UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Hristijan Sardjoski

Activity recognition using accelerometers

Master's Thesis (30 ECTS)

Supervisor: Meelis Kull, PhD

Tartu 2019

Activity recognition using accelerometers

Abstract: Activity recognition is considered to have a wide range of applications, especially in the health sector. The assessment of different activities of daily living is useful because it can highlight information related to a specific health condition such as obesity, overweight, stroke, or fall. Moreover, the prevalence of different user-friendly wearable devices enables collecting tri-axial accelerometer data in a non-intrusive and discrete manner.

The accelerometer data used for activity recognition in this thesis is provided by SPHERE [1]. The accelerometer readings are recorded from four wearables attached on a single person's hands and legs.

This thesis compares the capabilities for activity recognition of the random forest model and the long short-term memory neural network to discern among 9 in-door activities including *brushing teeth*, *eating a meal*, *flossing*, *getting dressed/undressed*, *mixing (food)*, *spreading (food)*, *walking*, *washing hands*, *writing*. In addition, the list of activities is extended with an *unknown* activity. Greater focus is given on the following topics: feature extraction, segmentation of the time-series accelerometer data, parameter and hyper-parameter tuning, model training, model evaluation and generalization capability. The results suggest that the random forest model using the accelerometer-based extracted features slightly outperforms the long short-term memory neural network using raw accelerometer data when the activity recognition task is limited on the 9 chosen activities, and, additionally, when the *unknown* activity is included.

Keywords: activity recognition, accelerometer, feature engineering, time series segmentation, machine learning, neural networks

CERCS: P176 - Artificial intelligence; P170 - Computer science, numerical analysis, systems, control

Tegevuse tuvastamine kiirendusandurite abil

Lühikokkuvõte: Automaatsel tegevuse tuvastamisel on palju rakendusi, iseäranis tervise valdkonnas. Erinevate igapäevaeluliste tegevuste mõõtmine on kasulik, sest see võimaldab saada teavet terviseseisundite kohta, nagu näiteks ülekaalulisus, insult või kukkumine. Veelgi enam, erinevate kasutajasõbralike kantavate seadmete laialdane levik võimaldab koguda kolmeteljelise kiirendusanduri andmeid mittesegavalt ja diskreetselt.

Käesolevas töös on tegevuse tuvastamiseks kasutatud kiirendusanduri andmed on pärit projektist SPHERE [1]. Kiirendusmõõtmised on tehtud nelja kantava seadmega, mis olid kinnitatud katseisiku randmetele ning jalgadele.

Töö võrdleb otsustusmetsa (random forest) ning pika lühiajalise mäluga (LSTM) tehisnärvivõrkude võimet tuvastada 9 siseruumi tegevust: *hambapesu*, *söömine*, *hambaniiditamine*, *riietumine/lahtiriietumine*, *(toidu) segamine*, *(toidu) pealemäärimine*, *kõndimine*, *käte pesemine*, *kirjutamine*. Lisaks laiendatakse tegevuste hulka *teadmata* tegevusega. Suuremat tähelepanu pööratakse järgmistele teemadele: tunnuste eraldamine, kiirendusanduri aegrea tükeldamine, parameetrite ja hüperparameetrite häälestamine, mudeli treenimine, mudeli hindamine ning üldistusvõime. Tulemused näitavad, et kiirendusanduril põhinevaid ekstraheeritud tunnuseid kasutav otsustusmets ületab tuvastusvõimelt pisut kiirendusanduri mõõtetulemusi muutmata kujul kasutavat LSTM-võrku, ning seda nii 9 tegevuse tuvastamisel kui ka peale teadmata tegevuse lisamist.

Võtmesõnad: tegevuse tuvastamine, kiirendusandur, tunnuste töötlemine, aegrea tükeldamine, masinõpe, tehisnärvivõrgud

CERCS: P176 - Tehisintellekt; P170 - Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Contents

1 Introduction

The extraction of knowledge from the accelerometer data has proven useful in fields such as health monitoring [2], and fitness tracking [3]. The detection of the activity patterns is used to recognize the activities performed by a person, thus, providing informative feedback. For example, different activities of daily living (ADL) such as *food preparation*, *cleaning*, *transition(walking) between rooms*, *door interactions*, *washing dishes*, *drinking beverage*, *vacuuming*, *brushing teeth* provide information about the person's everyday habits. The feedback's activity assessment based on the repetitive activities highlights the person's current health and well-being [4]. More specifically, the recognized activities can be further utilized to improve the predictions on different medical conditions such as obesity, depression, stroke and falls [5] [6].

The main goal of this thesis is to provide machine learning models which are capable of performing activity recognition using the accelerometer data provided by SPHERE [1]. The data is obtained using tri-axial accelerometers worn by a single person on both of his arms and legs.

SPHERE represents an interdisciplinary research collaboration including engineers, social care professionals, clinicians, and others [6]. The SPHERE's main goal is to provide answers for long term health conditions. The undertaken approach includes data analysis over the output of a platform of networked sensors such as accelerometers, gyroscopes, RGB-cameras, environmental sensors placed in a home environment [6].

The tri-axial accelerometer sensors [7][8] are measuring the acceleration along the X, Y, and Z directions. The provided acceleration measurements from the 3D space can be used to recognize different movement patterns of the person who is wearing the device. Different types of wearable sensors beside the tri-axial accelerometer can be used for the HAR task, such as gyroscopes, magnetometers and mobile phone accelerometers [8][9]. Nevertheless, the tri-axial accelerometers are most widely accepted for collecting acceleration measurements due to their low-cost and low-power requirements [7][9].

Based on the available annotated data from SPHERE the following 9 in-door activities have been chosen: *brushing teeth*, *eating a meal*, *flossing*, *getting dressed/undressed*, *mixing (food)*, *spreading (food)*, *walking*, *washing hands*, *writing*. Two different human activity recognition (HAR) tasks are defined using the aforementioned activities. The first task is to provide suitable models for recognition between the 9 different activities. The second task is to provide models for activity recognition when additionally an *unknown* activity is included. The random forest model as part of a traditional machine learning approach, and the LSTM neural network are selected for addressing the defined HAR tasks, thus, different challenges arise.

The non-linear and time-dependent nature of the accelerometer data highlights the need of data pre-processing and feature extraction [8] when a classical machine-learning model is used. The first challenge addressed in this thesis is focused on alleviating the model's limitations to capture the complex dependencies present in the data. Furthermore, the solution for the aforementioned challenge is consisted of data pre-processing, time series segmentation, extensive feature extraction, choosing the appropriate model parameters and defining proper evaluation measure.

The imposed limitations from which the classical methods suffer are mostly alleviated due to the theoretical and practical advancements in the area of artificial neural networks. The non-linear and time-dependent nature of the accelerometer data can be addressed using a recurrent neural network (RNN) [10]. However, a significant drawback of the vanilla RNN implementation is its inability to process time series data given larger number of time steps [11]. In order to overcome this limitation, long short-term memory recurrent neural network (LSTM) is used due to its advanced weight updating operation which includes a multi-gated memory cell [12]. By using the LSTM, the challenges that prevail from the raw signal input are partially mitigated. However, additional challenges arise due to the numerous possible architecture configurations followed by extensive, time-consuming and power-consuming hyper-parameter tuning and data pre-processing as a prerequisite in order to define the specific input for the LSTM network.

The rest of the thesis is structured as follows. In Section 2 background information for the topic is presented. In Section 3 related work from the field is given along with the most suitable practices for human activity recognition. In Section 4 a thorough overview of the methodology used in the thesis is given. Section 5 gives an overview of the results obtained by using the traditional machine learning approach and the LSTM. In Section 6 the main conclusions are outlined and the impact of the thesis is outlined, and in Section 7 an overview of the future work based on the current implementation and results is given while also highlighting drawbacks and the potential improvements that can be performed.

2 Background

This section highlights the necessary background information, such as the nature of the accelerometer sensors that were used for collecting the data, structure of the data provided by SPHERE, used traditional machine learning models and used LSTM neural networks.

2.1 Accelerometer sensors

In general, there are two types of accelerometers, analog and digital [13][14][15]. Regardless of the type, an accelerometer sensor measures acceleration indirectly through a force that is applied on it. This force is usually caused by the acceleration.

The accelerometer readings for the purposes of this thesis are produced by tri-axial accelerometers [7] which measure the force of the acceleration in three different directions, x, y and z. Each accelerometer is also called a wearable device, even though the term wearable device is not limited only for accelerometers. Wearable accelerometers, regardless of the specific configuration, are resource-constrained because they must be lightweight and small in order to be acceptable [16]. Due to the physical requirements, challenges such as smaller batteries with low capacity, wireless performances and mobility have to be addressed. In addition, beside meeting these technical requirements the accelerometers must also be comfortable for wearing [16]. Since people are used to wrist-worn gadgets such as watches, the most acceptable location to attach an accelerometer sensor on a person's body is the wrist, resulting in low invasion in the everyday life of the person who wears it.

In details, the accelerometers measure the force of the acceleration [7]. The force vector R is composed of the projections R_x, R_y and R_z as shown in Figure 1 with respect to the wearable's X, Y and Z axes. These projections are considered to be linearly related to the values that the accelerometer will output. Due to the analog nature of the accelerometers in SPHERE, first the analog output needs to be converted to digital (ADC) output using the analog to digital converter which has a specific range of values. The range of the converter is predefined by the specification of the used accelerometer. In addition, in order to get voltage units from the ADC output, a reference voltage depending on the specification of the accelerometer is applied. Furthermore, the final output in g units is provided by using the accelerometer-specific zero-g voltage. The g-force is a measurement that represents the acceleration per unit mass. A g-force of $1g$ unit equals the Earth's gravitational acceleration ($1g = 9.81\frac{m}{s^2}$). Each configuration has its unique zero-g voltage that corresponds to $0g$. The zero-g level is used to determine the signed voltage value on the reference voltage applied to the ADC output. In the end, the value of the modified output is translated from voltage units to g units by using the sensitivity-specific values of the accelerometer.

The wearable used in SPHERE is called SPW-2 and it is presented in the studies by

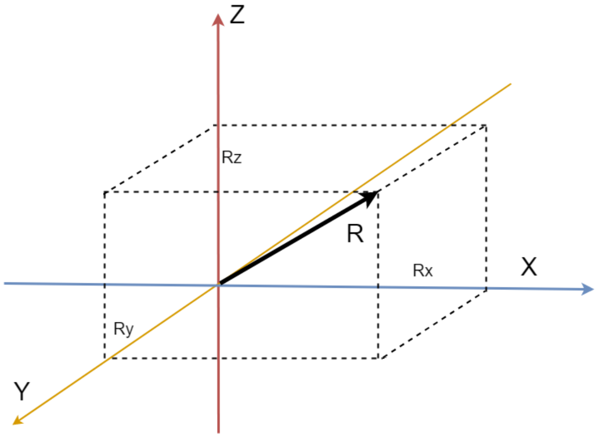


Figure 1. Accelerometer X, Y and Z axes. The force vector R is composed of the projections R_x, R_y and R_z with respect to X, Y and Z axes.

Atis Elsts et al. [1] and Xenofon Fafoutis et al. [16]. Each SPW-2 wearable consists of two ADXL362 accelerometers [15] and one LSM6DS0 gyroscope [17]. For the purposes of this thesis only the measurements coming from the ADXL362 accelerometers are taken into consideration. The sampling rate of the used acceleormeters is $24Hz$. The full implementation and specification of the accelerometers used in SPHERE is not the main scope of this thesis, therefore, further information about the advantages and specification of the SPW-2 wearable is presented in the study by Xenofon Fafoutis et al. [16].

