests are used to determine fitness and performance of the test subject. The subjects were also equipped with other sensors during the protocol. One of the additional sensors was placed on a device that was hanging around subjects' necks. For one subject, the acceleration signals during the 6-MWT had multiple peaks, larger than for all other subjects. This was because the device hanging around the participant's neck was swinging back and forth during the brisk walking and sometimes colliding with the GeneActiv sensor. Because the 6-MWT acceleration data for this subject looked different from all other participants, it was removed from the dataset. Another subject was not able to complete the 6-MWT due to fatigue and therefore the 6-MWT data from that subject was also removed from the dataset.

Table 3.1: The complete study protocol. The duration is per participant.

Activity	**Description**	**Duration [mm:ss]**
Activities in and around bed		
Lie supine	Lying in bed on the back	3:00
Lie left	Lying in bed on the left side	0:30
Lie right	Lying in bed on the right side	0:30
Restless in bed	Turning in bed	1:00
Physiotherapy in bed	Lying on the back while doing leg exercises	1:00
Reclined	Lying in bed with the bed raised at 45 degree angle	0:30
Upright	Sitting in bed with the bed raised at 90 degree angle	0:30
Sitting edge of bed	Sitting at the edge of the bed	0:30
Standing next to bed	Standing next to the bed	0:30
Bed entry and exits	Repeated entering the bed and lying down and getting out and standing next to the bed	1:00
Treadmill activities		
0.4 km/h	Walking on the treadmill at 0.4 km/h	2:00
0.6 km/h	Walking on the treadmill at 0.6 km/h	2:00
0.8 km/h	Walking on the treadmill at 0.8 km/h	2:00
1.0 km/h	Walking on the treadmill at 1.0 km/h	2:00
1.2 km/h	Walking on the treadmill at 1.2 km/h	2:00
1.5 km/h	Walking on the treadmill at 1.5 km/h	2:00
2.0 km/h	Walking on the treadmill at 2.0 km/h	2:00
3.0 km/h	Walking on the treadmill at 3.0 km/h	2:00
4.0 km/h	Walking on the treadmill at 4.0 km/h	2:00
Ebbeling	Standardized treadmill fitness test	~10:00
Activities of daily hospital living		
Dressing/undressing	Putting on hospital gown over clothes, untying and retying shoes, taking hospital gown off	1:00
Reading	Sitting on a chair, while reading a book	1:00
Physiotherapy on a chair	Sitting on a chair while doing leg exercises	1:00
Eating/drinking	Sitting on a chair while pretending to eat and drink	1:00
STS	Repeated sitting down on a chair and standing up	1:00
Hospital ambulation		
Patient transport in wheelchair	Participant sitting down in wheelchair while being pushed by researcher	1:00
Washing hands brushing teeth	Participant washing hands and pretending to brush teeth	1:00
Crutches	Walking with crutches	1:00
Anterior walker	Walking with an anterior walker	1:00
IV pole	Walking with an IV-pole	1:00
4-wheel rollator	Walking with a 4-wheel rollator	1:00
Self propelled wheelchair	Participant moving him/herself forward in wheelchair	1:00
6-MWT	Standardized 6 min walk fitness test, walking a 30m course	6:00
Stair walking		
Stair ascent one leg injured	Walking upstairs one stair at a time, i.e. putting 2nd leg at same step	1:00
Stair descent one leg injured	Walking downstairs one stair at a time, i.e. putting 2nd leg at same step	1:00
Stair ascent	Walking upstairs	1:00
Stair descent	Walking downstairs	1:00



(a) (b)

Figure 3.1: (a) A drawing of the GeneActiv accelerometer used for data acquisition and its axes orientation. (b) The approximate placement of the GeneActiv accelerometer.

3.2. Activity Annotations

The activities were annotated manually and later made sure that the annotations were in sync with the acceleration signal. The parts of the acceleration signal which were unlabelled (between activities) were removed from the dataset. The activities of the protocol were categorized into the following seven activity classes:

1. **Lying in bed**: Lying on left side, right side, lying supine, transitioning between different lying postures, performing hip and knee physiotherapy exercises while lying on back and lying reclined (approximately 30° from horizontal).

2. **Upright**: Sitting upright in bed, standing, putting on and taking off shoes and hospital gown, reading a book while sitting on a chair, knee physiotherapy exercises while sitting on a chair, passive wheelchair transport (the subject sits in a wheelchair while one of the instructors push the wheelchair forward), eating and drinking, transitioning between sitting and standing.

3. **Walking**: Walking on treadmill (speeds ranging from 0.4-6 km/h), walking briskly in the corridor (6-minute walk test).

4. **Wheelchair**: Active wheelchair transport (the subject sits in a wheelchair and pushes himself/herself forward).

5. **Walking aid**: Walking in the corridor at a self-selected slow speed while using crutches, an anterior walker, IV-pole and a rollator.

6. **Walking up stairs**: Each subject was first instructed to walk up the stairs as if one leg was injured or recovering from surgery, such that the healthy leg should go up first, followed by the injured leg placed on the same step. The second time walking up stairs was at a self-selected slow pace. The subjects were encouraged to use the handrails for support.

7. **Walking down stairs**: Walking down stairs, same instructions are given to the subject as for walking up stairs.

3.3. Preprocessing

For ML classifiers, datasets must be split into a part which is used for training the model and a second part which is used to evaluate the model performance on unseen data, called holdout data. During training, the model learns to recognize patterns in the training data and how to associate these patterns with the given output label. During evaluation, these same rules and conditions that are realized during training are applied to the holdout data to predict output labels. In the case of deep neural networks (DNNs), the training phase is slightly different. The dataset is split into three parts; training data, validation data and holdout data. The

model is trained on the training data and the model evaluates its performance on the validation data to update its weights accordingly. The training of deep learning models is explained in further detail in chapter 5. Python libraries such as Pandas, Numpy, Sklearn and more were used in this work for data preprocessing.

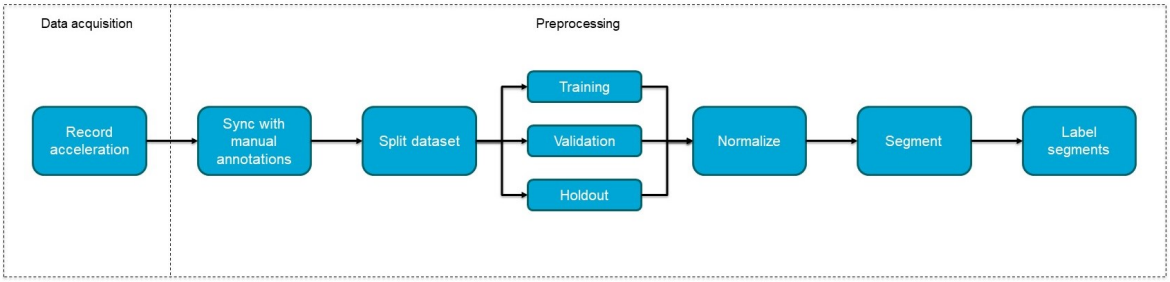


Figure 3.2: The preprocessing pipeline. After data acquisition, the acceleration signals were synced with manual annotations. The dataset was split into three smaller datasets, each one was normalized, segmented and labelled before used to train a classification model. For the feature-based machine learning approach, the pipeline is slightly different. Then the data is normalized after features have been extracted from each segment.

3.3.1. Splitting Dataset into Training and Holdout Datasets

For HAR it is important that the dataset is split into training dataset and holdout dataset based on subjects. The reason for this is because the acceleration data coming from each subject differs due to slightly different sensor orientation and subjects performing activities differently. If a model is trained on data that is randomly sampled from all subjects the model can learn how each patient performs each specific activity. This will result in good performance on the holdout data because the holdout data contains data that is similar to the training data. In this case the model will not be generalizable. That is, most likely the model will not perform well when predicting labels for a subject that it has not seen before.

The SHS dataset was split randomly based on subject IDs. Data from 25% of the subjects was randomly selected and kept aside as holdout data while the data from the other 75% subjects was used during training of the classification model. For the DNN, data from 50% of the subjects was used for training, 25% of the subjects for validation and 25% of the subjects for holdout evaluation.

3.3.2. Normalizing Data

A common normalization method for accelerometer-based HAR is by scaling data to have zero mean and unit variance. For the feature-based approach, the features were scaled according to this method, while for the deep learning approach, the raw acceleration data was scaled. The mean and variance parameters were computed from the training data, and same parameters used to normalize the validation and holdout datasets.

3.3.3. Segmentation

For activity recognition, accelerometer data should be segmented and each segment has an activity label assigned to it. Window sliding techniques are often used to segment the accelerometer data into windows of fixed length. Other segmentation techniques may involve automatic segmentation based on detected events.

There is no convention on how long segments to use for HAR. The optimal window length depends on the activities to be recognized and the types of features extracted from the data. Long window sizes do not necessarily lead to better classification performance. However, a too small value may not precisely capture the full characteristics of the activity. It has been reported that a window length of 1-2 seconds provides a good trade-off between performance and computational time [61]. However, due to the nature of activities included in the SHS trial, window length was set to 4 seconds. This is to make sure that relevant information is captured in each segment during activities like slow walking and wheelchair transport. Additionally, a 50% overlap between segments was used. Figure 3.3 shows examples of four segments from different activities, before and after preprocessing.

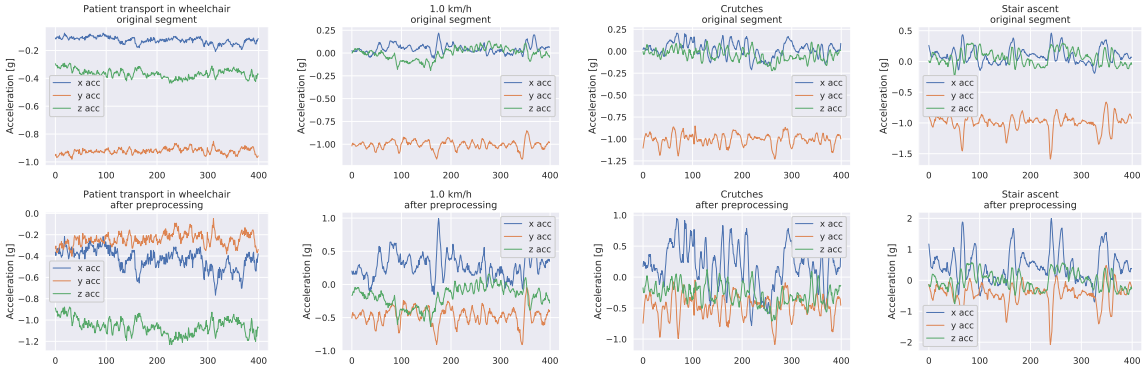


Figure 3.3: Examples of segments recorded from one subject before and after normalizing. The segments are recorded during (a) Active wheelchair transport, (b) Walking at 1 km/h on a treadmill, (c) Walking with crutches and (d) Stair ascent.

Labels were assigned to each segment and if a segment contained samples with more than one different labels, the segment got the label which was more prevalent in that segment. Between the controlled activities there were segments of unlabelled data. If a segment contained only unlabelled data or a majority of unlabelled data, it was not used for classification.

3.4. Handling Imbalanced Datasets

The SHS dataset is imbalanced as can be seen in Figure 3.4. The "Walking" class contains around 30 times more segments than the "Wheelchair" class. The challenge that follows imbalanced datasets is that a classifier would tend to prefer making predictions for the dominating class in order to achieve high accuracy. For example, a classifier applied to a dataset which has 9900 samples in one class and 100 samples in the second class can achieve 99% accuracy by simply predicting all training segments to the larger class. That would be a pointless classifier and this example shows that accuracy can be misleading in some cases. This is the reason why for imbalanced datasets, accuracy is often not a good measure for the performance of the classifier. There are other options for evaluating classifier performance such as precision, recall, F1-score and a confusion matrix. These metrics will be explained in the following section.

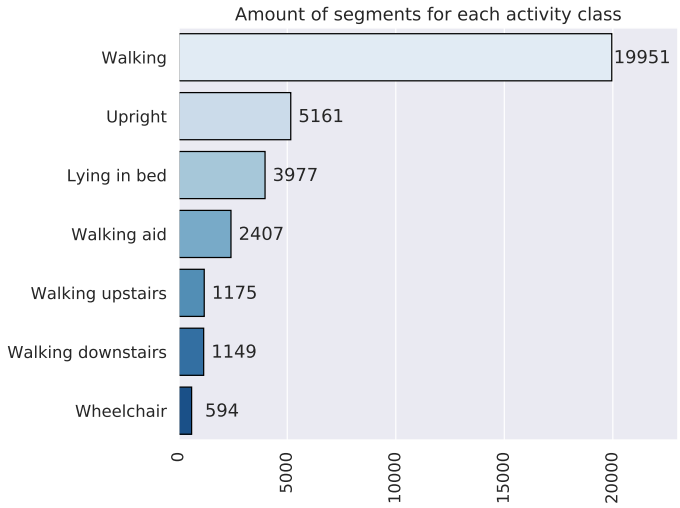


Figure 3.4: Distribution of amount of segments in each activity class. The "Walking" class is by far the largest class.

There are a few options to tackle classification problems with imbalanced data. A cost function can be used to penalise false positives of the majority class more than the smaller classes. Another option is resampling the dataset such that it becomes balanced. There are two options for resampling datasets for classification; undersample the larger classes to fit the size of the smaller classes or oversample the smaller classes to fit the size of the larger classes. The disadvantage of the undersampling strategy is that potentially useful data is discarded. The disadvantage of oversampling is that creating multiple copies of existing samples in minority classes can lead to overfitting [62]. A method called SMOTE (Synthetic Minority Over-sampling Technique) [63] is an alternative to creating exact copies of samples from minority classes. Instead, the algorithm creates new samples, based on the features of the samples belonging to minority classes.

3.5. Classification and Evaluation Metrics

Feature extraction is a critical step for all ML classifiers. Features are characteristics in data such as mean, standard deviation, Fourier coefficients and so on. The features determine the quality of the classifier. With poorly selected features, no matter the complexity of the classifier, the results will not be good. Three feature-based ML classifiers were tested on the SHS dataset. Those were Naïve Bayes, Random Forest and Support Vector Machine which all require hand crafted features. Chapter 4 presents the classification performance of these three models. Secondly, a deep neural network (DNN) was trained on the same dataset. The DNN model uses raw acceleration data and extracts features automatically. The networks can learn more high-level and meaningful features than the statistical hand-crafted features. Chapter 5 explains the development of the CNN classification model.

There are several metrics that can be used to evaluate classification performance; confusion matrix, accuracy, precision, recall, F1-score and logarithmic loss. All of these metrics are used throughout the rest of the thesis.

3.5.1. Confusion Matrix

A confusion matrix is a way to visualize right and wrong predictions of the model. One axis shows the amount of predictions made for each class while the other axis shows the actual amount of segments belong to each class.

There are four types of predictions for each class that can be seen in the confusion matrix. Figure 3.5 shows a confusion matrix highligting the "Walking" class. The four types of predictions that can be seen in the confusion matrix for the "Walking" class are the following:

- True positives (TP): Segments that were correctly classified as walking.
- True negatives (TN): Segments belonging to other classes that were correctly classified as not walking.
- False positives (FP): Segments belonging to other classes that were misclassified as walking
- False negatives (FN): Segments belonging to walking that were misclassified

3.5.2. Accuracy

Accuracy is the ratio of number of correct predictions to the total number of predictions made. This metric only works well if the dataset is relatively balanced.

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

3.5.3. Precision

Precision is the number of true positives divided by the number of predicted positives.

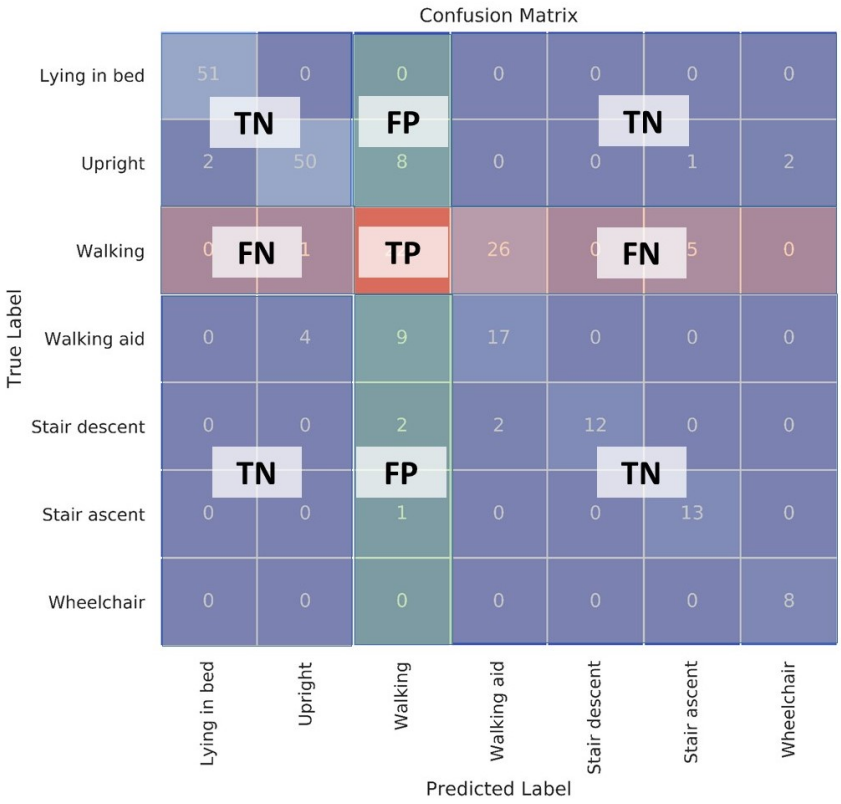


Figure 3.5: An example of a confusion matrix. TP, FP, FN and TN stand for True Positives, False Positives, False Negatives and True Negatives, respectively.

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

3.5.4. Recall

Recall is the number of true positives divided by the number of all samples that should have been predicted as positive.

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

3.5.5. F1-score

F1-score is the harmonic mean between precision and recall. It is a measure of how precise the model is, and is a better measure than accuracy when it comes to imbalanced datasets.

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

3.5.6. Logarithmic Loss

Logarithmic loss works by penalizing wrong predictions and works well for multi-class classifications. This metric is used for training the neural networks in chapter 5. In order to use Logarithmic Loss, the classification model must compute probabilities of each sample belonging to each class. The model aims to minimize the logarithmic loss. The equation is the following:

$$Logarithmicloss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} \cdot log(p_{ij}) \tag{3.5}$$

where N is the number of samples belonging to M classes, y_{ij} indicates whether sample i belongs to class j or not and p_{ij} indicates the probability of a sample i belonging to class j.

4

Feature-based machine learning approaches

This chapter presents a feature-based machine learning (ML) solution to recognize activities from the SHS dataset. A total of 86 features were extracted from each accelerometer segment. Features were normalized to zero mean and unit variance. Three types of ML classifiers were considered; Gaussian Naive Bayes, Random Forest and Support Vector Machine. The performance of the classifiers were evaluated on a holdout dataset and presented at the end of the chapter.

4.1. Feature Extraction

A total of 86 features, from both time and frequency domain, are extracted from each acceleration window. The features are listed in Table 4.1. Each feature is computed from the x-, y-, z-acceleration and the acceleration magnitude.

Acceleration Magnitude The acceleration magnitude, was computed at each data point. The acceleration magnitude is computed as follows:

$$a_{magn} = \sqrt{a_x^2 + a_y^2 + a_z^2} \tag{4.1}$$

Where a_x, a_y and a_z are acceleration values along the x, y and z axis, respectively.

Skewness (SK) is a measure of symmetry of data distributions. A normal distribution has a skewness of zero and any symmetric data also has a skewness around zero. Negative values of skewness indicate that more datapoints are placed to the left of the mean of the distribution. Skewness is computed as follows:

$$SK = \frac{E(x-\mu)^3}{\sigma^3} \tag{4.2}$$

Where $E(X)$ is the expected value of X, μ is the mean value of the distribution and σ is the standard deviation of that same distribution.

Kurtosis (KT) is a measure of whether the distribution is heavy-tailed or light-tailed, relative to a normal distribution. It is computed as follows:

$$KT = \frac{E(x-\mu)^4}{\sigma^4} \tag{4.3}$$

Table 4.1: The features extracted from each acceleration segment. Each feature was extracted from four signals; the x-, y-, z-acceleration and the acceleration magnitude.

Feature	**Description**
Mean	Mean value of the vector
Absolute mean	Mean of absolute values in the vector
Median	Median value of the vector
Mean absolute deviation	Mean absolute deviation of the vector
Standard deviation	Standard deviation of the vector
Variance	Variance of the vector
Minimum value	Lowest value in the vector
Maximum value	Highest value in the vector
Full range	Difference between the maximum and minimum value of the vector
Interquartile range	Difference between the 1st and 3rd quartile
Area	Sum of all values in the vector
Absolute area	Sum of all absolute values in the vector
Energy	Sum of squared components of the vector
Correlation	Correlation coefficients between each pair of vectors
Skewness	Shape of distribution
Kurtosis	Shape of distribution
Spectral entropy	A measure of the complexity of a signal
Spectral centroid	Mean of fourier transform
Spectral variance	Variance of fourier transform
Spectral skewness	Skewness of fourier transform
Spectral kurtosis	Kurtosis of fourier transform

Where $E(X)$ is the expected value of X, μ is the mean value of the distribution and σ is the standard deviation of that same distribution.

A Discrete Fourier Transform (DFT) is computed to extract the four frequency domain features; spectral centroid (mean), variance, skewness and kurtosis. The DFT of a discrete vector x is described as follows:

$$X(f) = \sum_{i=0}^{N-1} x_i \exp^{-j2\pi f i/N} \tag{4.4}$$

where X denotes the frequency spectrum, f the f^{th} Fourier coefficient and N the length of the vector x.

Spectral Entropy (SE) is defined to be the Shannon Entropy of the Power Spectral Density (PSD) of the data. It can be used as a measure of the complexity of a signal and is computed as follows:

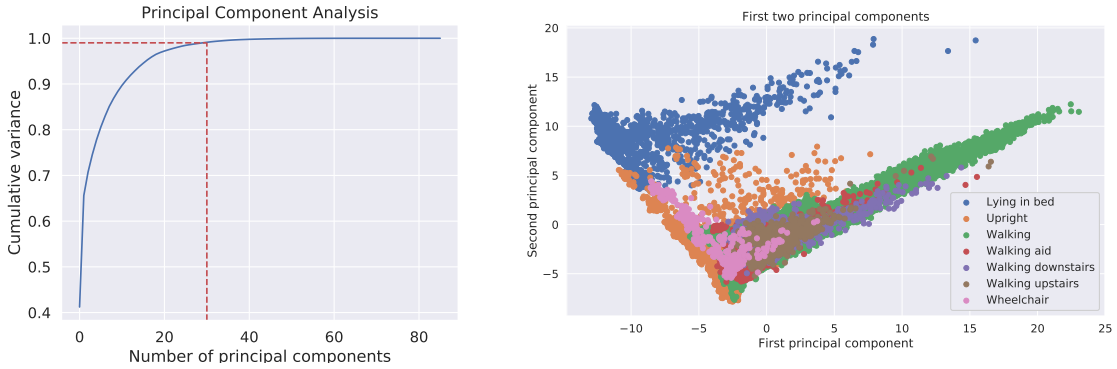
$$SE = -\sum_{x_i \in X} x_i \cdot \log_2 x_i \tag{4.5}$$

Where X is the frequency spectrum of a vector.

4.2. Feature Reduction with PCA

Prinicpal Component Analysis (PCA) has commonly been used for reducing dimensionality of a feature set used for HAR [36, 45, 60, 64]. PCA is an orthogonal transformation that creates a set of linearly uncorrelated variables called Principal Components. The first principal component accounts for the largest variance in the data set. The second principal component is constructed perpendicular to the first one, so that they are uncorrelated. For PCA to function properly, the features should be scaled such that their variances are in the same range. Therefore, the features are normalized to have zero mean and unit variance. Figure 4.1 shows

how reducing the feature dimensionality by half using PCA still maintains close to 100% of the variance in the data. By using the first 30 principal components, 99% of the cumulative variance of the original data can be maintained.



(a) (b)

Figure 4.1: Principal component analysis (PCA) of the extracted features. (a) shows that 99% of the cumulative variance in the original data can be maintained using only the first 30 principal components. (b) Shows how the first two principal components try to separate lying in bed from upright and walking activities.

4.3. Classifiers

4.3.1. Naïve Bayes Classifier

The Naïve Bayes (NB) classifier is a simple probabilistic classifier which is based on Bayes' Theorem: $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$. The classifier assigns a feature vector $\mathbf{x}$ to the class with the highest posterior probability $p(\omega_i|\mathbf{x})$, which is the probability of $\mathbf{x}$ belonging to a class ω_i. To do this, the classifier needs to estimate the class-conditional probability density function (pdf) $p(\mathbf{x}|\omega_i)$. The Gaussian NB classifier assumes that features follow a Gaussian distribution and therefore computes mean and standard deviation of features for estimating the class-conditional pdfs. The NB classifier assumes that the features are independent of each other, hence the name Naïve. In real-world data this is rarely a valid assumption, nevertheless, the NB performs well in classification tasks even when features show strong dependencies [65]. The NB classifier has been found to work better than other algorithms both in accuracy performance and computational time for HAR [66].

Implementation: The classifier was implemented using the GaussianNB function of the Sklearn library. The function estimates prior probabilities of classes according to the class distribution of the training data. PCA was used to reduce the dimensionality of the feature set and improved F1-score on holdout data from 0.60 to 0.74. However, class imbalance lead to the classifier predicting the majority class much more frequently than the minority classes (see Figures A.1 and A.2 in the Appendix). For this reason, SMOTE was applied to create new instances of the minority classes. This resulted in a more balanced prediction (see Figure A.3 in the Appendix), however, the F1-score decreased to 0.66.

4.3.2. Random Forest Classifier

The building block of the Random Forest (RF) [67] classifier is a decision tree. A decision tree classifier assigns a class label to an object via a sequence of decisions along the nodes of the decision tree, see Figure 4.2. At each node, a question is asked which splits the node into two descendants until a certain stop-splitting criterion is reached. Those nodes which fulfil the stop-splitting criterion are called leaves and each leaf is associated with one class label. During training, the decision tree classifier learns the relationship between features and class labels and thereby computes the best questions to be asked at each node. More details about decision trees can be found in [26].

A drawback of decision tree classifiers is their high variance [26]. Each time a decision tree is trained, even

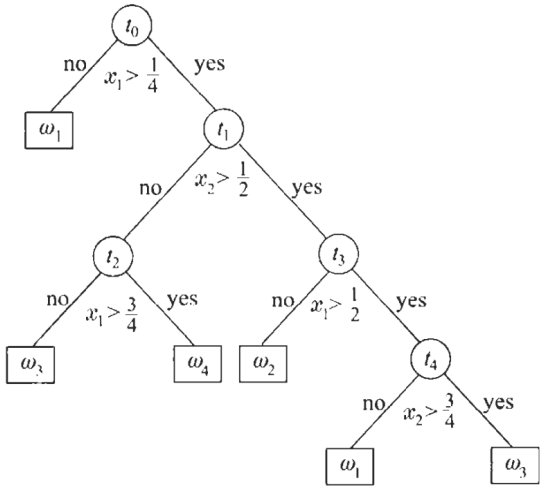


Figure 4.2: An illustration of a single decision tree. The circles marked with t_0, t_1, t_2, t_3, t_4, represent nodes which are split into two descendants based on the corresponding question. The question typically compares a feature (in this case x_1 or x_2) to a threshold value. The squares represent the leaves of the decision tree which assigns one of the four classes ($\omega_1, \omega_2, \omega_3$ or ω_4) to samples which end up in that leaf.

on the same data, it can result in different trees. The idea behind RF classifiers is combine multiple decision trees into a single model to get a more accurate and robust prediction. Each decision tree in the random forest model is constructed using a random subset of the available features and a random subset of the training data samples. The RF classifier will assign labels to samples based on the majority vote of all individual decision trees. This results in a more robust model than a stand-alone decision tree. RF classifiers have been shown to perform well in accelerometer-based activity recognition [16, 43, 68] and is therefore a suitable option for the presented classification problem.

Implementation: The RF classifier has its own feature selection method and therefore feature reduction is not needed in this case. The BalancedRandomForestClassifier (BRFC) function of the Imblearn library was used to implement a RF classifier. The BRFC randomly under-samples each subset used to construct its decision trees and is therefore well-suited for handling imbalanced datasets. A parameter grid and the RandomizedSearchCV algorithm of the Sklearn library was used to find a suitable combination of hyperparameters. The search algorithm randomly selects combinations provided by the parameter grid and performs cross validation with the training data provided to evaluate the performance of each combination. Once a proper selection of hyperparameters were found, the BRFC was evaluated on the holdout data, resulting in an F1-score 0.75. The confusion matrix can be found in Figure A.4 in the Appendix.

4.3.3. Support Vector Machines

Support vector machines (SVMs) define optimal hyperplanes to separate classes. If two classes are linearly separable, the optimal hyperplane is the one which provides the maximum distance (margin) from the hyperplane to data points of both classes. Support vectors are the data points that are closer to the hyperplane and influence the position and orientation of the optimal hyperplane (see Figure 4.3). Data is not always linearly separable as in Figure 4.3, therefore SVMs can use certain transformations, called kernels, to come up with a hyperplane which can best separate one class from other classes. SVMs perform well in high-dimensional spaces. A disadvantage of SVMs is the high computational burden, both during training and in the test phase.

Implementation: The classifier was implemented using the SVC function of Sklearn. A radial basis kernel was used and class weights inversely proportional to class size. The γ parameter of the SVM controls how closely-fit the hyperplanes of the classifer are to the training data. Therefore, to decreasing overfitting, the gamma parameter was reduced $\gamma = 0.001$. Using all 86 features resulted in slightly better classification per-

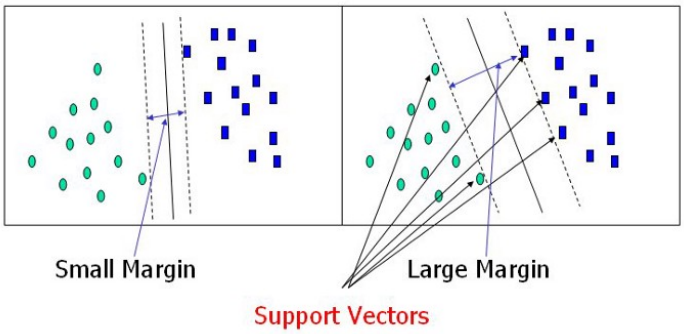


Figure 4.3: A support vector machine aims to select hyperplanes such that margins from support vectors to the margin are maximized.

formance than when only the first 30 principal components were used. Feature selection based on removing highly correlated features did not improve performance either. Therefore, the best SVM was found to be the one using the whole feature set and $\gamma = 0.001$, resulting in F1 score 0.81 on the holdout data. This classifier performed just as well on the training data (see confusion matrices in Figure A.5), which indicates good generalizability of the classifier.