As an example, the accelerometer readings with a length of 12 seconds for the activity *writing* are presented in Figure 2. Due to the sampling rate of $24Hz$, there are 288 consecutive readings for each axis. The visual inspection of the provided readings in Figure 2 points out the numerous spikes present in each of the axis. Additionally, the interlacing between the readings of the X and Z axes is clearly visible, and is almost always above 0, while the readings for the Y axis are always below 0. In addition, in Figure 3 another time window representing the activity *writing* is presented. When inspecting the time window in Figure 3 it can be concluded that similar patterns emerge regardless of the time difference between the first and second time window of approximately 1 minute. By applying further inspection on the data many other similarities may emerge between the provided time windows.

The recorded accelerometer readings of the right hand movement during the activity of *writing* in Figure 2 suggest that there is no evident acceleration in each of the axes that can be singled out as the most dominant. Additionally, from the readings for each axis in Figure 2 it can be concluded that the accelerometer is rotated in a way that neither axis is facing explicitly upwards or downwards. The distribution of the readings in all of the three axes is quite similar and ranges between $-1g$ and $1g$. In conclusion, the person's

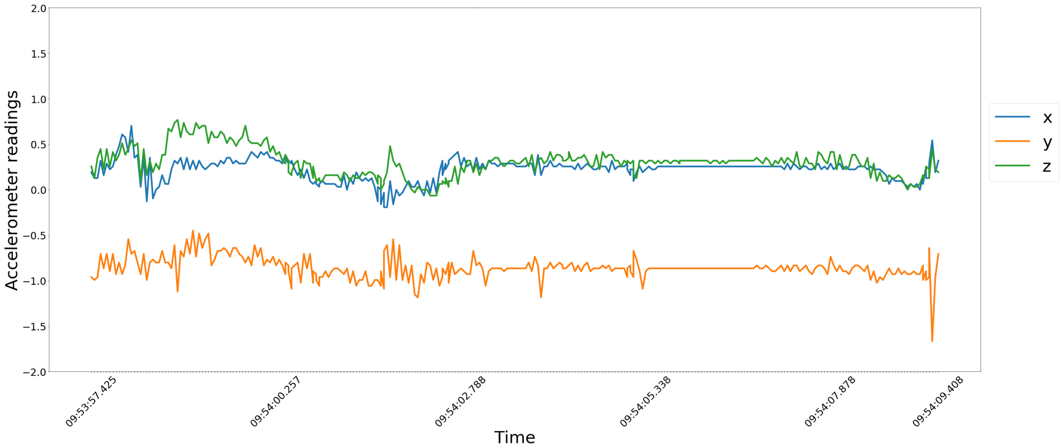


Figure 2. A time window representing the activity *writing* with a length of 12 seconds. The sampling rate of the accelerometer is $24Hz$, thus, for 12 seconds there are 288 consecutive readings.

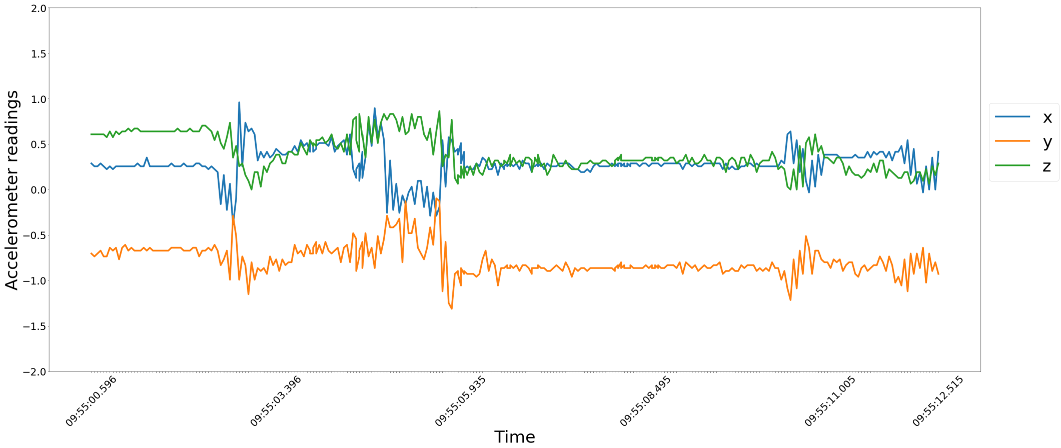


Figure 3. Different time window representing the activity *writing* with a length of 12 seconds.

hand during the writing motion slightly rotates with low acceleration, thus, the hand is most probably leaning on a hard surface (e.g. table).

The models do not try to recognise each accelerometer data sample, instead, they try to recognise a pattern present in the data over a time window. Therefore, instead of classifying every single reading from the raw accelerometer data, it is useful to extract features over a defined time window. For example, classifying every single reading with a decision tree model will significantly increase the width and depth of the tree, thus, representing each data point as a different feature. In addition, the decision tree model will have difficulties capturing the temporal dependence between the consecutive readings. The similarities that can emerge between the groups of consecutive readings pointing to a single activity as presented in Figures 2 and 3 can be used as part of the process called feature extraction(also referred as feature engineering). The feature extraction process enables inferring additional knowledge from the data that can help the models when a discrimination between more activities is performed. The extraction of features is useful because it decreases the computational load of the models and reduces the noise effects. Additionally, the temporal dependence is partially alleviated because the features are computed over time windows.

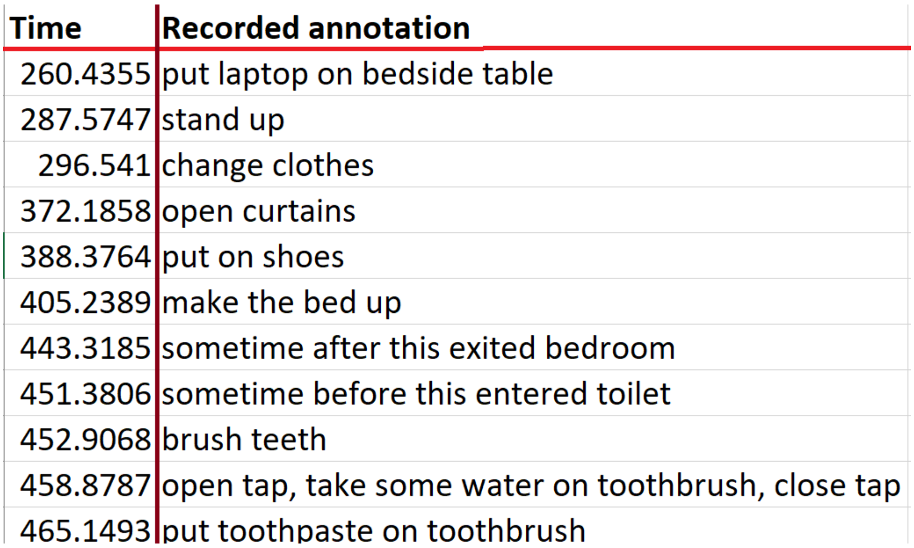
2.2 SPHERE one-month dataset

The SPHERE one-month dataset consists of records from a house equipped with several types of sensors for a time period of one month. The SPHERE smart-house includes PIR environmental sensors, RGB cameras, motion sensors, accelerometers, gyroscopes and voice recorders. Except the accelerometers, all other sensors are out of the scope of this thesis. Only a small portion of the data is annotated (manually labelled with the activity that occurs at a given time). [1]

The accelerometers output the data in packages. Further information about the package format is given in Appendix. Beside the accelerometer records, there are annotation files carrying activity information for specific segments of the accelerometer data. A small subset of annotations from a single annotation file is shown in Figure 4. The first column in Figure 4 represents the time relative to the beginning of the annotation episode. The annotations are self-reported by the participant using an attached audio recorder, and later transcribed. The second column in Figure 4 represents the annotations provided by the participant in the SPHERE house. Each of the annotation files has consecutive annotations covering one segment of the accelerometer data with a duration between 30 minutes to 2 hours.

The annotated segments expand over 7 consecutive days. Additionally, there are approximately 2 to 4 hours of annotated accelerometer readings for each day. Even though the accelerometer data readings for one month are of a large quantity, when the annotated segments are added together, only 24 hours of the one-month data is annotated.

The provided annotations have limitations that arise from the level of granularity. A considerable amount of annotations such as *put laptop on table*, *pick up laptop from table*, *take plate from tabletop*, *put plate down on tabletop*, are very specific, thus, they point to a single atomic activity. On the contrary, there are annotations that are general such as *change clothes*, *prepare a meal*, *prepare a drink*, *tidy up*. Such annotations are with a low level of detail, thus, they usually encapsulate more than one atomic activity. For example, the activity represented by the annotation *prepare a meal* can be consisted of many atomic activities including *take plate from tabletop*, *take milk from fridge*, *take ham from fridge*, *take cheese from fridge*, *put ham down on tabletop*, etc. Given the



Time	Recorded annotation
260.4355	put laptop on bedside table
287.5747	stand up
296.541	change clothes
372.1858	open curtains
388.3764	put on shoes
405.2389	make the bed up
443.3185	sometime after this exited bedroom
451.3806	sometime before this entered toilet
452.9068	brush teeth
458.8787	open tap, take some water on toothbrush, close tap
465.1493	put toothpaste on toothbrush

Figure 4. Initial set of annotations with relative timing. The first column represents the relative time of each annotation. The second column represents the recorded annotation provided by the participant who resides in the SPHERE's house.

previous, the annotations are not in a format suitable for practical use as target variables, thus, adaptation of the annotations is necessary.

2.3 Supervised and unsupervised machine learning

In general, the main focus of machine learning is to provide the appropriate models by selecting the right features in order to address specific tasks. The following section is mainly based on the book "Machine learning: the art and science of algorithms that make sense of data" by Peter Flach [18] and encapsulates two different machine learning approaches, supervised and unsupervised.

Supervised learning is a type of learning where the training data is consisted of features and targets (labels) and the models try to find a relation between the features in the data and the targets. The task of the supervised learning is to predict the targets on test data by using the discovered relations from the learning process [18]. There are many different supervised machine learning methods such as linear regression, random forest, artificial neural network, etc [18].

On the contrary, unsupervised learning is a learning process when the data is not labeled(does not have target values) [18]. The goal here is to learn the structure from the data and it usually has different tasks such as clustering, anomaly detection, topic discovery, etc. Additionally, there are different unsupervised machine learning methods such as PCA (principal component analysis), hierarchical clustering, k-means clustering, etc [18].

One of the main goals of this thesis is delivering models that are able to discriminate between given set of activities by using the X, Y and Z accelerometer data records. Therefore, two different supervised machine learning models including random forest and long short-term memory neural network are trained. Additionally, the feature extraction process is considered when building the random forest model in order to increase the model's discriminatory power. The feature extraction techniques included in this thesis are based on supervised and unsupervised machine learning models, statistical analysis on the time series data and time series transformations in frequency domain using Fast Fourier transform. In the following sections more details regarding the machine learning models along with the Fourier transform method used in this thesis are presented.

2.4 Feature extraction techniques

This section highlights the used supervised and unsupervised machine learning models as feature extraction techniques. Additionally, a greater focus is given on the Fast Fourier transform.

2.4.1 Simple linear regression

The simple linear regression represents a linear regression model with only one explanatory variable. It is a statistical method for defining a function approximator for the dependent variable y based on the single independent variable x as shown in Figure 5. A short description of the linear regression method is presented following the book "Methods of multivariate analysis" by Alvin C. Rencher [19], along with a practical explanation of why this method is chosen.