4.4. Discussion

All four classifiers are trained on 75% of the SHS data (randomly selected 15 participants), and tested on the remaining 25% holdout data. Applying PCA improves performance for the NB classifier, while for the BRFC and SVM it slightly decreases performance. This can be explained due to the difference in complexity of the classifiers. The more complex classifiers perform well in high-dimensional data, while the NB benefits from the data transformation by the PCA.

Table 4.2 shows the Accuracy, weighted precision, recall and F1-scores of the three classifiers on the holdout data. The SVM outperforms the other classifiers in all metrics, however, it is also more computationally expensive.

Table 4.2: The performance scores of the three classifiers on the holdout data after tuning to reach better performance. The classifiers are Naïve Bayes (NB), Balanced Random Forest Classifier (BRFC) and Support Vector Machine (SVM) classifier.

	Accuracy	Precision	Recall	F1-score
NB	0.738	0.7492	0.738	0.7356
BRFC	0.7231	0.8394	0.7231	0.7515
SVM	0.7911	0.8577	0.7911	0.8096

Even though the NB classifier and BRFC seem to reach similar performance, the confusion matrices in the Appendix show that when looking at classification accuracy for each individual activity, the BRFC performs much better in recognizing the classes "Upright", "Walking aid", "Stair descent" and "Stair ascent". The NB classifier was not able to recognize the "Stair ascent" class, with only 6.6% of true positives for that class. Due to the NB classifier preferring predictions for the "Walking" class and obtaining a high accuracy for that class, the weighted precision, recall and F1-scores become high.

Due to the high scores of the SVM in all performance metrics and low degree of overfitting, the SVM with $\gamma = 0.001$ was chosen for comparison with the deep learning approach presented in the following chapter.

5

Deep Neural Network Model Architecture

This chapter gives an introduction to deep learning with neural networks and explains the two types of neural networks that have been successfully used for HAR. Those types of neural networks are Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks. Subsequently, the chapter covers the process of coming up with a suitable model architecture and hyperparameters.

5.1. Introduction to Deep Learning with Neural Networks

Deep neural networks (DNNs) is a class of deep learning algorithms that are able to automatically extract high-level features from raw input. DNNs have developed and advanced considerably in recent years and has brought about breakthroughs in fields such as visual object recognition and natural language processing [69]. DNNs have recently been applied to HAR problems with good results [56, 59, 70–72]. The advantage of using DNNs over conventional machine learning classifiers is that hand-crafted feature extraction is no longer needed.

The building blocks of neural networks is the neuron. It takes in inputs, multiplies each input with a certain weight and sums them all together, as illustrated in figure 5.1(a). The neuron fires a single output depending on an activation function. The most commonly used activation function is the Rectified Linear Unit (ReLU), shown in figure 5.1(b).

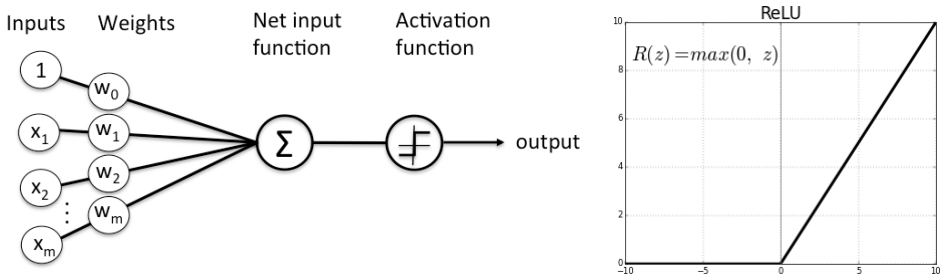


Figure 5.1: (a) An illustration of a neuron. The inputs x_1, x_2, ..., x_m (and input bias noted as 1) are multiplied with corresponding weights w_0, w_1, w_2, ..., w_m. The neuron sums together all inputs multiplied by their weights and outputs a value based on an activation function. (b) The ReLU activation function.

Many neurons together form a layer and layers form networks. The simplest form of neural networks is one using fully connected hidden layers. This means that every neuron in a hidden layer is connected to every input node and every neuron's output is fed to the subsequent layer. Figure 5.2 shows an example of such a network.

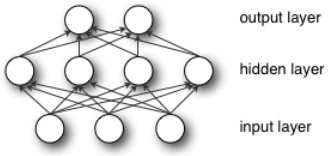


Figure 5.2: A simple neural network with one fully connected layer as a hidden layer.

During the training phase the neural network learns how to associate the input signal with the output signal by assigning its weights accordingly. Initially, all weights are assigned randomly and the model makes an initial prediction based on these random weights. The predicted values are compared to the actual output and an error, or loss, is computed, which represents the difference between the true and predicted values. The error is computed by a specified loss function, which can for example be the logarithmic loss described in section 3.5. The loss is fed back through the network by means of backpropagation [73] to provide a sort of feedback so the network can update its weights with the goal of minimizing the loss. For minimizing the loss, the network uses an optimization algorithm called gradient descent. This process is repeated multiple times until the model reaches convergence.

Two types of DNN structures have been shown to perform well in accelerometer based activity recognition in literature. These are convolutional layers, Long short-term memory (LSTM) layers and a combination of both. This chapter will explain these two structures and cover the process of developing a DNN model for HAR.

5.2. Convolutional Neural Networks

Convolutional neural networks (CNNs) [74] are a type of neural networks. CNNs have the advantages of local dependency and scale invariance which make them suitable for image processing and analysing sensor data such as accelerometer data.

There are three types of layers that make up CNNs; convolutional, pooling and dropout layers.

Convolutional layers The convolutional layers of CNNs are the feature extractors of the network. Convolutional layers are composed of neurons that are called filters. The filters have a fixed kernel size which determines the "receptive field" of the filter. Each filter slides across the input data and computes a dot product (convolution operation) between the filter weights and the input to return a feature map. Stride determines the step of filter from one convolution to the next. Figure 5.3 shows an example of how one filter with a kernel size equal to 5 slides across a 10x3 input segment and extracts a 6x1 feature map. Because the filter only moves in one direction, this is called one-dimensional (1D) convolution. The borders of the input data can be padded with zeros to maintain information at the edges, resulting in a 10x1 feature map.

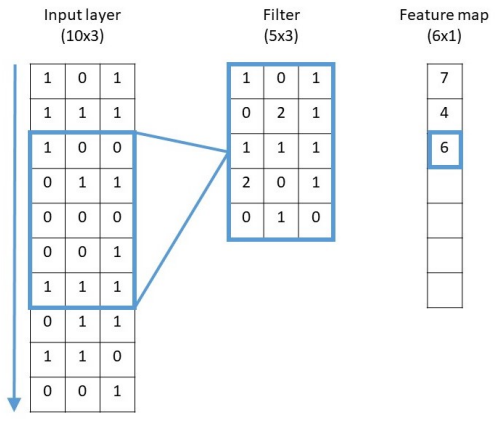


Figure 5.3: An illustration of a one-dimensional convolution with two-dimensional input data. A filter slides in one direction through the input data, at each location computes a dot product with the input data and returns a scalar.

Pooling layers Pooling layers are used to reduce the dimensionality of the feature map extracted by the convolutional layer which makes computation more efficient. The pool size determines to what extent the feature map is reduced. Max-pooling layers extract only the maximum value out of ps adjacent features, where ps is the pool size while the other $ps-1$ features are discarded. Another pooling option is using average-pooling which instead of extracting the maximum feature value, returns the average of the ps adjacent features. Pooling layers usually have strides equal to the pool size. Again, zero padding can be added to borders to maintain information at the edges.

Dropout Dropout layers drop, or ignore, randomly selected neurons. The dropout ratio determines the amount of neurons that are dropped. Using droput layers forces models to learn more robust features and thus can reduce overfitting.

5.3. Long Short-Term Memory (LSTM) layers

Long short-term memory networks [75] are a type of recurrent neural networks (RNNs). LSTM networks have the ability to learn long term dependencies in data. The building blocks of LSTM layers are units called memory cells, each composed of an input gate, a neuron with a self-recurrent connection, a forget gate and an output gate. The forget gate consists of a sigmoid activation function which is responsible for deciding whether new information entering the cell is relevant or can be ignored. The input gate consists of both a sigmoid and a hyberbolic tangent (tanh) activation functions. The purpose of the input gate is deciding what information to be stored or added to the memory cell. Lastly the output gate controls what information should be output from the cell. LSTM networks are well suited to analyse time series data, such as sensors data, where there can be long time periods between important events.

5.4. Model Architecture

The performance of a deep learning model depends heavily on hyperparameter selection. Deep learning models have a vast amount of hyperparameters and therefore a good practice when developing a deep learning model is to start with only a few layers and add layers one by one. The following hyperparameters were kept constant during model development:

Loss function: categorical crossentropy, or logarithmic loss as described in section 3.5. It is the preferred option for multi-class classification problems.

Optimizer: Adam [76] is a stochastic gradient-based optimizer. The optimizer typically requires little tuning and is a good choice as an optimizer for any classification problem.

Activation function: ReLU (Rectified Linear Unit) is a simple and efficient activation function. It is a common choice in hidden layers of deep neural networks.

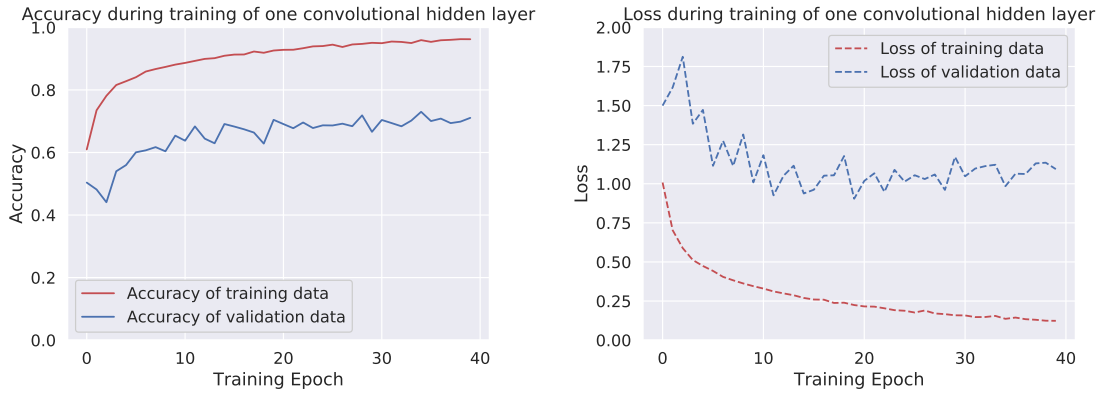
Batch normalization layers were added after each convolutional layer as it has been shown to be effective in accelerating training of deep networks by reducing internal covariate shift [77]. Internal covariate shift is the changes in input distribution of a learning system. The layer normalizes each batch to zero mean and unit variance, just as was done for the input data.

Because the SHS dataset is imbalanced, models were trained using a balanced batch generator the by imbalanced-learn library [78]. The purpose of the balanced batch generator is making sure that in each batch presented to the model during training, there is an equal amount of samples from each class. The batch generator does so by creating copies of randomly selected samples belonging to all classes except the majority class of the batch. Without the balanced batch generator the model learns to prefer predictions for the walking class because that class is by far the largest class, therefore by accurately predicting the walking class a

high accuracy score can be obtained.

During model development multiple different combinations of hidden layers were tested, while the input and output layers stayed the same. The input layer makes sure that the segment is presented to the hidden layers as a nx3 matrix where n is the window length and the three columns represent x-, y- and z-acceleration. The hidden layers extract features from the segments and feed the feature maps to an output layer. The output layer is a fully connected layer layer with seven neurons (one for each class) with a Softmax activation. The output layer returns posterior probabilities for each class. The model was developed using Keras with TensorFlow backend.

The first step in developing the DNN model architecture was starting with a single convolutional layer. Amount of filters, kernel size and stride are important hyperparameters for convolutional layers. To find a suitable combination of hyperparameters a grid search was performed with amount of filters $f_1 \in [4,8,16,32,64,128]$ and kernel size $ks_1 \in [3,5,7,9,11,13,15]$. Zero-padding was used to preserve information at the beginning and end of segments and stride was kept at $s = 1$ for a smaller parameter grid, resulting in less computational time. The best results were obtained using amount of filters $f_1 = 8$ and kernel size $ks_1 = 15$, both with regard to loss score for validation data and learning curves of the model during training. However, when pooling layers and dropout was added to this convolutional layer, no combination gave a good convergence of the validation loss curve. Therefore larger kernel sizes were tested, $ks_1 \in [17,19,21,23,25]$, while keeping the amount of filters $f_1 = 8$. The best results were obtained using kernel size $ks_1 = 23$. Figure 5.4 shows the learning curves of the model with one hidden convolutional layer with 8 filters and kernel size 23. This convolutional layer outputs a feature map the size of 400x8 for each input which is a 400x3 segment.



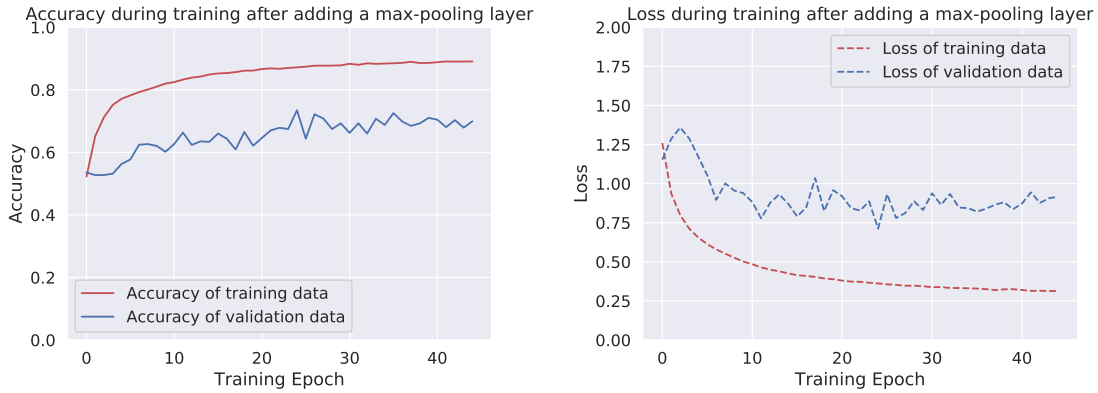
(a) (b)

Figure 5.4: Learning curves during training of a model with one hidden convolutional layer followed by batch normalization. (a) Accuracy of training and validation data increases as amount of training epochs increase. (b) Loss of training and validation data decreases as amount of training epochs increase.

Figure 5.4 shows that around the tenth epoch the model stops improving. The accuracy of the training data is close to 100% while the accuracy of the validation data is close to 75%. This is a sign of overfitting. A max-pooling layer was added after the convolutional layer in an effort to decrease the overfitting. A grid search was performed with pool sizes $ps_1 \in [2,4,5,10,20,50,100]$. The best results were obtained using a pool size of 10. The learning curves of this model can be seen in Figure 5.5.