The independent variable x in Figure 5 is used to approximate the values of the target variable y, thus, the distance between the y approximations and the fitting line represents the prediction error (or residual error), respectively. The simple linear regression method is given with the following expression:

$$y = \alpha + \beta x + \epsilon \tag{1}$$

where α and β are the model parameters, ϵ is the error term and x represents the independent variable. The objective (loss) function of the linear regression is to minimize the error term by using the method of least squares by estimating the intercept α and weight coefficient β.

This method is utilized as a feature extraction method due to its ability to find the best fitting line when only two variables are given (consecutive time points and accelerometer readings per axis) and provide the regression line which best corresponds to the values of each of the three accelerometer axes. Because the regression line depends on the x-axis

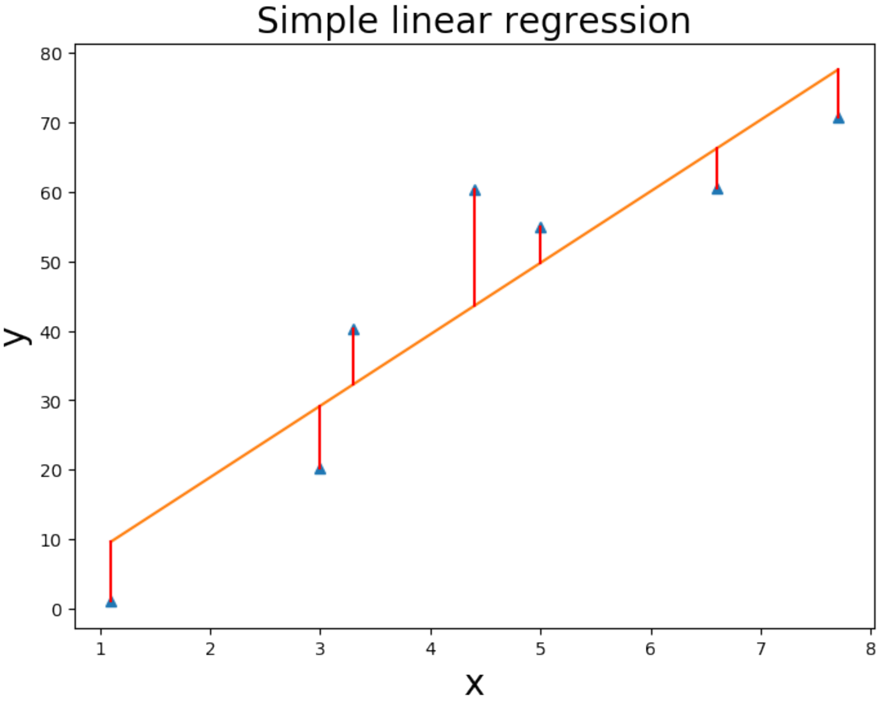


Figure 5. The simple linear regression's best fitting line for the relationship between the independent variable x and the target variable y. The triangles represent the approximations when the linear regression is used, while the offset from the best fitting line represents the residuals.

scale, proper min-max normalization of the values is performed, thus they are in the range between 0 and 1. The goal is to define the slope of the regression line.

The three sub-figures in Figure 6 represent the regression lines along with their corresponding intercepts after applying the linear regression for the X, Y and Z axes respectively. Furthermore, in each of the three sub-figures in Figure 6 the slope of the regression line is presented. For each activity a small range of possible values is expected to be formed with respect to the axes.

2.4.2 Principal component analysis

Another machine learning method used for feature extraction purposes is the principal component analysis (PCA). A description of the principal component analysis is given following the book "Methods of multivariate analysis" by Alvin C. Rencher [19].

PCA represents a procedure for transformation of the original variables and their corresponding observations present in the data into linearly uncorrelated variables called principal components. PCA's goal is the maximization of the variance given the linear

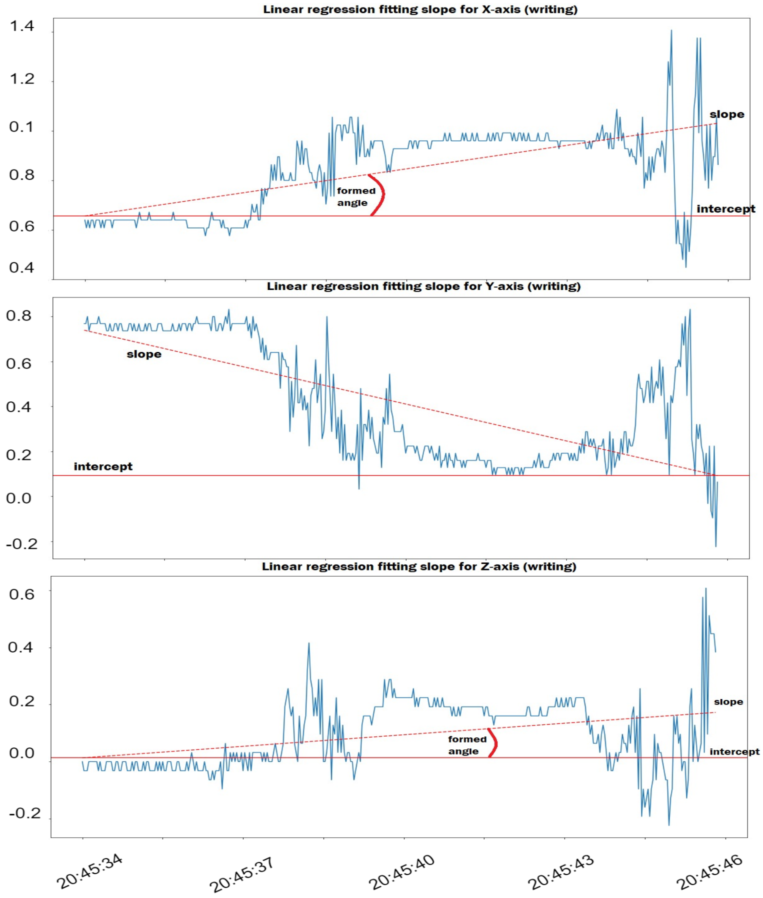


Figure 6. The defined slopes of the regression lines after the simple linear regression is applied over the accelerometer readings for each axis.

combination of the features from the data.

The principal components represent the underlying structure in the data. The principal component analysis derives projections based on the features where each projection is orthogonal to all others following a variance decreasing order as shown in Figure 7. The number of principal components is equal to the number of the original features. The first principal component in Figure 7, noted as PC1 has the highest variance as the original data is most spread following this projection. Additionally, the second principal component in Figure 7, noted as PC2, is orthogonal to the first component, and it carries less variance compared to PC1.

PCA can be used either as a dimensionality-reduction technique by reducing the number of features with high dependence or as a feature extracting technique. The PCA in this thesis is used as feature extracting technique. The reason why PCA is incorporated as a feature extraction technique is because of the correlation between

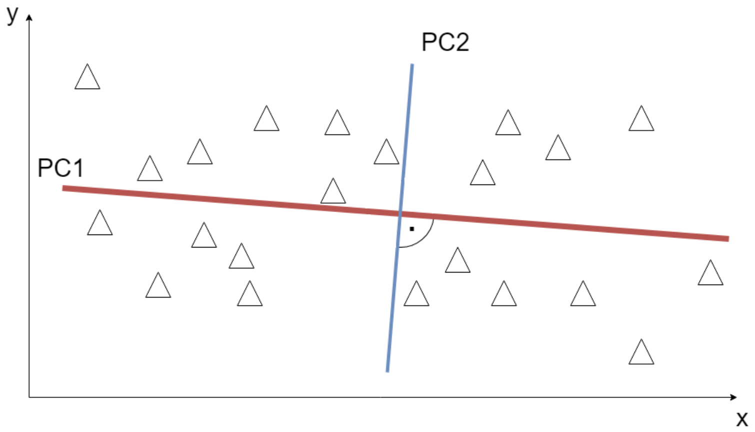


Figure 7. Principal component analysis on a data with two features. The triangles represent the original data. The red line, noted as PC1 represents the first principal component, and carries the highest variance. The blue line, noted as PC2 represents the second principal component, which is orthogonal to PC1, while carrying less variance.

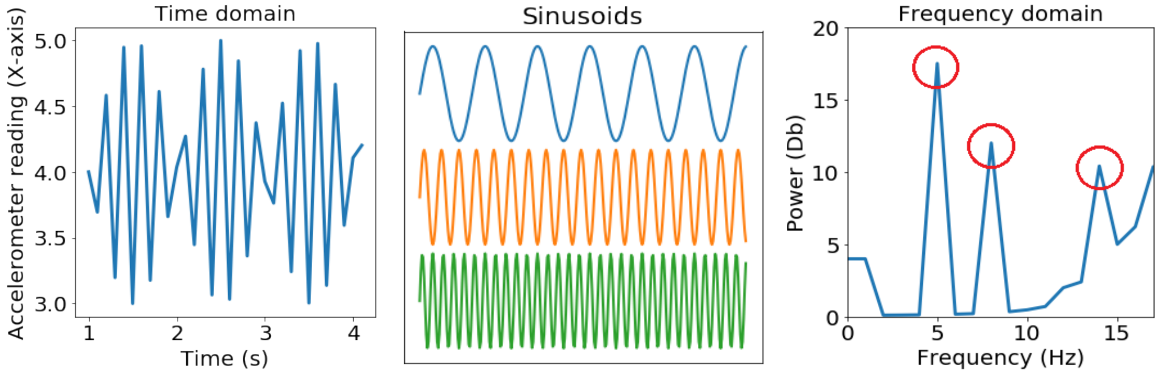
the accelerometer data readings for the X, Y and Z axes. The idea is to provide the principal component variances of the first two PCs when the data is projected in the sub-domain space. The expectation is that the first two principal components will carry similar variance respectively given the same activity throughout different time periods. For example, the variance in the first two PCs for the activity *writing* has similar values throughout different periods in time, and when compared to the activity *walking* there is a noticeable difference as shown in Table 1.

Table 1. The variance values of the first two principal components for 5 different time windows for the activities *writing* and *walking*.

	writing		***walking***	
Time windows	PC1	PC2	PC1	PC2
1	0.68	0.22	0.73	0.16
2	0.63	0.29	0.76	0.13
3	0.64	0.24	0.73	0.18
4	0.66	0.23	0.73	0.15
5	0.67	0.23	0.72	0.18

2.4.3 Fast Fourier transform

Furthermore, for the purpose of this thesis Fast Fourier transform is used to capture the rhythmicity of the accelerometer data. The Fourier transform provides an algorithm



(a) (b) (c)

Figure 8. (a)The $F(t)$ recorded signal in time domain. (b) The $F(t)$ recorded signal disolved in many different sinusoids. (c). The power spectrum of the $F(t)$ recorded signal in frequency domain.

for transforming the time domain sampled data $F(t)$ in frequency domain $\Phi(v)$. The explanation follows the book "A student's guide to Fourier transforms with applications in physics and engineering" by John Francis et al. [20].