After adding a max-pooling layer the validation loss decreases considerably while the training loss increases. The difference between the performance on training and validation data has decreased, so the max-pooling layer was effective in decreasing overfitting. Dropout layers can further decrease overfitting. However, since the max-pooling layer decreases the feature map drastically from 400x8 to 40x8, a dropout layer is not added after this max-pooling layer. Rather a new convolutional layer is added to extract new features.

This process was repeated, adding one layer at a time (convolution, max-pooling and dropout), each time

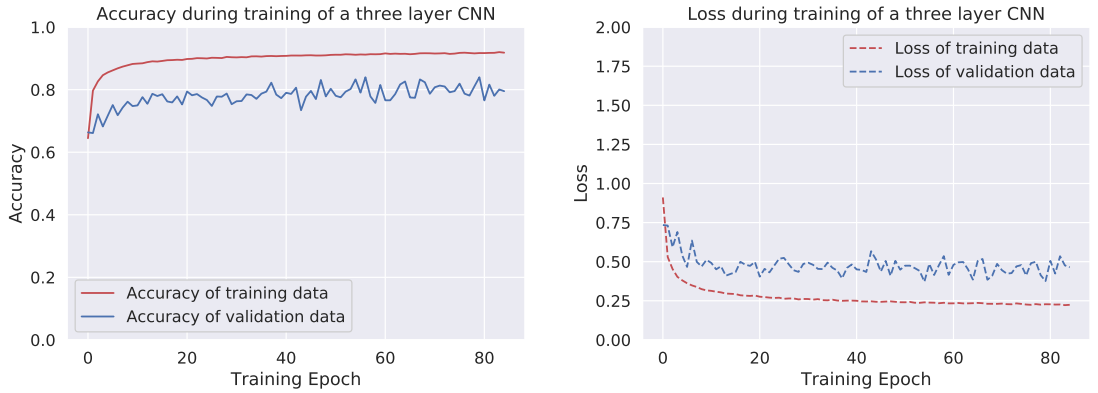


(a) (b)

Figure 5.5: Learning curves during training of a model with one hidden convolutional layer followed a max-pooling layer and batch normalization. (a) Accuracy of training and validation data increases as amount of training epochs increase. (b) Loss of training and validation data decreases as amount of training epochs increase.

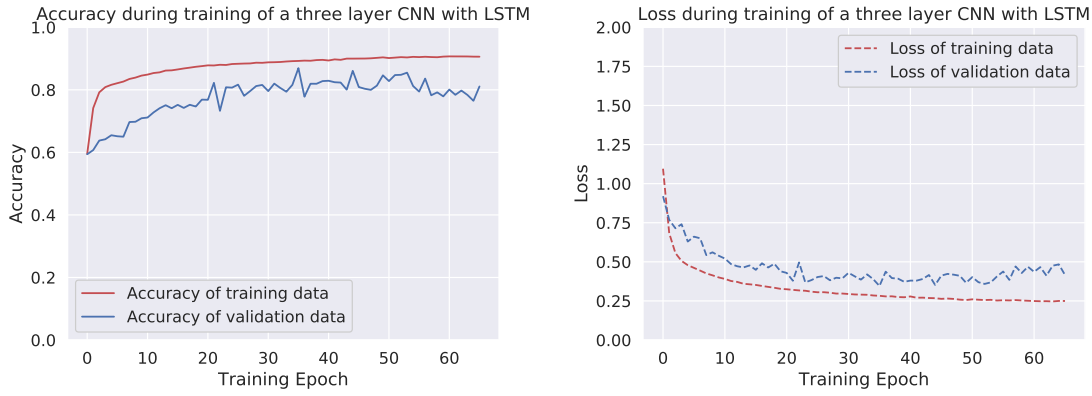
performing a gridsearch to find good combinations of layer hyperparameters. Table 5.1 shows the order in which layers were added, the parameter grid used to come up with a proper combination of hyperparameters and the selected hyperparameters. In the end, a three layer CNN structure is selected. After the third layer has been applied the size of the feature maps has become 2x16, therefore more convolutional layers were not added.

Li et al. [79] and Ordonez et al. [58] have shown that adding LSTM layers after a CNN network can improve model performance for HAR. Therefore an LSTM layer was added after the third dropout (and batch normalization) layer. A gridsearch was performed with amount of units $l \in [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16]$ with no dropout. An LSTM layer with 6 units was chosen as the best option. Adding the LSTM resulted in slightly better performance and less overfitting as can be seen in figure 5.7. Figure 5.11 shows an illustration of the model architecture.



(a) (b)

Figure 5.6: Learning curves during training of the Convolutional Neural Network (CNN) without a Long Short-Term Memory (LSTM) layer. (a) Accuracy of the 3-layer CNN model. (b) Loss of the 3-layer CNN model.



(a) (b)

Figure 5.7: Learning curves during training of the Convolutional Neural Network (CNN) after adding a Long Short-Term Memory (LSTM) layer. (a) Accuracy of the validation data improves after adding an LSTM layer to the CNN model and (b) Loss decreases.

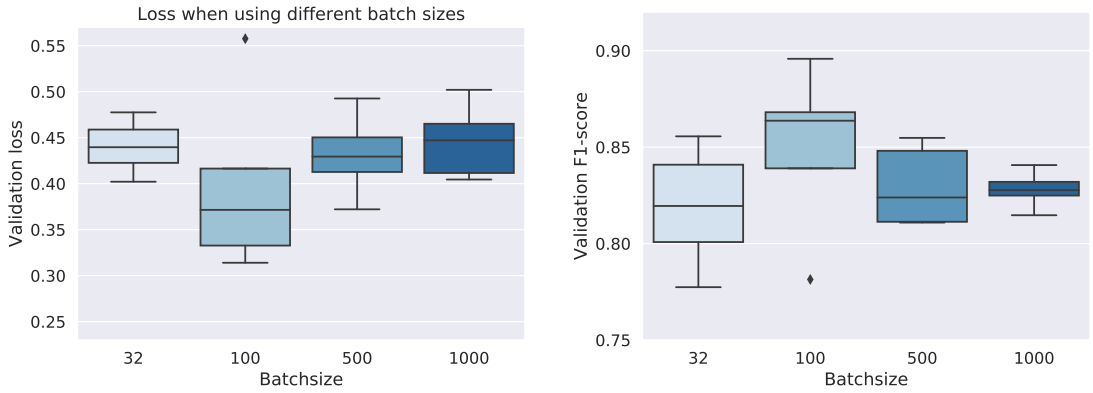
5.5. Improving Performance of Deep Neural Network by Hyperparameter Tuning

Deep learning models are stochastic, therefore, model performance differs when the model is retrained using the same data and same hyperparameters. This is because parameters such as initial weights and dropout are determined randomly which will lead to different results each time the model is retrained. In addition, the optimizer may find end up in different local minimas instead of ending up in the global minima in each run. In order to deal with this problem there are two solutions; repeating the training multiple times or set a seed for the random generator to make sure that the same random values are used each time. For the hyperparameter tuning in this section the first option was chosen because allowing different initial weights can lead to finding a better performing model. Three hyperparameters were tuned; batch size, scaler for input data and segment sizes.

5.5.1. Batch Size

In general, using larger batch sizes result in faster training, but the model doesn't always converge as fast. Smaller batch sizes train slower, but can converge faster. Batch sizes $b \in [32, 100, 500, 1000]$ were compared. The model was trained five times for each batch size as results can differ slightly between runs. Figure 5.8 shows the distribution of the validation loss and F1-score for each batch size.

Batch size of 100 samples for the batch generator gives the best loss and F1-score on the validation data and is therefore chosen for the subsequent analysis. It should be noted that this is the batch size of the Balanced Batch Generator, which then oversamples all minority classes to match the size of the largest class. Therefore the batch used for training the model will be somewhat larger than 100 segments.

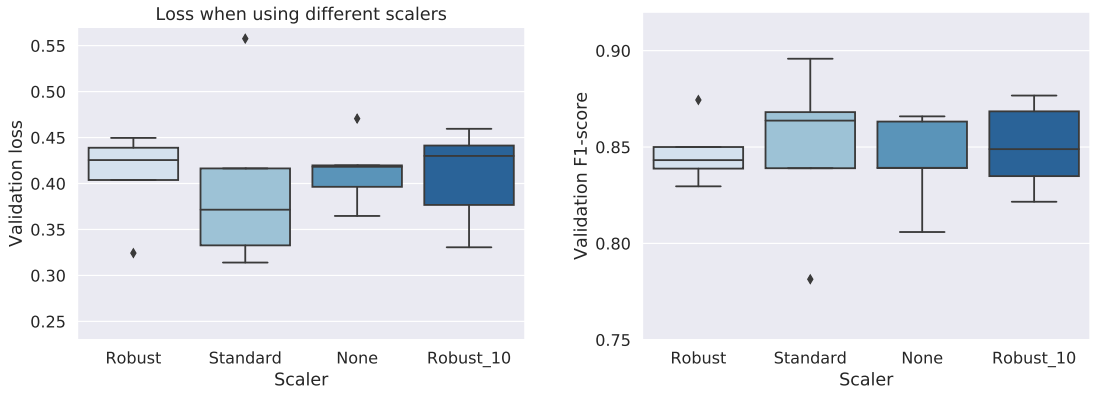


(a) (b)

Figure 5.8: Affect of different batch sizes on (a) Validation loss and (b) Validation F1-score. The model is trained five times using each batch size.

5.5.2. Scaling Input Data

Four different options were tested for scaling the acceleration input data. Those were a Robust scaler, Standard scaler, Minmax scaler and no scaling. The Robust scaler scales data within the interquartile range of the data provided. Robust_10 means that the data is scaled to fit within the 1st and 9th quantile. The Robust scaler is less sensitive to outliers in data than the Standard scaler and Minmax scaler.



(a) (b)

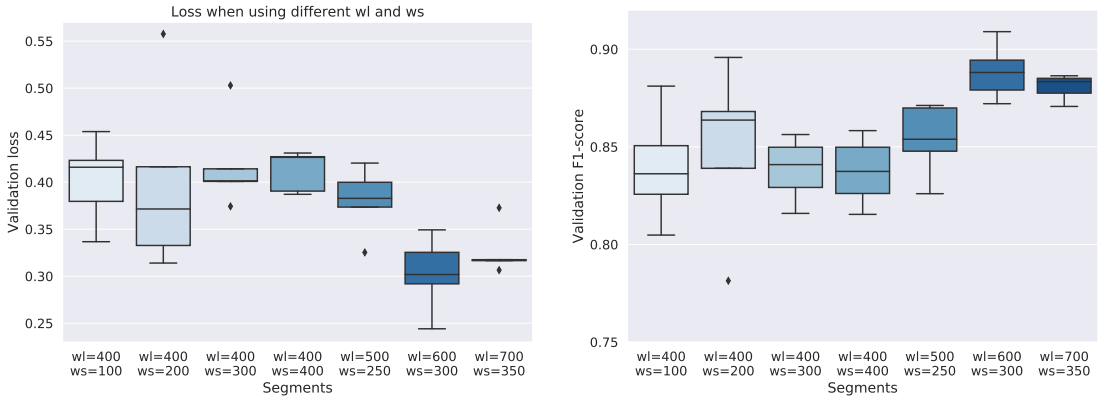
Figure 5.9: Affect of different scalers on (a) Validation loss and (b) Validation F1-score. The model is trained five times using each batch size.

Using the Minmax scaler did not result in a stable model. The validation loss and accuracy oscillated and did not converge. Scaling the data to a zero mean and unit variance (Standard scaler) performed better than Minmax, Robust and no scaling.

5.5.3. Window Sizes and Overlap

Increasing window size results in more information being captured per segment, which can be beneficial for activities performed at a slow pace. The downside of using larger window sizes is that it decreases the amount of samples available for training the model. On the other hand, increasing the overlap between segments results in an increase of segments available for training, although segments will share a lot of common information. Figure 5.10 shows the changes in validation loss and accuracy when changing the amount of overlap for the 4 second long windows and increasing window length, keeping 50% overlap. Using window

length of 600 samples (or 6 seconds) and 50% overlap turned out to improve performance compared to using 400 samples (4 seconds) and 50% overlap.



(a) (b)

Figure 5.10: Affect of using different window lengths (wl) and window steps (ws) on (a) Validation loss and (b) Validation F1-score. The model is trained five times using each batch size.

Table 5.1: Hyperparameter selection for the Deep Neural Network model

Number of layer	Type of layer	Parameter grid	Selected hyperparameters
1	Convolution	Filters $\in [4,8,16,32,64,128]$ Kernel size $\in [3,5,7,9,11,13,15,17,19,21,23,25]$	$f_1 = 8$ $ks_1 = 23$
2	Max pooling	Pool size $\in [2,3,5,10,20,40,100]$	$ps_1 = 10$
3	Batch normalization		
4	Convolution	Filters $\in [8,16,32,64]$ Kernel size $\in [5,10,15,23]$	$f_2 = 8$ $ks_2 = 10$
5	Max pooling	Pool size $\in [2,4,5,8,10]$	$ps_2 = 4$
6	Dropout	Ratio $\in [0.1,0.2,0.3,0.4,0.5]$	$d_2 = 0.3$
7	Batch normalization		
8	Convolution	Filters $\in [2,4,8,16]$ Kernel size $\in [1,2,3,4,5,6,7,8,9,10]$	$f_3 = 16$ $ks_3 = 7$
9	Max pooling	Pool size $\in [1,2,3]$	$ps_3 = 2$
10	Dropout	Ratio $\in [0.1,0.2,0.3,0.4,0.5]$	$d_3 = 0.3$
11	Batch normalization		
12	LSTM	Units $\in [1,2,3,4,5,6,7,8,9,10]$	$l = 6$

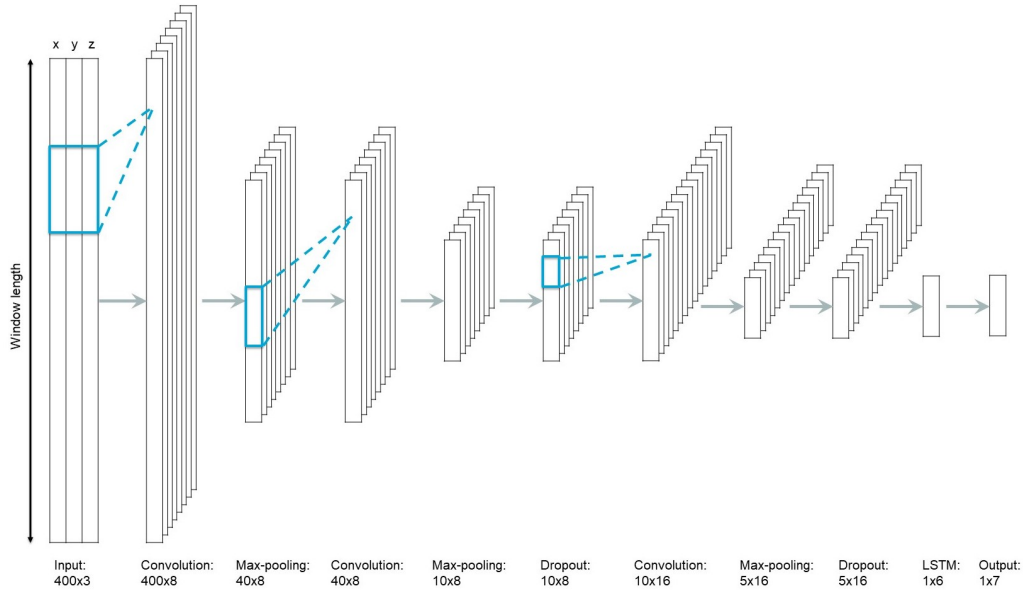


Figure 5.11: The model architecture of the final Deep Neural Network model. The batch normalization layers are not shown for simplicity. The size of the feature maps after each layer are noted at the bottom.