The basic idea is to represent the time recorded signal $F(t)$ as a sum of sinusoids. In Figure 8, the $F(t)$ signal is presented in (a). Next, the middle sub-figure (b) in Figure 8 represents the different sinusoids. The sinusoids are separated oscillations at different frequencies. The extraction of various frequencies and amplitudes from the recorded signal is called Fourier analysis. In Figure 8 the right-most sub-figure (c) represents the $F(t)$ signal in frequency domain. The sinusoids that better match the recorded signal have the largest power. In addition, regardless of the periodicity of $F(t)$ a full description will always include the signal's sines and cosines. The following expressions are considered as a formal statement of the Fourier transform and are called a 'Fourier pair':

$$\Phi(v) = \int_{-\infty}^{\infty} F(t)e^{2\pi ivt}dt, \tag{2}$$

$$F(t) = \int_{-\infty}^{\infty} \Phi(v)e^{-2\pi ivt}dv \tag{3}$$

where $\Phi(v)$ is the Fourier transform of $F(t)$, but also, $F(t)$ is the Fourier transform of $\Phi(v)$. All of the sinusoids with better match to $F(t)$ will have larger coefficient, and thus, larger power spectrum.

As an example, in Figure 9 the first row represents the activity *walking* in a time window of approximately 5 seconds, and in a frequency window of $5Hz$. Furthermore, the second row represents different 5 seconds of the same activity, recorded few days later. In both of these rows, the left sub-figures in Figure 9 represent the accelerometer readings for the activity *walking* in the time domain. The right sub-figures represent the *walking* in the frequency domain, after the FFT is applied. The right sub-figures in Figure 9 in both rows point out that the largest power spectrums are near $1Hz$ and $4.5Hz$ respectively. The FFT offers another point of view of the same data, thus, enabling to extend the feature extraction process.

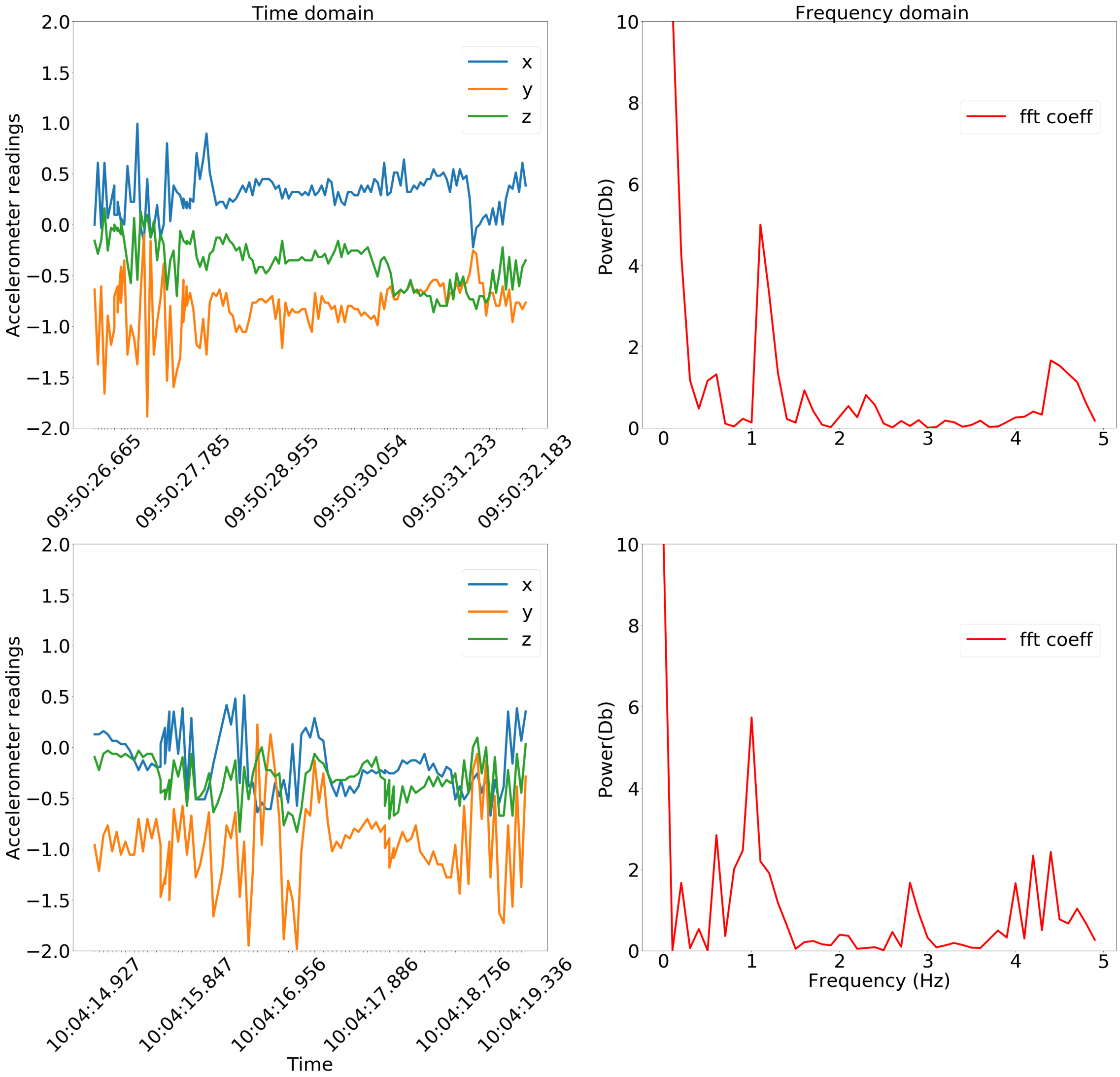


Figure 9. The first row represents the activity *walking* in both, time and frequency domain. The second row represent the same activity, recorded few days later. Both of the frequency sub-figures point out that the *walking* has it's power spectrums near $1Hz$ and $4.5Hz$ respectively.

2.5 Supervised machine learning models

This section provides greater details for the selected machine learning models used for recognition of activities.

2.5.1 Random forest

Random forest (RF) represents an ensemble learning method widely used for classification and regression problems. Its implementation includes multiple classification or regression decision trees, where each tree is built following a specific measure as a splitting criteria [21][22]. During the training process, each tree in the ensemble is built with a different subset of features from the training data. The training process of an RF includes bootstrapping of the original data and generating data sets with equal size. However, the data sets for each decision tree in the ensemble consist of different instances which is important in order to prevent overfitting. When the number of trees is high usually there is a correlation between the dominating features from each tree. In the case of a classification task, the RF outputs the mode of the classes predicted by each of the decision trees (majority voting) in order to produce a prediction. In the case of a regression task, the random forest outputs the mean of the outputs of each of the decision trees. The biggest advantage of the random forest method over the rest of the traditional methods is the ability to handle the curse of dimensionality.

The random forest algorithm combined with a suitable feature extraction process can provide a powerful discrimination model when the accelerometer data is the considered input, thus, it is part of this thesis.

2.5.2 Artificial neural networks

Artificial Neural Network (ANN) represents a collection of connected units, also called neurons, organized in layers where each unit carries a non-linear activation function as shown in Figure 10. By definition the activation function can take any number of input values, but it outputs only a single value. Due to the usage of nonlinear activation functions in each neuron, the ANNs are capable of capturing the non-linear dependencies present in the data. The connections between the neurons are called weights. The weight adjustment is done by minimization of a loss function. The pass of the information from the beginning to the end of the network is called a forward-pass, while the propagation of the errors is called a backward-pass. Overall, the network's learning process is simply weight re-adjustment after a single forward-pass and a backward-pass by minimizing the chosen loss function when presented with train target variables. The architecture of the network can vary due to the number of layers, the number of neurons in each of the layers, and different activation functions in the neurons (tanh, sigmoid, relu, etc.). [23]

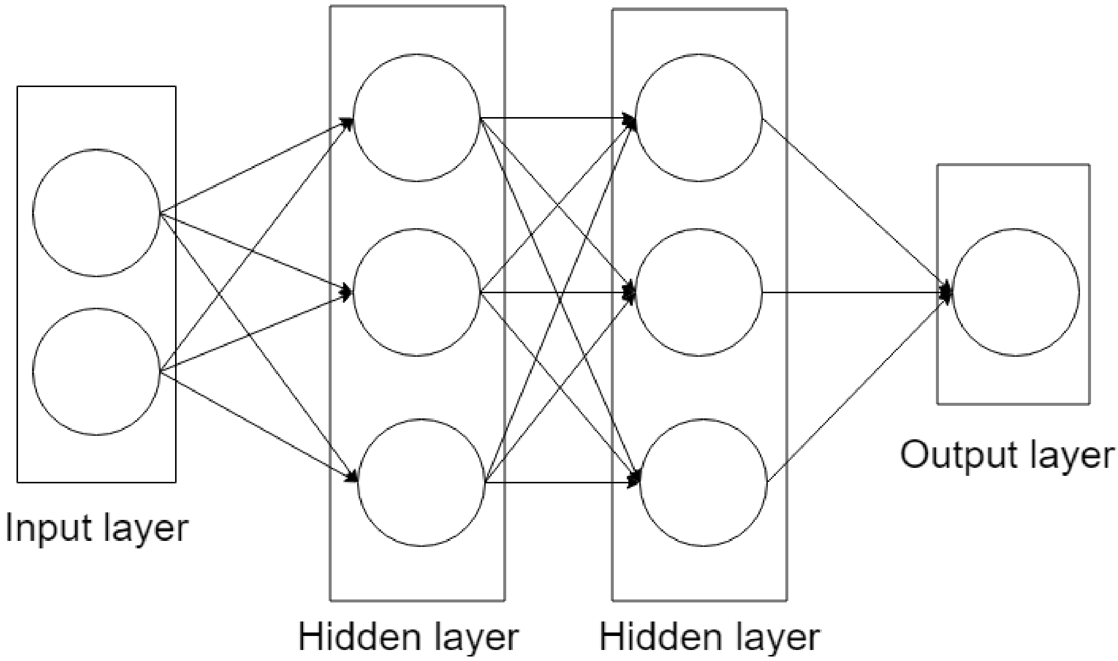


Figure 10. A sample architecture of a feed forward neural network with two neurons in the input layer, three neurons in each of the hidden layers, and one neuron in the output layer.

Feed-forward neural network. There are many different types of artificial neural networks. One of the most widely used types of neural networks for classification and function approximation is the feed-forward neural network. The information in this neural network goes only in one direction, from the input layer toward the output layer as presented in Figure 10, thus, not creating a directed cycle [8].

Reccurent neural network (RNN). The RNNs basic principals are the same as those of the feed-forward neural networks including a specific number of layers, specific number of neurons in each layer, selection of activation function, and initialization of a proper loss function [10]. However, the recurrent neural network has one significant improvement which allows recurrent edges to span adjacent time steps over the standard feed-forward neural network [10].

The training of the RNN for each neuron in one layer includes a number of iterations based on the pre-defined time steps t. When the predefined iterations are done for each of the hidden layers, then one forward pass is completed and is immediately followed by a standard back-propagation which is applied on all of the recurrent neurons.

In each time step t, the input of the RNN's neuron represents a combination between the output of the same neuron from the previous iteration $t-1$ which uses the weight w_x, the part of the data input corresponding to the current time step t using a new weight w_y and input that sums up information from the previous layer. The summed up information from the previous layer represents the historical step inputs which are merged with information from the previous step input. Furthermore, the current step input will be merged with this historical information to be used for the next layer. Consequently, a decision that a recurrent network has reached at time step $t-1$ will affect the decision it will reach at time step t, meaning that the recurrent neural networks have two sources of input, the present and the recent past, and they are combined to determine how they respond to new data. This represents the main distinction between the recurrent neural networks and feed-forward neural networks: the recurrent networks maintain a feedback loop that is connected to the past decisions and allows them to use the outputs of the neurons as inputs in a next step. The sequential information that the recurrent networks carry is preserved in their hidden state. The hidden state extents to many time steps as it cascades forward to affect the processing of each new example, and it allows it to find "long-term dependencies" between the events. This long-term dependencies can be find by sharing weights over time. Thus, the recurrent neural network maintains a matrix W_x storing the weights of the recurrent edges and W_y storing the weights of the new data inputs. This implementation allows each node to have hidden state which represents a memory cell containing time-dependent information. In the process of training the neurons, a complete forward pass is performed in order to reveal a chain of linked cells. The previous can be summed up with the following expression:

$$y_t = \phi(W_x \cdot x_{t-1} + W_y \cdot y_t + b) \tag{4}$$

where ϕ represents the chosen activation function, and b represents the bias vector with a size equal to number of neurons in the RNN network.

Due to this important improvement [10], the RNNs have the ability of modelling temporal dependencies along the temporal dimension of the accelerometer data, and thus, a very specific RNN called long short-term memory (LSTM) is chosen for the purposes of this thesis. The vanilla RNN's main shortcomings are connected to the effects of the vanishing and exploding gradients, therefore, different improvements of the vanilla RNN implementation try to cover the flaws of the vanilla RNNs and one particularly good solution is the LSTM.

Long short-term memory (LSTM). The long short-term memory (LSTM) network is chosen for the purposes of this thesis in order to model the complex time-dependencies which the vanilla recurrent neural networks are incapable to model properly [12][24]. Each LSTM cell has internal mechanisms, called gates which are responsible for the regulation of the information flow. Starting from their introduction in 1997 [12] and after several refinements, now they are widely used for problems in many different areas including time series data such as speech recognition, machine translation, language modeling, etc.

The LSTM's design is upgraded with the powerful concept of cell state [24]. Instead of having a simple memory chain which in time weakens, the cell state allows the information to flow from the beginning of time (beginning of training) up to a specific moment in time allowing minor interactions that can merely change small parts of the "collective" memory of the network. Additionally, the LSTM's design includes control gates. The main responsibility of the gates is the information interactions in the cell state, and thus, adding or removing information to the cell state is done by following a specific order of operations including the sigmoid function, the tanh function, point-wise matrix multiplication and point-wise matrix addition. The following steps include the key points reflected during one iteration in one LSTM cell presented in Figure 11:

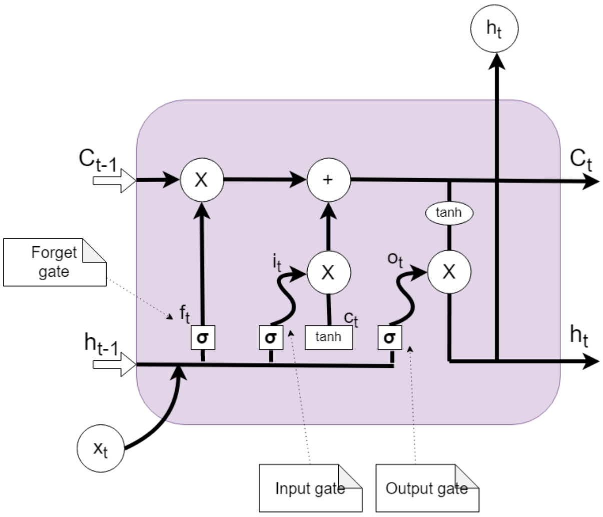


Figure 11. A full architecture of the LSTM cell including the cell-state pipe, the 'forget' gate, the 'input' gate and the 'output' gate.

1. In the first step, a combination of the input data x_t and the hidden state h_{t-1} is applied in the forget gate. The forget gate incorporates the sigmoid function and it decides what information will be retained from the cell state C_{t-1} based on the input. The output of the sigmoid function is a value between 0 and 1. For each number in the cell state C_{t-1}, the value can be considered a weight, where a higher weight (value closer to 1) indicates higher significance of that piece of information, otherwise it is irrelevant and should be discarded.

2. Next, the combination of the input data x_t and the hidden state h_{t-1} passes through two control mechanisms. The first of the two represents a sigmoid activation function, also known as input gate, and the second one is a tanh function. The purpose of this sigmoid activation function differs slightly from the one in the

previous step - the values define what part of the combined data should be updated. In addition, the tahn activation function is responsible for defining the vector of values C_t that might be updated and added to the cell state based on the output values in the next step.

3. The first part of this step is updating the old cell state C_{t-1} into the new cell state C_t. To achieve this, multiplication between the old state C_{t-1} and the output from the forget gate is necessary. The second part is a point-wise addition between the new cell state C_t and the old cell state.

4. The last step produces the output by filtering the cell state, thus is referred as an output gate. The filtering is done using sigmoid and tanh activation functions. The sigmoid function is used for selecting the parts of the cell state that will represent the output, while the tanh function normalizes the values in the range between -1 and 1. The final output of the LSTM cell represents the the multiplication between the outputs of the two aforementioned activation functions.

3 Related work

In this section, an overview of the research on accelerometer-based activity recognition is given, followed by a description on some of the open questions and best practices. According to the comprehensive study by Niall Twomey et al. [8], there is no unique approach which guarantees the best results. However, the authors [8] indicate that similar setups incorporate and follow a specific set of practices. The practices include the types of classification models, evaluation methods, accelerometer sampling rates, strategies for time series segmentation, the activity set and feature extraction techniques. The presented list is not comprehensive, and based on the differences in the setups, additional practices may be included, while also some of the existing ones may be excluded.

In the study by Tâm Huynh et al. [25], the use of tri-axial accelerometer data produced by one sensor attached to the person's arm is focused on deriving a method for feature importance and activity classification for six different activities including *walking*, *standing*, *jogging*, *skipping*, *hopping* and *riding bus*. The feature importance method is based on clustering, and it outputs the cluster precision. Given the presented results, the features that have the highest cluster precision are those derived from the Fast Fourier transform coefficients. Additionally, the best classification results are recorded when the segmentation process of the time series data generates small time windows up to 2 seconds.

In another study by Ling Bao et al. [26] the data that is being used is produced by subjects who were wearing 5 bi-axial accelerometers. The study [25] extends the list of useful features from the previous study [25] with the mean, energy, correlation, etc. The presented results [26] point out that the best models which can be fitted according to the previously mentioned features are the decision trees. Additionally, the presented study by Niall Twomey et al. [26] points out the possibility of involving different body locations for the sensor, but also to extend the list of basic activities with the everyday household activities. In addition, the study by Nishkam Ravi et al. [27] points out that one of the most consistent approaches for solving the classification activity recognition task is to use plurality voting because of its consistent performances across different settings. The plurality voting approach incorporates a group of classifiers [27]. Each classifier outputs its prediction (vote), whereas the class with the largest number of votes is taken as the final prediction. Nevertheless, in the study by Nishkam Ravi et al. [27] the authors counter the need of concurrently using more accelerometers compared to the study from Ling Bao et al. [26] by specifying that most of the activities can be recognized by using only a single tri-axial accelerometer and plurality voting based on decision trees.

On the other hand, the overall activity recognition task can be viewed from a different perspective and can be approached by solving smaller problems in order to provide fusion of solutions that ultimately will provide classification results. One such fusion solution is presented in the study by Jamie Ward et al. [28] where the authors use data from two different sources, accelerometers and microphones. All of the necessary activities are segmented from the original stream data, and for these segments Hidden Markov Models are applied on the accelerometer data, while Linear Discriminant Analysis (LDA) is applied for the sound channel. The overall idea presented in the study by Jamie Ward et al. [28] is that the recognition of activities can be performed after performing a fusion between two or more data sources.

Recently, the neural networks approach has been providing comparable or better results in regards to the traditional methods [29] [30] [31]. In the last decade, neural networks have experienced a huge expansion, therefore, their impact in the area of activity recognition has been inevitable. Applying neural networks on the raw accelerometer data has proven successful. In the study by Sojeong Ha et al. [29] the authors explore the possibilities of different configurations for applying convolutional neural networks using the convolution and pooling operations on the raw data while being able to successfully capture the temporal dimension. The derived CNNs in the study [29] has the ability to capture the modality specific characteristics and the common characteristics of the sensor data by using two layers in the selected architecture. The results [29] point out that one-dimensional CNN and the modified CNN presented outperform all of the other used machine learning models. Nevertheless, the convolutional neural networks represent only one type among the many different types of network architectures for addressing the HAR problem.

Furthermore, successful results have been reported by using recurrent neural networks [30]. In the study by Yu Guan et al. [30], the authors investigate the possibilities of the LSTM neural networks as a special case of the RNNs. The main goal of the study [30] is to explore the possible application of the LSTM cell on accelerometer data. Furthermore, it explores how to develop an ensemble of LSTM networks suitable for recognition of activities. In order to achieve this goal they define modified training procedures for the LSTMs. The main difference with other related studies is the alleviation of the sliding window paradigm by replacing it with sample wise prediction, which increases the robustness of the final recognition system.

Finally, since the CNNs and LSTMs have proven to be useful in the HAR domain, in the study by Francisco Ordóñez et al. [31] an implementation of a deep neural recognition model is presented as a combination of the aforementioned neural networks where the first layers is a convolutional layer followed by a pooling operator, while the next two layers are LSTM cells. The presented architecture of the deep neural network is tested on two publicly available data sets against linear discriminant analysis (LDA) model, quadratic discriminant analysis (QDA), feed forward neural network and convolutional neural network. The displayed results highlight best accuracy performance for the presented deep neural recognition model.

This general overview of the related work tried to capture the chronological order of addressing the HAR problem starting from the early 2000's, when basic machine learning methods were being used, until today when complex frameworks have been developed.

Regardless of the chosen model for solving the HAR problem the most important issues and the general approach to them have been the same throughout the last two decades. Therefore, in the following sections the most important open questions related to activity recognition that need specific addressing are discussed.

3.1 Activities

Before addressing a specific activity recognition task, the first step is defining the set of activities. Many of the aforementioned studies investigate and try to discriminate different sets of activities collected under various circumstances. In addition, some of the used data sets include atomic activities such as *walking*, *standing*, *jogging*, *sitting* and *hopping* [25] [26] . Others [30] [31] include complex activities of daily living such as *open and close fridge*, *vacuum cleaning*, *clean table*, *drink from cup*, *preparing and drinking a coffee*, *preparing and eating a meal*, *cleaning up*, etc. The comprehensive study for accelerometers by Niall Twomey et al. [8] highlights a list of seventy activities, including activities of daily living (ADL), posture, ambulation and transition activities that are being used throughout most of the studies related to human activity recognition. Furthermore, the activities in the presented list can be separated in three subgroups of activities. The first group includes activities that don't require hand usage (e.g. *walking*, *sitting*), the second group includes activities that require usage of only one hand (e.g. *brushing teeth*), while the third group includes activities that require the usage of both of the hands (e.g. *flossing*, *pouring liquid from bottle in a cup*). Even though the presented list is not formal and definite, all of the aforementioned studies work with some subset of activities that is part of this list. Therefore, for the purposes of this thesis the presented list by Niall Twomey et al. [8] is used as a guideline to point out to the possible activities that can take part of the final set of activities. Additionally, the chosen list of activities in this thesis also follows the informal rule of hand-usage in order to provide three groups of activities.

3.2 Time-series segmentation

This section is solely based on the study by Oresti Banos et al. [32]. The study [32] summarizes various time series segmentation possibilities when the data is used for recognition of activities. One of the vital parts for effective recognition for given activities is choosing the best segmentation strategy for splitting the accelerometer time-series data. The study [32] highlights three different strategies such as activity-defined windows, event-driven windows and sliding windows.