6

Results and Evaluation

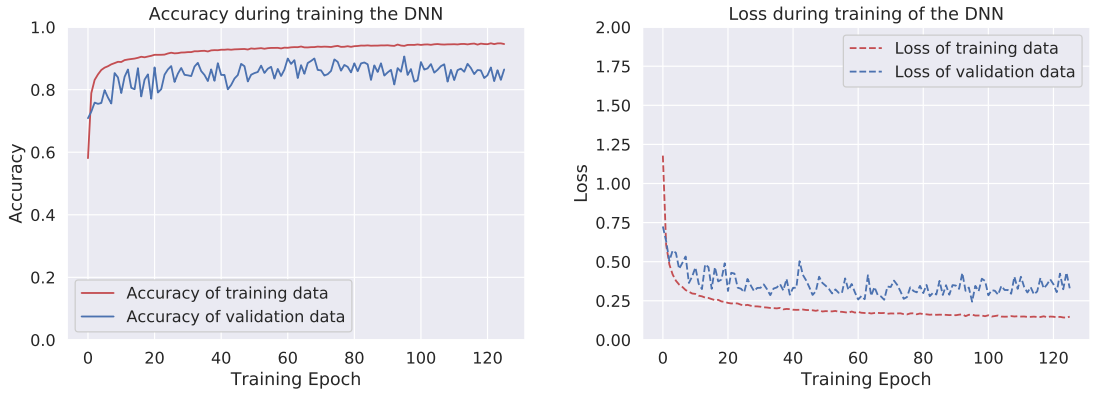
This chapter presents the classification performance of the DNN model on the SHS dataset. The second section presents the results of the SVM for comparison, this time using 6 second long windows. Both models were evaluated on the same holdout dataset belonging to 5 out of the 20 participants in the SHS dataset. The third section presents the performance of both models when the "Walking aid" class is combined with the "Walking" class. Lastly, the classification performance of both models are tested on a complete session from one of the SHS subjects, including the unlabelled segments in between activities.

6.1. Deep Neural Network (DNN) Classification Performance

The performance of the DNN model on predicting labels for the holdout data is evaluated using batch size of 100 segments and 6 second long windows with 50% overlap. Table 6.1 shows the improvement in classification performance before and after hyperparameter tuning. Figure 6.1 shows the loss and accuracy for both training and validation data during training of the model.

Table 6.1: Comparison of classification performance of the Deep Neural Network before and after hyperparameter tuning. wl represents window length in seconds and b represents batch size used for the BatchGenerator.

	Accuracy	Precision	Recall	F1-score
$wl = 4$ and $b = 500$	0.867	0.888	0.867	0.8748
$wl = 6$ and $b = 100$	0.9034	0.9036	0.9034	0.9019



(a) (b)

Figure 6.1: Learning curves during training of the Deep Neural Network (DNN) after hyperparameter tuning. (a) Accuracy of the training and validation data (b) Loss of the training and validation data.

The DNN model with updated hyperparameters reached higher accuracy and lower loss compared to the learning curves before, as shown in Figure 5.7. The predictions of the updated model is shown in Figure 6.2.

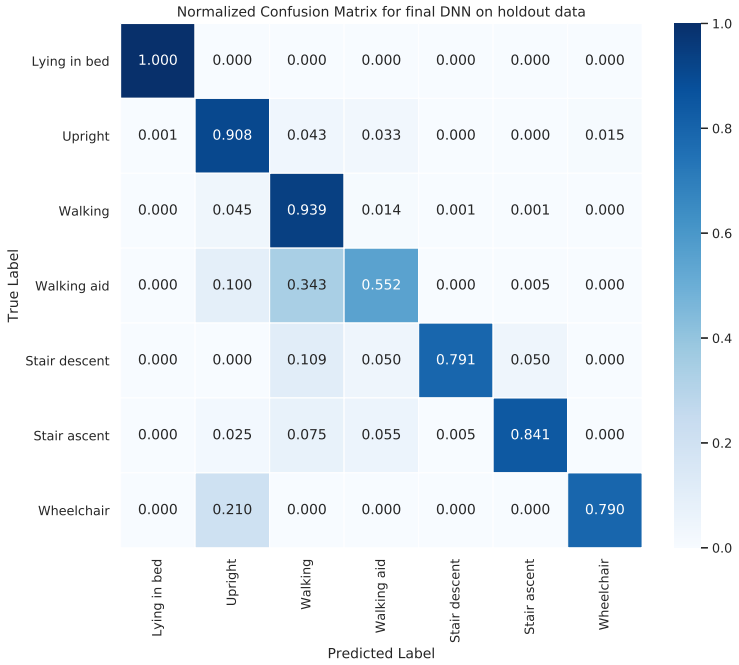


Figure 6.2: The normalized confusion matrix for the Deep Neural Network (DNN) on holdout data after updating hyperparameters.

The confusion matrix in Figure 6.2 shows that the "Walking aid" class is the most difficult to predict. The DNN misclassified 34.3% of "Walking aid" segments with "Walking". Similarly, 10.9% of "Stair descent" and 7.5% of "Stair ascent" were misclassified as "Walking". 21% of "Wheelchair" segments were misclassified as "Upright". This could be due to some segments maintaining a rather constant velocity which would lead to the acceleration signal looking similar to when a person is sitting still. Figure 6.3 shows a break-down of the wrong predictions made by the DNN model. Each sub-activity that wasn't 100% correctly classified is listed along the x-axis and the different coloured parts of the bar represents which class the model predicted. The complete list of sub-activities is shown in table 3.1. This enables visualising the accuracy of detecting walking at different speeds and whether certain types of walking aid is easier to classify than others.

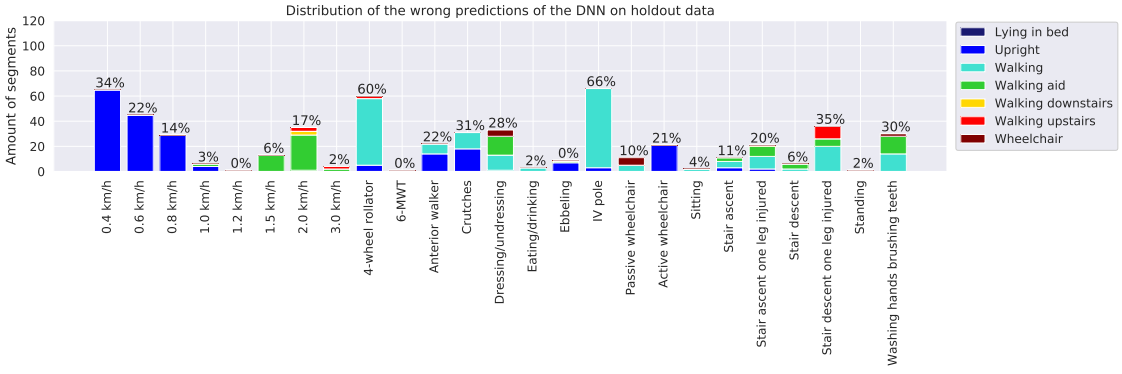


Figure 6.3: Wrong predictions of the Deep Neural Network (DNN) on holdout data, sorted by sub-activity. The height of the bars represent amount of segments that are misclassified and the colors represent that class the model predicted. The percentages explain the ratio of wrongly predicted segments to total amount of segments per sub-activity.

The sub-activities treadmill walking at 0.4 km/h, walking with a 4-wheel rollator and an IV pole are those

that were most frequently misclassified, with around 60 segments assigned wrong labels. However, those three activities were performed for a different duration, with treadmill walking activities 2 minutes each and the "Walking aid" sub-activities were 1 minutes each per participant. Therefore the two "Walking aid" sub-activities are substantially more difficult to classify than others, with 60% of all walking with a 4-wheel rollator segments and 66% of walking with an IV pole classified as "Walking" instead of "Walking aid". This can be explained by the similarity of the activities. Especially for the IV-pole, the subjects are walking slowly while holding the pole with one hand. The acceleration of the trunk will therefore look similar to when walking slowly. When walking at 0.4 km/h the DNN misclassifies 34% of segments as "Upright". The walking detection improves as the speed increases to 1.2 km/h. At 1.5 and 2 km/h, the DNN starts misclassifying some segments as belonging to the "Walking aid" class. This could possibly be an indication that the features that the model is learning for the "Walking aid" class are not robust and are possibly speed-related. For this reason a final evaluation is performed with the two classes "Walking" and "Walking aid" combined as one class.

6.2. Classification Performance of the Support Vector Machine (SVM)

Since classification performance of the DNN model improved when using larger window length, the performance of the SVM using window length $wl = 6$ was evaluated on the holdout dataset. Table 6.2 shows that using 6 second long windows improved the F1-score from 0.8096 to 0.8211, compared to the 4 second long windows. Figure 6.4 shows the confusion matrix of the SVM on the holdout data using 6 second long segments with 50% overlap.

Table 6.2: The performance scores of the Support Vector Machine (SVM) classifier on the holdout data using 4 and 6 second long windows, respectively. wl represents window length.

	Accuracy	Precision	Recall	F1-score
$wl = 4$	0.7911	0.8577	0.7911	0.8096
$wl = 6$	0.8046	0.8631	0.8046	0.8211

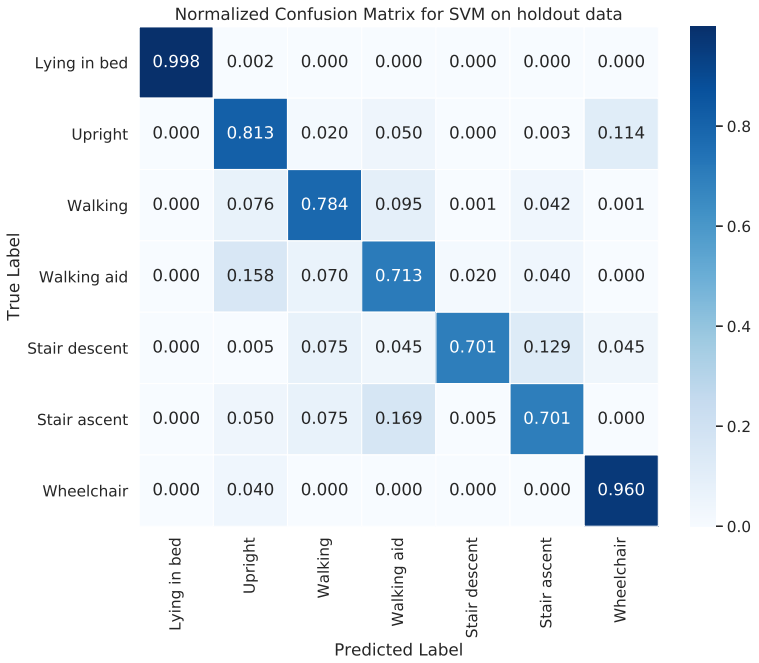


Figure 6.4: The normalized confusion matrix of the Support Vector Machine (SVM) on holdout data using 6 second long segments.

The "Wheelchair" class was correctly predicted for 96% of the segments which is a substantially better accuracy rate than the DNN for that class. Instead, the SVM misclassifies 11.4% of "Upright" segments as

"Wheelchair". The two stair walking classes and "Walking aid" are the classes which are most difficult to recognize. Figure 6.5 shows the break-down of the wrong predictions by the SVM classifier to compare with the DNN in figure 6.3.

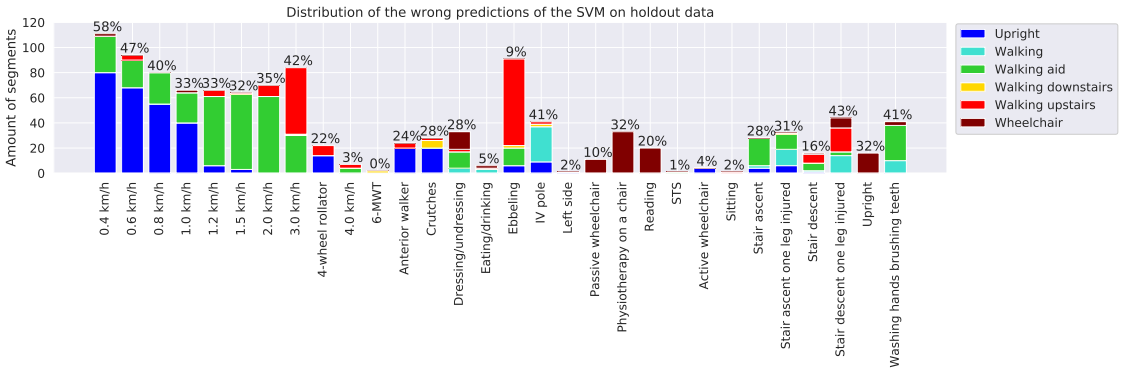


Figure 6.5: Wrong predictions of the Support Vector Machine (SVM), sorted by sub-activity. The colors represent the wrongly predicted class. The percentages explain the ratio of wrongly predicted segments to total amount of segments per sub-activity.

The SVM misclassifies parts of all treadmill walking activities as "Walking aid", mostly at speeds between 1.2-2 km/h. The DNN model also showed wrong predictions for the "Walking aid" class at speeds 1.5 and 2 km/h, however, substantially less than the SVM. Both classification models tend to misclassify the slower walking paces (<1 km/h) as "Upright". Although, the DNN is better at detecting slow walking. For the DNN model, the "4-wheel rollator" and "IV pole" are two of the mostly misclassified sub-activities, while for the SVM the treadmill walking and Ebbeling activities are those with the largest amount of wrong predictions.

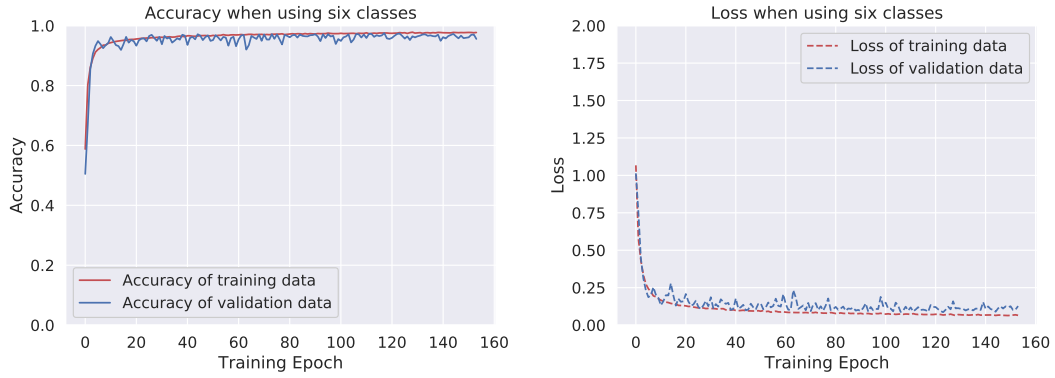
6.3. Classification Performance After Combining two Walking Classes

The confusion between the classes "Walking" and "Walking aid" suggests that the acceleration data from the trunk-worn sensor is not sufficient to extract meaningful and robust features to differentiate between the two activities. For a more robust HAR model the two classes should be combined. Table 6.3 shows the performance of both the DNN model and SVM classifier on holdout data once the "Walking" and "Walking aid" classes have been combined to one. The DNN performance scores are substantially higher than of the SVM.