The main goal of the activity-defined windows is capturing the starting moment and the ending moment of one activity. By defining the beginning and the end for each of the activities the accelerometer data is labeled accordingly and the time series is segmented into small subsets with different time duration where each of these corresponds to only one activity. Nevertheless, the biggest challenge when defining this type of segmentation is discovering the precise moment in time when the transition between the current and the following activity occurs. In order to address this challenge various strategies exists such as analysis of the variations in the frequency characteristics of the data, heuristics that are able to differentiate among static and dynamic actions, notations provided by the participant which has the wearable attached, etc. The last strategy is used by SPHERE [1], where the participants explicitly state the performed activities using an audio recorder. Later the activities are transcribed.

The need of event-defined windows stems from complex activities (e.g. *cleaning*, *food/drink preparation*) which can be dissolved in a sequence of simple movements or actions. Furthermore, in order to detect movement or actions an identification of events needs to be performed. The time located events which are a result of the simple movements or actions as part of complex activities are used for determining the segmentation points in the time series data. In order to detect these specific events many strategies can be used such as movement gait analysis, phase detection in the gait, Gaussian mixture models for classifying the event related data samples. Nevertheless, the event-defined window approach is not incorporated in this thesis, however, more information can be found in the study by Oresti Banos et al. [32].

The sliding window or the "windowing" approach is widely accepted as a segmentation technique for activity recognition. The main reason why this approach is more popular compared to the other two approaches is because the pre-processing part is fairly straightforward to implement by excluding the need to use complex methods. The entire pre-processing part of this approach is the segmentation of the time series data in small, continuous and consecutive subsets of data with fixed and equal size. In addition, this approach also allows overlap between the fixed consecutive windows and this can be easily modified during the pre-processing, however, in practice this is rarely used. The success of this method is highly notable especially when recognition on periodic and static activities needs to be performed. Nevertheless, this method is not limited only to these types of activities, and also can be applied when complex activities are included in the recognition process. Overall, the aforementioned studies use the windowing approach due to its implementation simplicity. In addition, the incorporated accelerometer data in their work is already labeled for recognition purposes and can easily be partitioned in windows with fixed size.

3.3 Relevant features in the accelerometer data that are useful for prediction

One of the essential parts in solving the HAR problem is defining suitable features derived from the accelerometer data. In order to have successful recognition of activities the feature extraction process must be carefully designed and followed by feature selection process. At the end of this vital part a relevant set of features will be used for the classification task. Even though the process of feature extraction is not single and definite and it can be done in various ways, most of the derived features can either be classified as time-domain features or frequency domain features. Time-domain features include features which are derived after a statistical analysis performed directly on the accelerometer data and they include computations such as the mean, standard deviation, correlation, magnitude and many other artificial constructs that can be derived from the time series data. On the contrary, additional computations are required in order to transform the sensor data from time domain to frequency domain using the Fast Fourier transform (FFT). The coefficients of the output vector of the FFT represent the sensor data in frequency domain. Similar to the statistical analysis performed on the time series data, additional analysis is performed on the received FFT coefficients in order to produce features such as energy, coherence, maximal spectrum and many other constructs. [8]

All of the aforementioned studies use hand-crafted features, designed solely for their purpose and for the specific activity recognition task. Nevertheless, there are additional feature extraction techniques which can be implemented in one's work [8] in order to avoid manual extraction such as *sparse coding and dictionary learning technique* presented by Chenglong Bao et al. [33] and *fixed dictionaries technique* presented by Lingyue Xie et al. [34]. However, for the purposes of this thesis, as in most of the available literature, the manual extraction of features is considered as the ultimate approach due to its potential to force the practitioner to understand the very basic nature of the accelerometer data.

In the study by Nishkam Ravi et al. [27] the authors implement their own hand-crafted feature extraction process where most of the extracted features are from the time domain. As it can be seen from their results and conclusions, the time domain features are not enough, therefore the suggestion is extending the feature set by including features derived from the frequency domain [8] [25] [26]. Again, if only the frequency set of features is taking part in the final predictions, a lack of information is inevitable. The simple and logical conclusion is to take in consideration both of the feature sets as equally important because the information they hold is mutually exclusive.

At the end of the feature extraction process there can be a lot of features that are not equally useful. Therefore, in order to reduce the number of features a specific feature selection method can partake in the final part of the process. There are many examples in the literature where the authors implement their own feature selection methods such the aforementioned studies [8][25]. On the contrary, other authors may us already existing tools which incorporate selection methods such as filter or wrapper [8]. One particular example is the filter-based approach Relief-F selection tool [35]. Additionally, as presented in one of the aforementioned studies [8] when the number of features is high the principal component analysis (PCA) can be used due to its ability to map the original features into a lower dimensional subspace where the newly constructed features will not suffer from high correlation, thus, reducing the number of features significantly.

3.4 Location of sensors on the body

Following the comprehensive study for accelerometers by Niall Twomey et al. [8] there are up to ten different positions for placing accelerometer sensors, including: hip, wrist, upper arm, ankle, thigh, chest/trunk, armpit, trouser pocket, shirt pocket, necklace. In the studies discussed in the beginning of this section, the data is collected from accelerometers placed on different parts of the body of the participant. In the studies by Tâm Huynh et al. [25], and Nishkam Ravi et al. [27] the acceleration is recorded by accelerometers placed on the wrist of the dominant arm. Additionally, in SPHERE [1] more than one sensor is used simultaneously in order to provide accelerometer data from both legs and arms, while in the study by Ling Bao et al. [26] five different sensors are being used(both on the legs and arms, and additionally on the belt).

The comprehensive study [8] highlights that usually the best accuracy results can be obtained using data that is recorded from accelerometers placed either on the wrist of the arms, the pockets and on the ankles of the legs.

In conclusion, different studies use data recorded either from one accelerometer usually placed on the dominant arm of the person, or from several accelerometers placed on different parts of the body.

4 Methodology

This chapter highlights the necessary parts for providing a full activity recognition pipeline by incorporating the background information presented in Section 2 while following the best practises and guidelines presented in Section 3. The pipeline is separated into three logical units as presented in Figure 12.

The first unit encapsulates the data pre-processing part and includes all of the necessary operations for defining the accelerometer data in a structured format suitable for machine learning models.

The second unit represents the main part of the presented pipeline. The main focus is building the appropriate classification models, the random forest and the LSTM by utilizing the structured accelerometer data. This unit incorporates all of the necessary tools for building and evaluating the machine learning models and it can be split in two sub-units. The first sub-unit is focused on building the traditional machine learning model with a greater focus on the time-series segmentation strategy and the feature extraction process, while the second sub-unit is based on creating the architecture of the LSTM neural network.

The last unit incorporates the built models and utilizes them for activity recognition in the SPHERE's unlabeled accelerometer data segments.

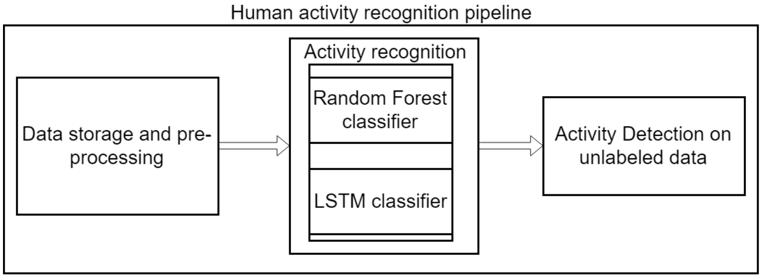

Human activity recognition pipeline
Data storage and pre-processing
Activity recognition
Random Forest classifier
LSTM classifier
Activity Detection on unlabeled data

Figure 12. Overview of the HAR pipeline with its three main units: data storage and pre-processing, activity recognition, activity recognition on unlabeled data.

In the following subsections a thorough overview for each of these units is given.

4.1 Data pre-processing

The starting point of the Human Activity Recognition pipeline is the pre-processing of the data where the sensor records need to be adjusted for suitable usage in the machine learning models accordingly. In addition, this subsection expands on all of the necessary decisions in order to define a structured data format of the accelerometer readings. Therefore, the following subsections focus on the most essential technical details and peculiarities that took part in order to achieve the aforementioned goal. More specific information about additional technical details is given in the Appendix.

4.1.1 Synchronization between the SPHERE wearable system time and the audio recorder time

The wearables incorporate the wearable system time clock and output the time of recording as part of the recording output. Additionally, the audio recorder incorporates the audio time clock. The overall time difference between the two clocks is approximately 4 seconds per day. In addition, there are different annotation files, which are annotated for specific segments of the accelerometer data, but the starting time for each annotation file is unknown. The participant in the SPHERE house signifies the start of the recording of the annotations by performing five consecutive hand taps (claps). Additionally, at the end of the recording session another sequence of five hand taps is performed by the participant wearing the wearable sensors. Given the previous information the main task is to synchronize the two clocks in order to find the starting times for each annotation file.

Due to the enormous amount of accelerometer records, simple filtering of the data in order to find the taps is not the best approach. Therefore, to narrow the time period in which the groups of consecutive taps might occur the two clocks provided by SPHERE are utilized. The specific way of how the clocks are used for narrowing down the possible data segments is beyond the scope of this thesis. Furthermore, a manual check is performed in each of the narrowed segments. The performed check consists of two parts. The first part is a visual inspection of the segment which confirms that a group of five taps is present as shown in Figure 13, while the second part is finding the minimum value of the accelerometer readings that represent the first tap, indicating the possible starting time. In Figure 13 one area of five consecutive taps is presented pointing to a distinction between the records that represent the taps and the other records present in the same data segment. At the end of the synchronisation process the annotation files have their absolute start times defined.

4.1.2 Adapting the annotations following an ADL ontology

The addressed shortcomings of the annotations in Section 2.2 arise the need of adaptation the same in a more strict and compact manner following a specific ontology. The ontology which is used as a benchmark is introduced by Emma L. Tonkin et al. [36] and it defines three levels of activities as shown in Figure 14. The first level in Figure 14 encapsulates groups of many different atomic activities (e.g. home activity). The second level in Figure 14 represents subgroups of activities which are consisted of a small number of atomic activities (e.g. door interaction). The third level in Figure 14 represents only the atomic activities. (e.g. open door/close door). Furthermore, in order to define a clear set

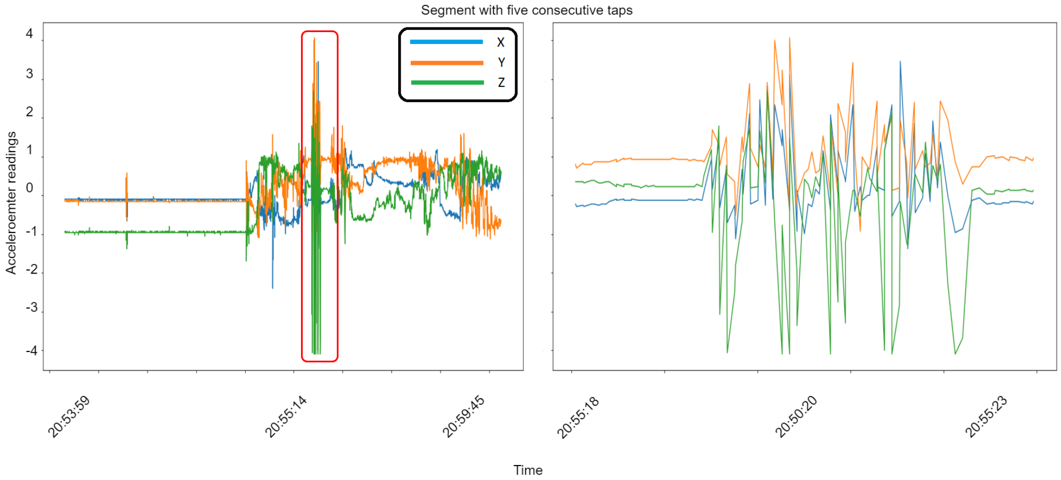


Figure 13. The X, Y and Z accelerometer recordings for a narrowed time segment presented in the first figure. From the second figure it can be seen that there are 5 different peaks representing the 5 different taps.

of activities, for the purposes of the thesis, additional focus is given on the naming in order to define as much clarity as possible. The initial annotations are manually adapted in strict and formal annotations by utilizing the aforementioned three levels of the ADL ontology.

Concurrently, with the annotation adaptation, the location in the house is also manually determined by following a pre-defined rule for location. The rule is based on the movement of the person in the house. In the annotations additional information about the transition of the participant between the rooms is given, therefore, a simple and straightforward rule for checking the last transition is applied in order to determine the last location of the participant. The usefulness of this information stems out of the simple constraint imposed by structure of the rooms that restrict the possibilities to practice all of the possible activities (e.g. washing dishes can not be performed in the bedroom, or in the basement). In the end of the process the annotations are fully adapted and follow a strict and precise format as presented in Figure 15.

The last step is to compute the absolute timestamps for each of the annotations. The annotation files (representing the known segments) consist annotations which have relative timestamps. After the process described in Section 4.1.1 is finished, the starting absolute timestamps for each of the known segments is defined. Therefore, in order to provide the absolute timestamps for each adapted annotation a simple addition between the relative timestamps of the annotation and the absolute starting time of the corresponding annotation file is performed. The computation of the absolute time stamps for each adapted annotation produces activity-defined windows, thus the long time series data is segmented into smaller parts explained by a single adapted annotation.

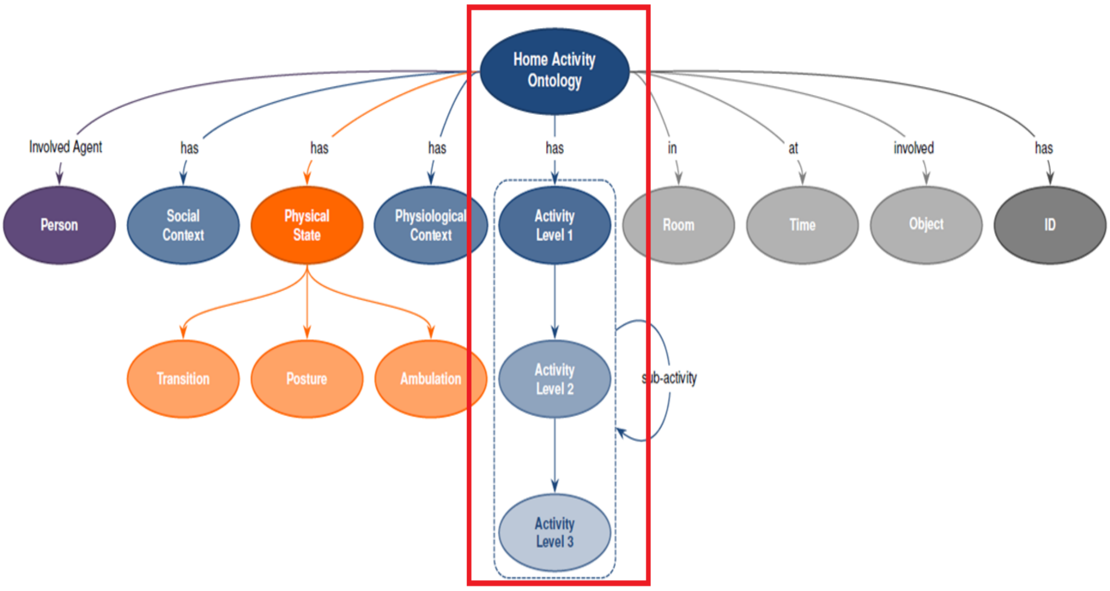


Figure 14. The three levels of activities according to the defined ontology [36]. The first level represents the general activity groups consisted of many subgroups of activities. The second level represents the subgroups of activities. The third level represents all of the atomic activities which may take part of one subgroup, and one general group, respectively. The Figure is part of the study by Emma L. Tonkin et al. [36].

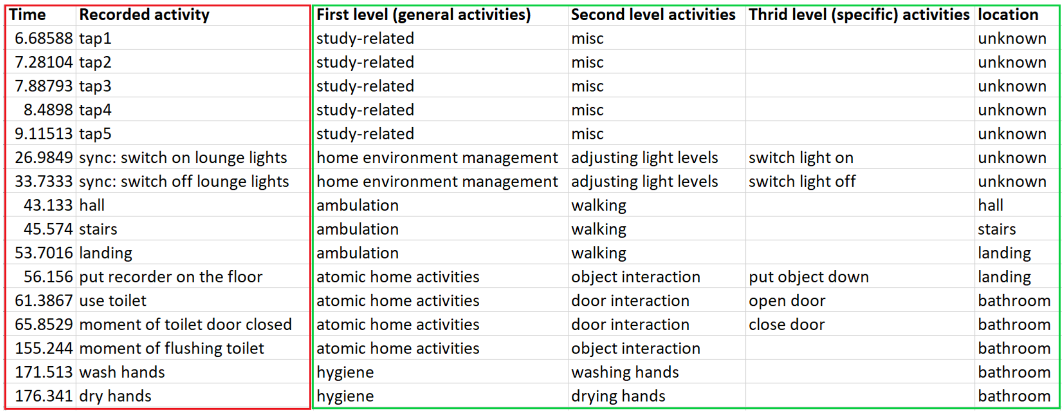
4.1.3 Synchronization between the annotation files and accelerometer readings

The final data sets represent a combination between the fully formatted accelerometer segments addressed in Appendix and the adapted annotations corresponding to these segments addressed in Section 4.1.2. The synchronization is solely based on the derived absolute timestamps for the annotations and the accelerometer records. In the end, the newly generated data sets are stored in separate files following the structure presented in Figure 16. These files are consisted of many activity-defined windows which are easily adaptable for various needs of the pipeline.

4.1.4 Activity selection

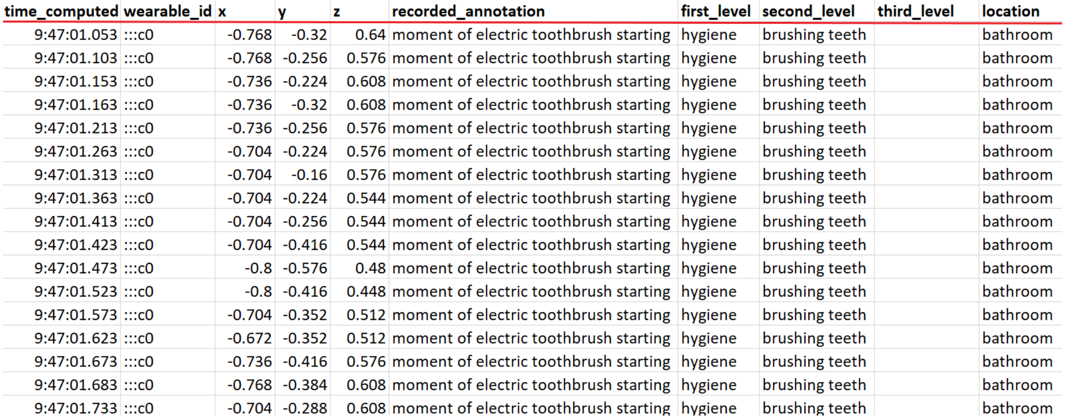
In the end, the process of generating the final accelerometer data sets is followed by the selection of 9 different activities using the adapted annotations. The selected activities are highlighted in Figure 17. The chosen activities can be split in groups regarding their complexity and regarding hand usage.

Complexity-wise there are two different groups of activities. The first one includes the atomic activity *walking* and the second one includes the remaining 8 ADL activities.



Time	Recorded activity	First level (general activities)	Second level activities	Thrid level (specific) activities	location
6.68588	tap1	study-related	misc		unknown
7.28104	tap2	study-related	misc		unknown
7.88793	tap3	study-related	misc		unknown
8.4898	tap4	study-related	misc		unknown
9.11513	tap5	study-related	misc		unknown
26.9849	sync: switch on lounge lights	home environment management	adjusting light levels	switch light on	unknown
33.7333	sync: switch off lounge lights	home environment management	adjusting light levels	switch light off	unknown
43.133	hall	ambulation	walking		hall
45.574	stairs	ambulation	walking		stairs
53.7016	landing	ambulation	walking		landing
56.156	put recorder on the floor	atomic home activities	object interaction	put object down	landing
61.3867	use toilet	atomic home activities	door interaction	open door	bathroom
65.8529	moment of toilet door closed	atomic home activities	door interaction	close door	bathroom
155.244	moment of flushing toilet	atomic home activities	object interaction		bathroom
171.513	wash hands	hygiene	washing hands		bathroom
176.341	dry hands	hygiene	drying hands		bathroom

Figure 15. The format of the adapted annotation files. The red rectangle highlights the initial annotations, while the green rectangle higlihgts the annotations after they are adapted.



time_computed	wearable_id	x	y	z	recorded_annotation	first_level	second_level	third_level	location
9:47:01.053	:::c0	-0.768	-0.32	0.64	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.103	:::c0	-0.768	-0.256	0.576	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.153	:::c0	-0.736	-0.224	0.608	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.163	:::c0	-0.736	-0.32	0.608	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.213	:::c0	-0.736	-0.256	0.576	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.263	:::c0	-0.704	-0.224	0.576	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.313	:::c0	-0.704	-0.16	0.576	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.363	:::c0	-0.704	-0.224	0.544	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.413	:::c0	-0.704	-0.256	0.544	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.423	:::c0	-0.704	-0.416	0.544	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.473	:::c0	-0.8	-0.576	0.48	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.523	:::c0	-0.8	-0.416	0.448	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.573	:::c0	-0.704	-0.352	0.512	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.623	:::c0	-0.672	-0.352	0.512	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.673	:::c0	-0.736	-0.416	0.576	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.683	:::c0	-0.768	-0.384	0.608	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom
9:47:01.733	:::c0	-0.704	-0.288	0.608	moment of electric toothbrush starting	hygiene	brushing teeth		bathroom

Figure 16. Final format of the accelerometer data representing the accelerometer readings and their corresponding annotations.

Hand-usage wise there are three different groups of activities such as hands free activities (*walking*), activities that require only one hand (*brushing teeth*, *writing)*, and activities that require the usage of two hands (*flossing*, *getting dressed/undressed*, *washing hands*, *mixing (food)*, *spreading (food)*, *eating a meal*). The goal is to define models that can discriminate between similar activities, activities with different level of complexity and activities which can be placed in the same hand-usage group.