Table 6.3: Classification performance of Support Vector Machine (SVM) and Deep Neural Network (DNN) after reducing number of classes to six

	Accuracy	Precision	Recall	F1-score
SVM	0.8414	0.893	0.8414	0.8562
DNN	0.9452	0.9507	0.9452	0.9464

Figure 6.6 shows the accuracy and loss when the DNN model was retrained with the edited labels. The margin between the scores on the training and validation data became much less than when using all seven classes (see Figure 6.1). Figure 6.7 shows the confusion matrix of this DNN model.



(a) (b)

Figure 6.6: Learning curves during training of the Deep Neural Network (DNN) after combining "Walking aid" with the "Walking" class. (a) Accuracy of the training and validation data (b) Loss of the training and validation data.

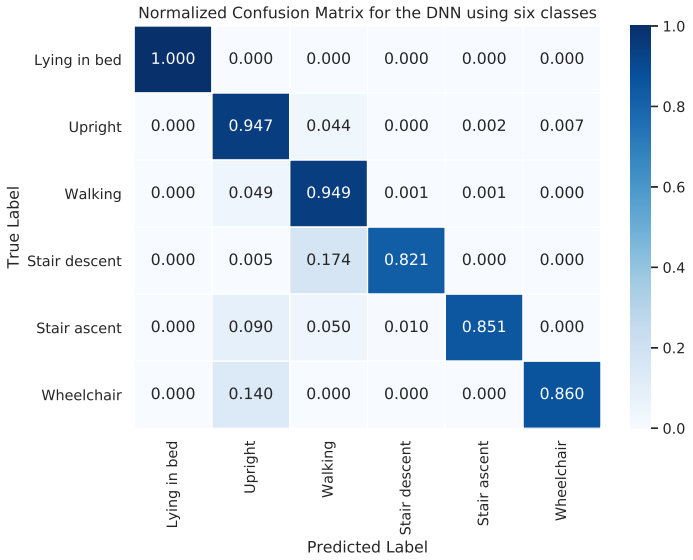


Figure 6.7: The normalized confusion matrix showing the predictions of the Deep Neural Network (DNN) after combining the "Walking aid" with the "Walking" class.

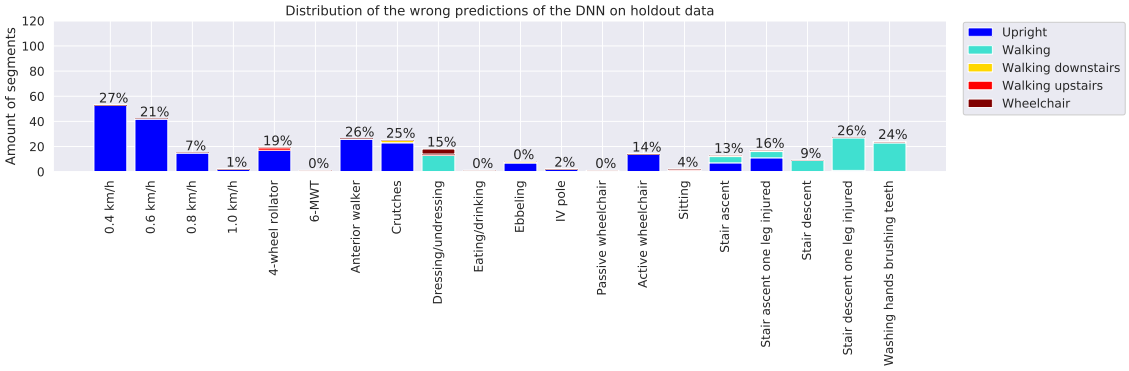


Figure 6.8: Wrong predictions of the Deep Neural Network (DNN) after after combining the "Walking aid" with the "Walking" class, sorted by sub-activity. The colours represent the wrongly predicted class. The percentages explain the ratio of wrongly predicted segments to total amount of segments per sub-activity.

Walking detection at slow paces improved for the DNN model by reducing the amount of classes to six. The misclassified segments for walking at 0.4 km/h decreased from 34% to 27%.

Figure 6.9 shows the confusion matrix of the SVM after the number of classes has been reduced to six and Figure 6.10 shows the break-down of the wrong predictions.

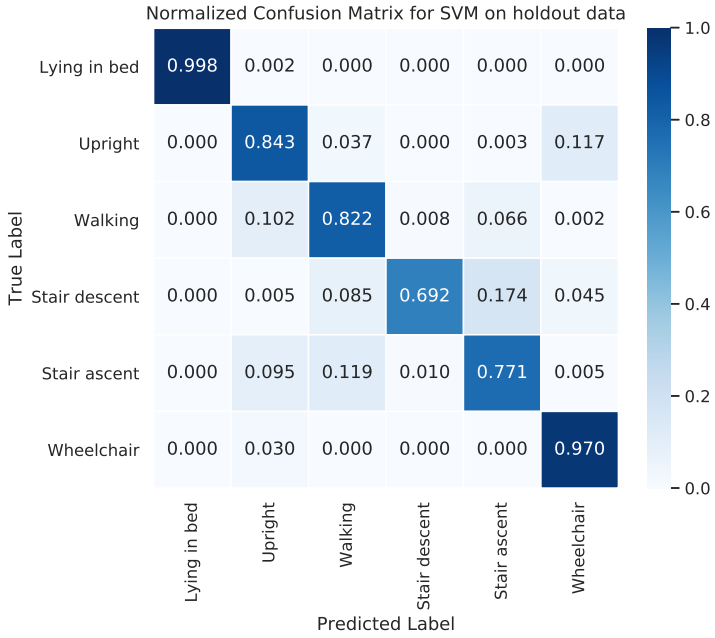


Figure 6.9: The normalized confusion matrix showing the predictions of the Support Vector Machine (SVM) after combining the "Walking aid" with the "Walking" class.

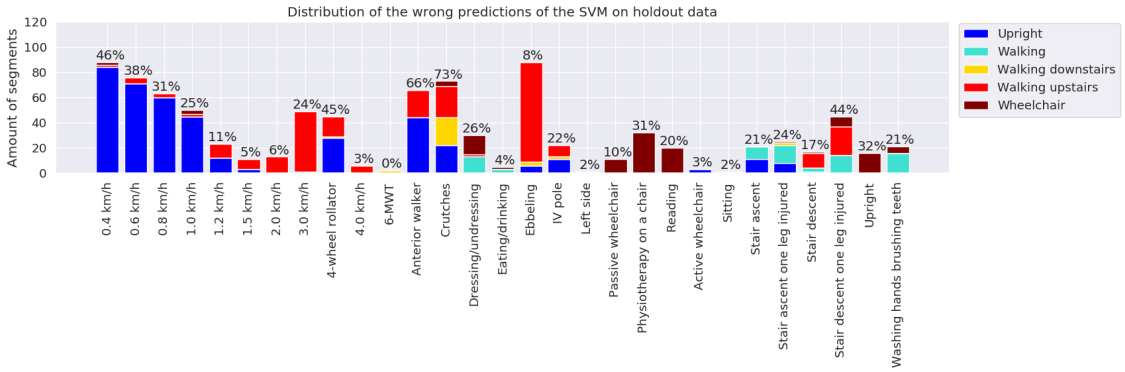


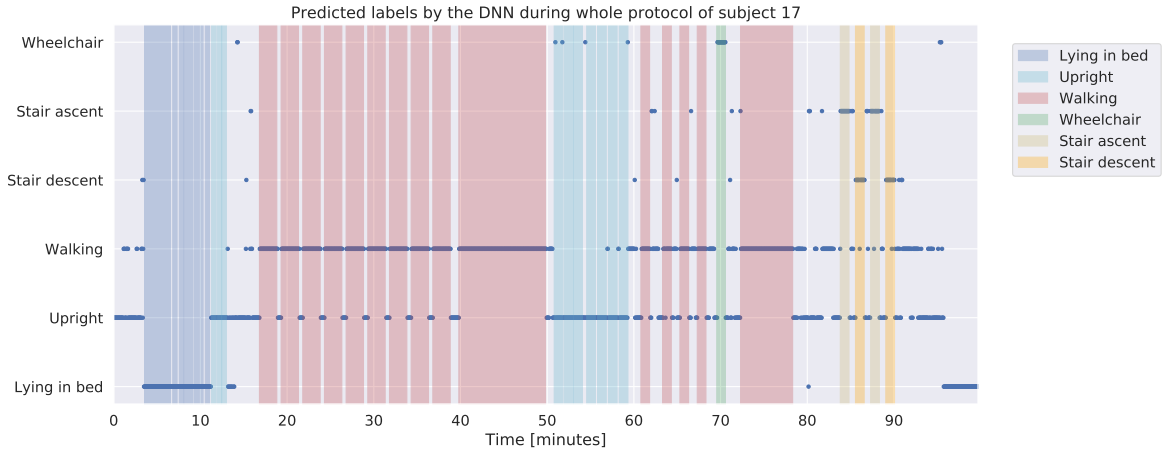
Figure 6.10: Wrong predictions of the Support Vector Machine (SVM) after combining the "Walking aid" with the "Walking" class, sorted by sub-activity. The colors represent the wrongly predicted class. The percentages explain the ratio of wrongly predicted segments to total amount of segments per sub-activity.

The slow walking detection also improved for the SVM by reducing number of classes to six. However, it is clear that the total number of wrong predictions is substantially less for the DNN compared to the SVM.

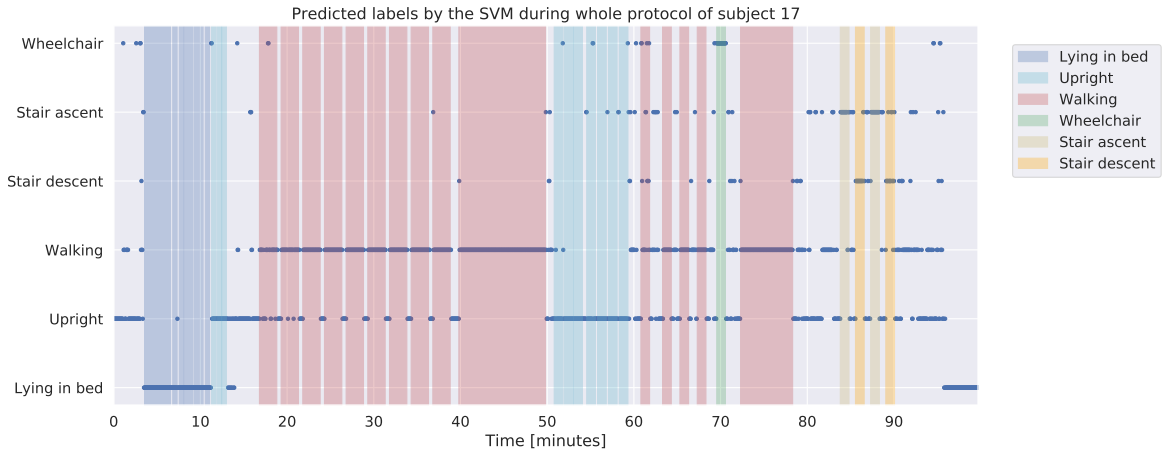
Classification improved for all six remaining classes, which can be expected when decreasing amount of classes.

6.4. Qualitative Evaluation of Classifier Performance on Continuous Data

The two classification models were used to predict labels for segments during a complete recording session of one subject, including the unlabelled data between activities, which the model has not seen before. This is to test the models' generalizability and performance on real continuous data. The data from the complete session was segmented with a sliding window approach, returning 6 second long segments with 50% overlap. For the SVM classifiers, features were extracted as explained in Chapter 4. Figure 6.11 shows the predictions for each segment during the continuous recording session of subject 17, which was one of the subjects in the holdout dataset. The grey background represents unlabelled time segments while the coloured areas represent labelled activities as specified in the legend. The height of the blue dots represents the prediction of a segment at a given time.



(a)



(b)

Figure 6.11: Predictions of the classification models when the whole recording session of subject 17 is passed into the model. The grey areas represent unlabelled activities. a) Deep Neural Network (DNN) predictions and b) Support Vector Machine (SVM) predictions

The unlabelled data should mostly fall either into the walking or upright class. Exceptions include the last few minutes where the sensor has been removed from the subject and around the stair walking activities. Both models exhibit false predictions during the unlabelled activities for this particular subject. The three uppermost activity classes, "Wheelchair", "Stair ascent", and "Stair descent" are the ones that should be limited only to the green, yellow and orange areas, respectively. The amount of predictions belonging to these three classes outside of their marked areas indicates the classification accuracy.

The first half of the protocol consists of activities in and around the bed followed by treadmill walking. Both models have very few false predictions during these parts of the protocol, only a few one-off false predictions for the uppermost three classes. During the second half of the protocol, the SVM misclassifies a few of the ADL activities (light blue) as "Stair ascent". Both models predict several segments as stair walking in between the four "Walking aid" activities, which most likely should be "Walking".

7

Concluding Remarks

7.1. Conclusions

This thesis presented the development of a Deep Neural Network (DNN) classification model for patient activity recognition. Data was collected from 20 healthy subjects that were asked to perform the activities slowly and act as if they were patients in the hospital. Seven activity classes were chosen to represent typical ADLs of hospitalised patients. Those activities were lying in bed, upright posture, walking, walking with walking aid, active wheelchair transport and walking up and down stairs. The DNN model reaches an F1-score 0.9019 on holdout data. A Support Vector Machine (SVM) classifier achieved an F1-score 0.8211 on holdout data for comparison. Both models showed a large number of misclassfications between the walking class and walking with walking aid. By combining those two classes the F1-scores became 0.9464 and 0.8562 for the DNN and SVM models, respectively. The results indicate that the deep learning approach is substantially better than the feature-based machine learning approach.

The DNN model presented in this report is a reliable algorithm for recognizing low-impact activities that are typical of patient ADLs in the hospital. The model can accurately detect walking at speeds down to 1 km/h. For 0.4 km/h, the slowest speed recorded in this study, the model recognizes around 2/3 of the segments from the holdout data as walking, the rest is classified as sitting/standing upright. This work showed that a single trunk-worn accelerometer has the potential to monitor mobility of patients in hospitals. This would provide the nurses and doctors insight into the recovery process of their patients and valuable objective information for making decision about discharging patients.

7.2. Future Recommendations

There are endless possibilities of combinations of hyperparameters for the DNN model and only a few of them were explored in this thesis. Further tuning by using finer parameters grids and considering other hyperparameters such as amount of filters, kernel sizes, pool sizes in each layer and more might improve the performance of the DNN model. Only sequential DNN models were considered in this work. Other structures similar to the multi-headed model presented in [56] might also perform well for this classification problem. Another option would be to develop a hierarchical model that first predicts posture such as lying or upright. Then depending on the prediction of the first layer a second model can further determine whether the patient is standing still or walking. A third layer could be added to predict which type of walking activity the patient is performing. Other possible improvements would be to apply use data sharpening similar to Cho et al. [55]. That might improve performance in detecting walking at slow speeds. With better walking detection it becomes possible to further evaluate the quality of walking, for example stability.

Adding a sensor to patients' ankle might improve walking detection during slow walking or an accelerometer on patients' wrists for walking aid detection. However, adding more sensors can add discomfort for the patient. Accelerometers placed on walking aids might be a better option in cases where it is considered important to correctly detect walking with walking aids. In order to distinguish between sitting and standing, a separate model could be used for detecting postural transitions such as SiSt, StSi and other position changes.

This can be achieved by looking at the acceleration waveform as Godfrey et al. demonstrated in [32], or by analysing barometric pressure sensor signals as presented by Masse et al. in [80]. In cases of monitoring immobile patients, monitoring postural transitions while lying in bed is relevant for pressure ulcer prevention.

Differences in sensor wearing position and orientation might affect the classification performance. A possible approach to circumvent this problem could be to use a model that can predict the sensor's orientation, as in [50] or use signal vector magnitude for recognizing activities. By using signal vector magnitude the orientation of the sensor does not affect the signal. Signal vector magnitude alone does not provide sufficient information for posture analysis, only differentiating between static and dynamic activities and the ability to classify dynamic activities.

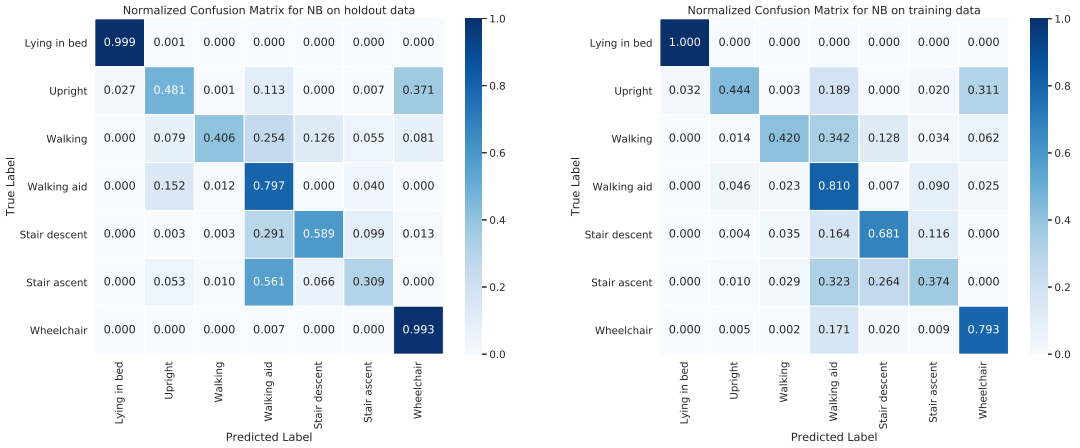
Post-processing can be applied to the predictions of the model to increase robustness. As an example, adding logic such as which sequence of activities are possible or requiring a minimum duration for an activity to be predicted. This could reduce the fluctuations from predicting one class to another. Another option would be to look closer into the class posterior probabilities returned by the softmax output layer. Instead of simply selecting the class with the highest probability in each case, there could be a condition about how certain the prediction should be. For example if there is no significant difference between class posterior probabilities for "Upright" and "Wheelchair" the model should pick the "Upright" class, unless it had made certain predictions that the previous segment was "Wheelchair".

This thesis showed that it is feasible to recognize low-intensity patient activities using a single accelerometer worn on the trunk. A next step is to collect real patient data in a free-living environment at a hospital. Trends in patient recovery can easily be identified with a reliable patient activity recognition system, giving nurses and doctors a better overview of their patients' day-to-day improvements.

A

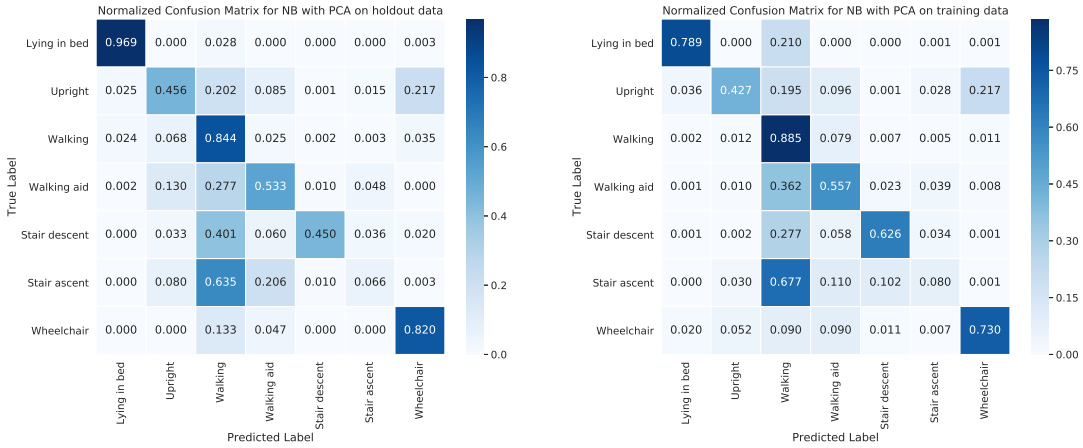
Confusion Matrices of the Feature-based Machine Learning Classifiers

A.1. Gaussian Naïve Bayes Classifier



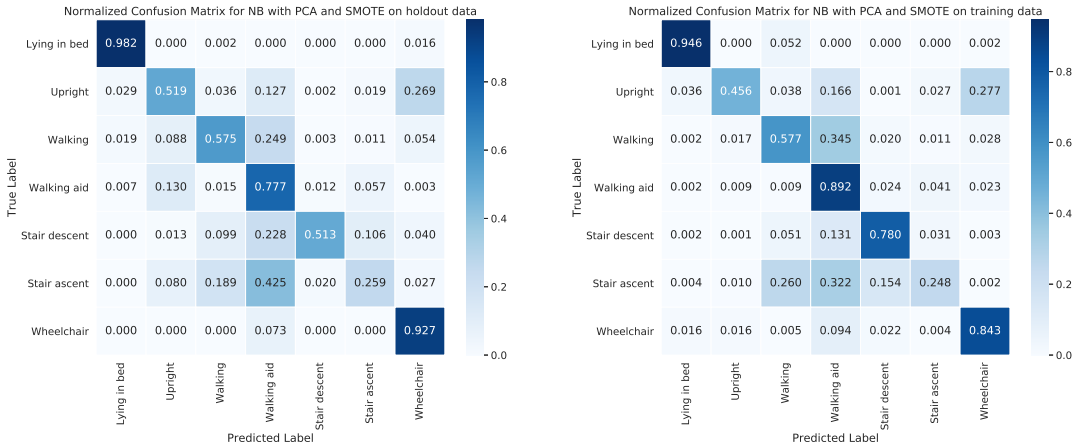
(a) (b)

Figure A.1: The normalized confusion matrices of the Naïve Bayes (NB) classifier using all 86 available features on (a) Training dataset and (b) Holdout dataset.



(a) (b)

Figure A.2: The normalized confusion matrices of the Naïve Bayes (NB) classifier when PCA was applied to reduce the dimensionality of the feature set from 86 to 30. The confusion matrices show predictions for the (a) Training dataset and (b) Holdout dataset.



(a) (b)

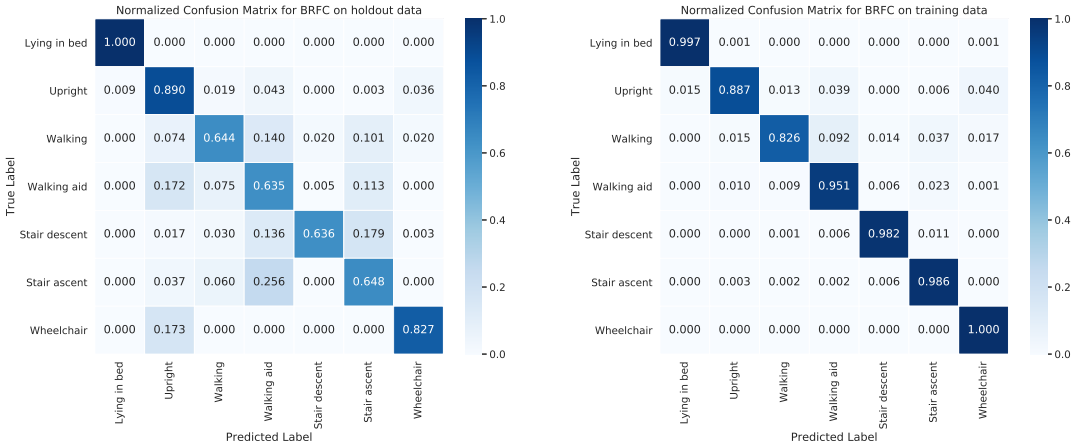
Figure A.3: The normalized confusion matrices of the Naïve Bayes (NB) classifier when PCA was applied to reduce the dimensionality of the feature set from 86 to 30 and SMOTE used to upsample minority classes. The confusion matrices show predictions for the (a) Training dataset and (b) Holdout dataset.

A.2. Balanced Random Forest Classifier

Table A.1 shows the parameter grid selected to search a suitable combination of hyperparameters and the selected values.

Table A.1: The hyperparameters tested and selected for the Balaced Random Forest Classifier (BRFC)

Hyperparameter	**Parameter grid**	**Selected value**
Criterion	Gini, Entropy	Gini
Maximum depth	25, 50, 75, 100, None	25
Minimum samples per split	2, 10, 30, 50	2
Minimum samples per leaf	1, 2, 5, 10, 20, 30, 40	1
Maximum features	10, 30, 60, None	None
Minimum impurity decrease	0, 1e-6, 1e-5, 1e-4	1e-6
Number of estimators		500
Class weights		Balanced subsample
Sampling strategy		Not majority



(a) (b)

Figure A.4: The normalized confusion matrices of the Balanced Random Forest classifier (BRFC) using the parameters resulting in the highest score after randomized search. The confusion matrices show predictions for the (a) Training dataset and (b) Holdout dataset.

A.3. Support Vector Machine

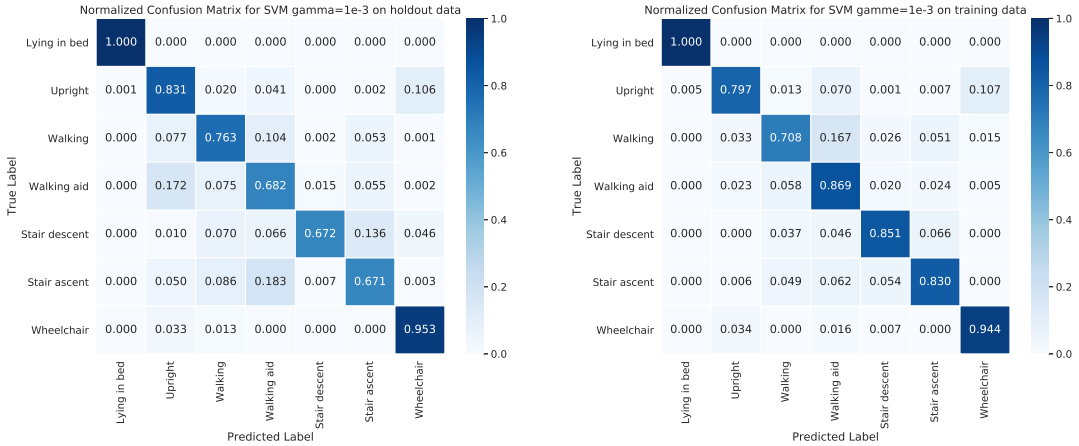


Figure A.5: The normalized confusion matrices of the Support Vector Machine (SVM) classifier when the γ parameter was reduced to 0.001. The confusion matrices show predictions for the (a) Training dataset and (b) Holdout dataset.

Bibliography

[1] H. M. Ghomrawi, L. M. Baumann, S. Kwon, F. Hebal, G. Hsiung, K. Williams, M. Reimann, C. Stake, E. K. Johnson, and F. Abdullah. Using accelerometers to characterize recovery after surgery in children. *Journal of Pediatric Surgery*, 53(8):1600–1605, aug 2018.

[2] R. Sallis, Y. Roddy-Sturm, E. Chijioke, K. Litman, M. H. Kanter, B. Z. Huang, E. Shen, and H. Q. Nguyen. Stepping toward discharge: Level of ambulation in hospitalized patients. *Journal of Hospital Medicine*, 10(6):384–389, 2015.

[3] D. T. Ubbink, E. Tump, J. A. Koenders, S. Kleiterp, J. C. Goslings, and F. E. Brölmann. Which Reasons Do Doctors, Nurses, and Patients Have for Hospital Discharge? A Mixed-Methods Study. *PLoS ONE*, 9(3), 2014.

[4] K. M. Culhane, G. M. Lyons, D. Hilton, P. A. Grace, and D. Lyons. Long-term mobility monitoring of older adults using accelerometers in a clinical environment. *Clinical Rehabilitation*, 18(3):335–343, may 2004.

[5] A. E. Mattlage, S. A. Redlin, M. A. Rippee, M. G. Abraham, M. M. Rymer, and S. A. Billinger. Use of Accelerometers to Examine Sedentary Time on an Acute Stroke Unit. *Journal of Neurologic Physical Therapy*, 39(3):166–171, 2015.

[6] M. S. Dhillon, S. A. McCombie, and D. B. McCombie. Towards the Prevention of Pressure Ulcers with a Wearable Patient Posture Monitor Based on Adaptive Accelerometer Alignment. In *2012 ANNUAL INTERNATIONAL CONFERENCE OF THE IEEE ENGINEERING IN MEDICINE AND BIOLOGY SOCIETY (EMBC)*, IEEE Engineering in Medicine and Biology Society Conference Proceedings, pages 4513–4516. IEEE; Engn Med & Biol Soc (EMBS); CAS; SMC; PubMed; MEDLINE, 2012.

[7] P. J. Agostini, B. Naidu, P. Rajesh, R. Steyn, E. Bishay, M. Kalkat, and S. Singh. Potentially modifiable factors contribute to limitation in physical activity following thoracotomy and lung resection: a prospective observational study. *Journal of Cardiothoracic Surgery*, 9(1):128, dec 2014.

[8] L. Browning, L. Denehy, and R. L. Scholes. The quantity of early upright mobilisation performed following upper abdominal surgery is low: An observational study. *Australian Journal of Physiotherapy*, 53(1):47–52, 2007.

[9] T. J. Daskivich, J. Houman, M. Lopez, M. Luu, P. Fleshner, K. Zaghiyan, S. Cunneen, M. Burch, C. Walsh, G. Paiement, T. Kremen, H. Soukiasian, A. Spitzer, T. Jackson, H. L. Kim, A. Li, and B. Spiegel. Association of Wearable Activity Monitors With Assessment of Daily Ambulation and Length of Stay Among Patients Undergoing Major Surgery. *JAMA Network Open*, 2(2):e187673, 2019.

[10] D. J. Cook, J. E. Thompson, S. K. Prinsen, J. A. Dearani, and C. Deschamps. Functional Recovery in the Elderly After Major Surgery: Assessment of Mobility Recovery Using Wireless Technology. *The Annals of Thoracic Surgery*, 96:1057–1061, 2013.

[11] B. Celler, T. Hesketh, W. Earnshaw, and E. Ilsar. An instrumentation system for the remote monitoring of changes in functional health status of the elderly at home. *Proceedings of 16th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 908–909.

[12] J. Chung, G. Demiris, and H. J. Thompson. Mobility limitation in community-dwelling older adults: a systematic review. *Communicating Nursing Research*, 45:426, 2012.

[13] G. Appelboom, A. H. Yang, B. R. Christophe, E. M. Bruce, J. Slomian, O. Bruyère, S. S. Bruce, B. E. Zacharia, J.-Y. Reginster, E. S. Connolly, O. Bruyere, S. S. Bruce, B. E. Zacharia, J.-Y. Reginster, and E. S. Connolly Jr. The promise of wearable activity sensors to define patient recovery. *Journal of Clinical Neuroscience*, 21(7):1089–1093, jul 2014.

[14] J. Pärkkä, M. Ermes, P. Korpipää, J. Mäntyjärvi, J. Peltola, and I. Korhonen. Activity Classification Using Realistic Data From Wearable Sensors. *IEEE TRANSACTIONS ON INFORMATION TECHNOLOGY IN BIOMEDICINE*, 10(1), 2006.

[15] A. Moncada-Torres, K. Leuenberger, R. Gonzenbach, A. Luft, and R. Gassert. Activity classification based on inertial and barometric pressure sensors at different anatomical locations. *Physiological Measurement*, 35(7):1245–1263, jul 2014.

[16] J. Zhu, R. San-Segundo, and J. M. Pardo. Feature extraction for robust physical activity recognition. 7(1):16, dec 2017.

[17] B. Barshan and M. Cihan Yüksek. Recognizing Daily and Sports Activities in Two Open Source Machine Learning Environments Using Body-Worn Sensor Units Handling editor: Ethem Alpaydin. 2013.

[18] F. Massé, R. Gonzenbach, A. Paraschiv-Ionescu, A. R. Luft, K. Aminian, F. Masse, R. Gonzenbach, A. Paraschiv-Ionescu, A. R. Luft, and K. Aminian. Wearable Barometric Pressure Sensor to Improve Postural Transition Recognition of Mobility-Impaired Stroke Patients. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 24(11):1210–1217, nov 2016.

[19] F. H. Beevi, J. Miranda, C. F. Pedersen, and S. Wagner. An Evaluation of Commercial Pedometers for Monitoring Slow Walking Speed Populations. *Telemedicine and e-Health*, 22(5):441–449, may 2016.

[20] R. De Ridder and C. De Blaiser. Activity trackers are not valid for step count registration when walking with crutches. *Gait and Posture*, 70:30–32, 2019.

[21] I. Cleland, B. Kikhia, C. Nugent, A. Boytsov, J. Hallberg, K. K. K. K. Synnes, S. McClean, and D. Finlay. Optimal Placement of Accelerometers for the Detection of Everyday Activities. *Sensors*, 13(7):9183–9200, jul 2013.

[22] S. Khatun and B. I. Morshed. Fully-Automated Human Activity Recognition with Transition Awareness from Wearable Sensor Data for mHealth. In *2018 IEEE International Conference on Electro/Information Technology (EIT)*, pages 0934–0938. IEEE, may 2018.

[23] L. Gao, A. K. Bourke, and J. Nelson. Evaluation of accelerometer based multi-sensor versus single-sensor activity recognition systems. *Medical Engineering and Physics*, 36(6):779–785, 2014.

[24] M. Awais, L. Chiari, E. A. F. Ihlen, J. L. Helbostad, and L. Palmerini. Physical Activity Classification for Elderly People in Free-Living Conditions. *IEEE JOURNAL OF BIOMEDICAL AND HEALTH INFORMATICS*, 23(1):197–207, jan 2019.

[25] Y.-T. Ching, C.-C. Cheng, G.-W. He, Y.-J. Yang, C.-C. Cheng, Y.-J. Yang, Acm, G.-W. He, and Y.-J. Yang. Full Model for Sensors Placement and Activities Recognition. In *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers on - UbiComp '17*, Proceedings of the 2017 Acm International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 Acm International Symposium on Wearable Computers, pages 17–20, New York, New York, USA, 2017. ACM Press.

[26] S. Theodoridis and K. Koutroumbas. *Pattern Recognition, Fourth Edition*. Academic Press, Inc., Orlando, FL, USA, 4th edition, 2008.

[27] J.-N. Kim, M.-H. Ryu, Y.-S. Yang, and J. Chang. Static Posture and Dynamic Activity Classification with a Holter-Mounted Accelerometer. *Sensors and Materials*, 27(8):585–591, 2015.

[28] J. Liu, J. Sohn, and S. Kim. Classification of Daily Activities for the Elderly Using Wearable Sensors. *Journal of Healthcare Engineering*, 2017:1–7, 2017.

[29] K. Aminian, P. Robert, E. E. Buchser, B. Rutschmann, D. Hayoz, and M. Depairon. Physical activity monitoring based on accelerometry: validation and comparison with video observation. *Medical & Biological Engineering & Computing*, 37(3):304–308, may 1999.

[30] K. Rauen, J. Schaffrath, C. Pradhan, R. Schniepp, and K. Jahn. Accelerometric Trunk Sensors to Detect Changes of Body Positions in Immobile Patients. *Sensors*, 18(10):3272, sep 2018.

[31] B. Najafi, K. Aminian, A. Paraschiv-Ionescu, F. Loew, C. Bula, P. Robert, C. J. Büla, P. Robert, S. Member, K. Aminian, A. Paraschiv-Ionescu, P. Robert, C. Bula, P. Robert, C. J. Büla, P. Robert, S. Member, K. Aminian, A. Paraschiv-Ionescu, and P. Robert. Ambulatory system for human motion analysis using a kinematic sensor: monitoring of daily physical activity in the elderly. *IEEE Transactions on Biomedical Engineering*, 50(6):711–723, jun 2003.

[32] A. Godfrey, A. K. Bourke, G. M. Ólaighin, P. Van De Ven, J. Nelson, G. M. Olaighin, P. Van De Ven, and J. Nelson. Activity classification using a single chest mounted tri-axial accelerometer. *Medical Engineering and Physics*, 33(9):1127–1135, nov 2011.

[33] M. Nazarahari and H. Rouhani. Detection of daily postures and walking modalities using a single chest-mounted tri-axial accelerometer. *Medical Engineering & Physics*, 57:75–81, jul 2018.

[34] J. Liu, J. Chen, H. Jiang, W. Jia, Q. Lin, Z. Wang, and Ieee. Activity Recognition in Wearable ECG Monitoring Aided by Accelerometer Data. In *2018 Ieee International Symposium on Circuits and Systems*, IEEE International Symposium on Circuits and Systems, pages 1–4. 2018.

[35] A.-y. Jeon, S.-y. Ye, J.-m. Park, K.-n. Kim, J.-h. J.-h. Kim, D.-k. Jung, G.-r. G. Jeon, J.-h. Ro, S.-y. Ye, and J.-h. J.-h. Kim. Emergency Detection System Using PDA Based on Self-Response Algorithm. *2007 International Conference on Convergence Information Technology (ICCIT 2007)*, pages 1207–1212, nov 2007.

[36] K. Altun, B. Barshan, O. Tunçel, O. Tuncel, O. Tunc -El, O. Tunçel, O. Tuncel, O. Tunc -El, O. Tunçel, O. Tunc -El, and O. Tuncel. Comparative study on classifying human activities with miniature inertial and magnetic sensors. *Pattern Recognition*, 43(10):3605–3620, oct 2010.

[37] J. J. Guiry, P. Van De Ven, J. Nelson, L. Warmerdam, and H. Riper. Activity recognition with smartphone support. *Medical Engineering & Physics*, 36(6):670–675, jun 2014.

[38] L. Breiman. Bagging Predictors. 140:123–140, 1996.

[39] M. Arif and A. Kattan. Physical Activities Monitoring Using Wearable Acceleration Sensors Attached to the Body. *PLOS ONE*, 10(7):e0130851, jul 2015.

[40] J. J. Rodrígues, L. I. Kuncheva, and C. J. Alonso. Rotation Forest : A New Classifier Ensemble Method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1619–1630, 2006.

[41] A. Reiss and D. Stricker. Introducing a Modular Activity Monitoring System. In *IEEE Engineering in Medicine and Biology Society*, pages 5621–5624. IEEE, aug 2011.

[42] J. Parera, C. Angulo, A. Rodríguez-Molinero, J. Cabestany, A. Rodriguez-Molinero, and J. Cabestany. User Daily Activity Classification from Accelerometry Using Feature Selection and SVM. In J. Cabestany, A. Prieto, F. Sandoval, and J. M. Corchado, editors, *Bio-Inspired Systems: Computational and Ambient Intelligence, Pt 1*, volume 5517 of *Lecture Notes in Computer Science*, pages 1137–+. 2009.

[43] F. Attal, S. Mohammed, M. Dedabrishvili, F. Chamroukhi, L. Oukhellou, and Y. Amirat. Physical Human Activity Recognition Using Wearable Sensors. *Sensors*, 15(12):31314–31338, dec 2015.

[44] K. Kira and L. A. Rendell. A Practical Approach to Feature Selection. In *Proceedings of the Ninth International Workshop on Machine Learning*, ML92, pages 249–256, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.

[45] A. Sharma, H. J. Lee, and W.-Y. Chung. Principal Component analysis based Ambulatory monitoring of elderly. *Journal of the Korea Institute Of Information and Communication Engineering*, 12(11):2105–2110, 2008.

[46] A. Sharma, Y.-D. Lee, and W.-Y. Chung. High Accuracy Human Activity Monitoring Using Neural Network. In *2008 Third International Conference on Convergence and Hybrid Information Technology*, Third 2008 International Conference on Convergence and Hybrid Information Technology, Vol 1, Proceedings, pages 430–435. IEEE, nov 2008.

[47] A. Li, L. Ji, S. Wang, and J. Wu. Physical activity classification using a single triaxial accelerometer based on HMM. In *IET International Conference on Wireless Sensor Network 2010 (IET-WSN 2010)*, number 3, pages 155–160. IET, 2010.

[48] S. Kozina, H. Gjoreski, M. M. Gams, M. Lustrek, and M. Luštrek. Three-layer Activity Recognition Combining Domain Knowledge and Meta-classification. *Journal of Medical and Biological Engineering*, 33(4):406–414, 2012.

[49] A. M. Khan, Young-Koo Lee, S. Y. Lee, Tae-Seong Kim, Y.-K. Lee, S. Y. Lee, and T.-S. Kim. A Triaxial Accelerometer-Based Physical-Activity Recognition via Augmented-Signal Features and a Hierarchical Recognizer. *IEEE Transactions on Information Technology in Biomedicine*, 14(5):1166–1172, sep 2010.

[50] A. M. Khan, Y.-K. Lee, S. Lee, and T.-S. Kim. Accelerometer's position independent physical activity recognition system for long-term activity monitoring in the elderly. *Medical & Biological Engineering & Computing*, 48(12):1271–1279, dec 2010.

[51] T. Hur, J. Bang, T. Huynh-the, J. Lee, J.-i. Kim, and S. Lee. Iss2Image: A Novel Signal-Encoding Technique for CNN-Based Human Activity Recognition. 2018.

[52] S. Ha, J. M. Yun, and S. Choi. Multi-modal Convolutional Neural Networks for Activity Recognition. *Proceedings - 2015 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2015*, pages 3017–3022, 2016.

[53] B. Zhou, J. Yang, and Q. Li. Smartphone-based activity recognition for indoor localization using a convolutional neural network. *Sensors (Switzerland)*, 19(3):1–15, 2019.

[54] A. Ignatov. Real-time human activity recognition from accelerometer data using Convolutional Neural Networks. *Applied Soft Computing Journal*, 62:915–922, 2018.

[55] H. Cho and S. M. Yoon. Divide and conquer-based 1D CNN human activity recognition using test data sharpening. *Sensors (Switzerland)*, 18(4):1–24, 2018.

[56] C. Avilés-Cruz, A. Ferreyra-Ramírez, and A. Zúñiga-López. Coarse-Fine Convolutional Deep-Learning Strategy for Human Activity Recognition. 2019.

[57] Y. Chen, K. Zhong, J. Zhang, Q. Sun, and X. Zhao. LSTM Networks for Mobile Human Activity Recognition. (Icaita):50–53, 2016.

[58] F. J. Ordóñez and D. Roggen. Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition. *Sensors (Switzerland)*, 16(1), 2016.

[59] F. Li, K. Shirahama, M. A. Nisar, L. Köping, and M. Grzegorzek. Comparison of feature learning methods for human activity recognition using wearable sensors. *Sensors (Switzerland)*, 18(2):1–22, 2018.

[60] V. N. Nguyen and H. Yu. Novel Automatic Posture Detection for In-patient Care Using IMU Sensors. In *PROCEEDINGS OF THE 2013 6TH IEEE CONFERENCE ON ROBOTICS, AUTOMATION AND MECHATRONICS (RAM)*, Proceedings of the 2013 6th Ieee Conference on Robotics, Automation and Mechatronics, pages 31–36. IEEE; IEEE Syst, Man & Cybernet Singapore Chapter; IEEE Robot & Automat Singapore Chapter; IEEE Singapore Sect; Natl Grid Corp Philippines; Tourism Promot Board Philippines, 2013.

[61] O. Banos, J. M. Galvez, M. Damas, H. Pomares, and I. Rojas. Window size impact in human activity recognition. *Sensors (Switzerland)*, 14(4):6474–6499, 2014.

[62] G. Weiss, K. McCarthy, and B. Zabar. Cost-sensitive learning vs. sampling: Which is best for handling unbalanced classes with unequal error costs? *Dmin*, pages 1–7, 2007.

[63] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16(1):321–357, 2002.

[64] Y. Chen and Z. Wang. A hierarchical method for human concurrent activity recognition using miniature inertial sensors. *Sensor Review*, 37(1):101–109, jan 2017.

[65] H. Zhang. The Optimality of Naive Bayes Naive Bayes and Augmented Naive Bayes. Technical report, 2004.

[66] Xiuxin Yang, A. Dinh, and Li Chen. Implementation of a wearerable real-time system for physical activity recognition based on Naive Bayes classifier. In *2010 International Conference on Bioinformatics and Biomedical Technology*, pages 101–105. IEEE, 2010.

[67] L. Breiman. Random Forests. Technical report, 2001.

[68] Ç. B. Erdaş, I. Atasoy, K. Açıcı, and H. Oğul. Integrating features for accelerometer-based activity recognition. *Procedia Computer Science*, 98:522–527, 2016.

[69] N. Rusk. Deep learning. *Nature Methods*, 13(1):35, 2015.

[70] Y. Jang, S. Kim, K. Kim, and D. Lee. Deep learning-based classification with improved time resolution for physical activities of children. *PeerJ*, 6:e5764, 2018.

[71] A. Soro, G. Brunner, S. Tanner, and R. Wattenhofer. Recognition and repetition counting for complex physical exercises with deep learning. *Sensors (Switzerland)*, 19(3), 2019.

[72] B. Almaslukh, A. M. Artoli, and J. Al-Muhtadi. A Robust Deep Learning Approach for Position-Independent Smartphone-Based Human Activity Recognition. 2018.

[73] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representation by back-propagating errors. *Nature*, 323(9):533–536, 1986.

[74] Y. LeCun and Y. Bengio. Convolutional networks for images, speech, and time series. *Handb. Brain Theory Neural Netw.*, 3361, 1995.

[75] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

[76] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. pages 1–15, 2014.

[77] S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. 2015.

[78] G. Lemaitre, F. Nogueira, and C. K. Aridas. Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017.

[79] H. Li and M. Trocan. Personal health indicators by deep learning of smart phone sensor data. *2017 3rd IEEE International Conference on Cybernetics, CYBCONF 2017 - Proceedings*, pages 1–5, 2017.

[80] F. Massé, A. K. Bourke, J. Chardonnens, A. Paraschiv-Ionescu, K. Aminian, F. Masse, A. K. Bourke, J. Chardonnens, A. Paraschiv-Ionescu, and K. Aminian. Suitability of commercial barometric pressure sensors to distinguish sitting and standing activities for wearable monitoring. *Medical Engineering & Physics*, 36(6):739–744, jun 2014.