4.2 Traditional machine learning approach

The core of the HAR pipeline are the traditional machine learning approach and the neural network approach. This section provides greater details for the traditional ma-

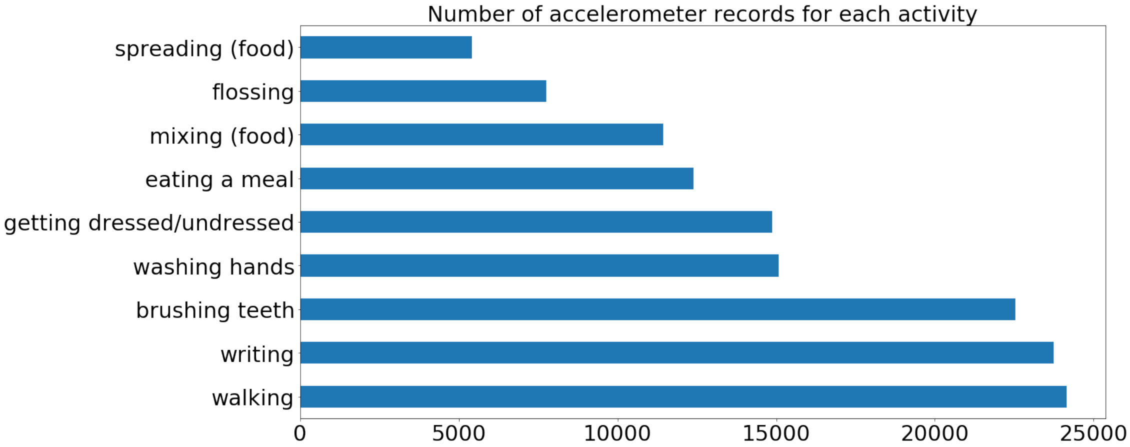


Figure 17. The number of accelerometer readings for each of the selected activities. The selected activities include: *brushing teeth*, *eating a meal*, *flossing*, *getting dressed/undressed*, *mixing (food)*, *spreading (food)*, *walking*, *washing hands*, *writing*.

chine learning approach. The main goal is developing two random forest classifiers for recognition of activities using the accelerometer data recorded from the wearables placed on the wrists of both hands.

First, a time series segmentation strategy is applied on the structured accelerometer data. Additionally, greater focus is given on the feature extraction process. Custom classifiers for different activities are defined in order to provide greater information whether the accelerometer data is suitable for feature extraction. The custom classifiers incorporate several feature extraction techniques and highlight whether there are visible differences between the received feature values for different activities. The general feature extraction process incorporates the findings provided by the custom classifiers, thus, a bigger set of features is defined. The last step includes building the random forest models based on the feature extracted values and performing an evaluation as part of the parameter tuning process. The overall architecture of the traditional machine learning component as part of the HAR pipeline is part of the Appendix. The following subsections expand on the aforementioned topics in greater details.

4.2.1 Time series segmentation

The windowing process has a significant role in defining the final data set used for building the random forest model. The windowing approach is used for further time series segmentation over the activity defined windows addressed in Section 4.1.2 in order to produce approximately equal sliding windows without introducing overlapping. Each activity defined window which is longer than 10 seconds takes part in the segmentation, and all of the activity defined windows which are below 10 seconds are discarded. The segmentation process of the activity defined windows defines time windows with a length of 10 seconds as provided in Figure 18.

The split is set to 10 seconds because the activity defined windows with shorter length may not carry sufficient information. On the contrary, long time windows carry a big amount of information. Therefore, the time series segmentation includes a trade-off: the increase of the window length increases the positive benefits when performing feature extraction, however, if the sliding window length becomes too large, the feature extraction might not capture everything that is important, and also there is the risk that some parts of the long data window may contain traits of other activities.

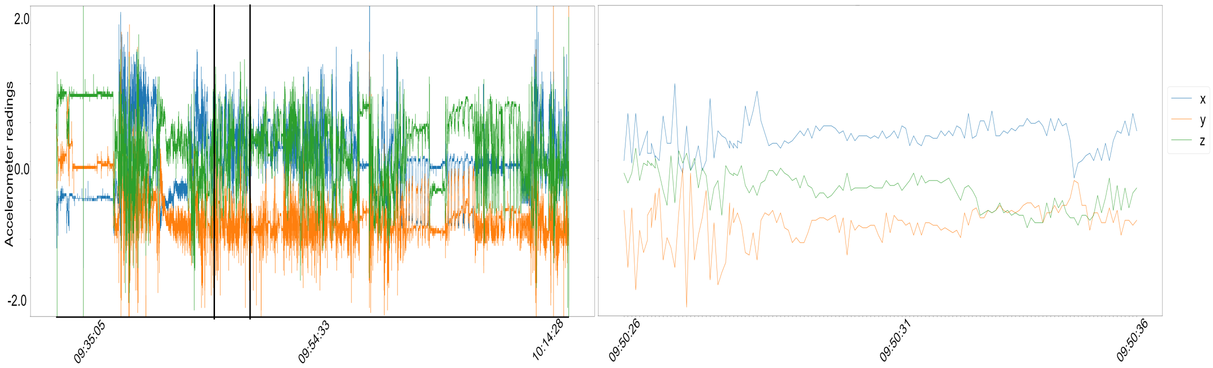


Figure 18. Time segmentation of the long time series data in small time windows with a length of 10 seconds. The time window encapsulating the activity *walking* presented in this Figure has a length of 10 seconds, consisting up to 240 accelerometer readings for the X, Y and Z axes.

In summary, the sampling rate of the accelerometers is $24Hz$, and the fixed length of the windows is 10 seconds, thus, each window will have up to 240 samples.

4.2.2 Custom classifiers

The custom classifiers are derived to confirm whether the accelerometer data is suitable for feature extraction. The main intention of these classifiers is to point the existence of activity-specific patterns. The found patterns can be further used for discrimination between different activities.

The process of pattern discovery is done using six different custom classifiers. These classifiers extract basic information from the accelerometer data in order to determine unique patterns for six different activities. The included activities require the usage of one hand such as *writing* and *brushing teeth*, the usage of both hands such as *washing hands*, *eating a meal*, and *flossing*, and hands free activities such as *walking*.

The accelerometer readings provided from the wearable devices attached on both hands are utilized for this task. The data for each activity is segmented in time windows with a length of 10 seconds. All of the features are computed within the time windows.

The initial features extracted for the aforementioned six activities are the mean values computed per axis over a time window. Additionally, the standard deviations are also computed. Figure 19 provides information related to the distribution of the mean values, the standard deviations and the outliers over the X, Y and Z axes for each activity, respectively. The distribution of mean values for the time windows over the X axis in Figure 19 points to a distinction between *brushing teeth* and all other activities. Furthermore, the distribution of mean values over the Y axis highlights the distinction between *flossing* and the rest of the activities. Additionally, the Z axis analysis provides clear distinction *brushing teeth* and *walking*.

In conclusion, axis-wise features offer poor discrimination when used in isolation. However, a clearer distinction when the features from all axes are used together for the activities *writing*, *brushing teeth*, *flossing* and *washing hands* can be achieved. On the contrary, these features do not provide enough information for describing the activities *walking* and *eating a meal*.

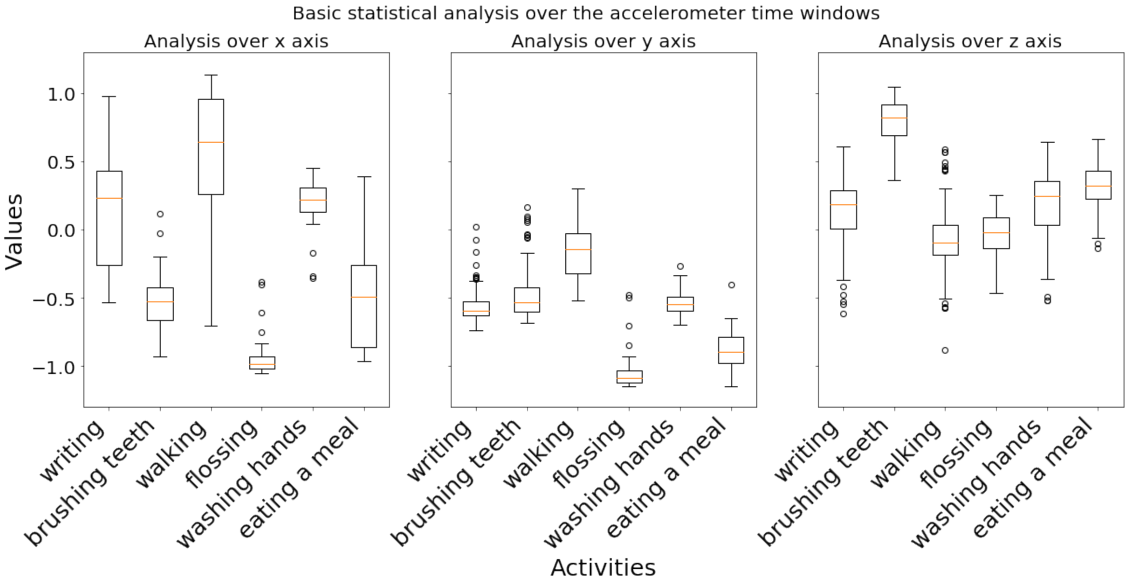


Figure 19. The distribution of the mean values computed over the time windows for each axis. The computations are performed for six different activities: *writing*, *brushing teeth* *walking*, *flossing*, *washing hands* and *eating a meal*.

Further investigation is done using similar basic features including the maximum axis value, minimum axis value and median axis value. Nevertheless, these have proven to be less important when compared to the means and their corresponding standard deviations.

Furthermore, the crossings between the axes are investigated as shown in Figure 20. The example of one segment of the activity *washing hands* provided in Figure 20 points

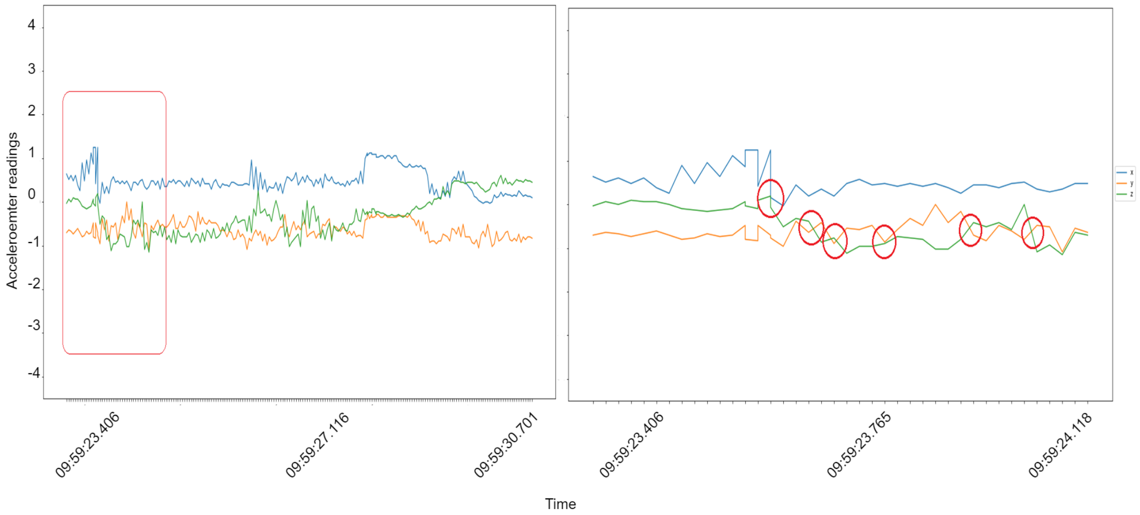


Figure 20. The accelerometer readings for the X, Y and Z axes for one time window of the activity *washing hands*. The red window in the left sub-figure represents the segment provided in the right sub-figure together with all of the crossings between the three axes.

out to the numerous crossings between the X, Y and Z axes.

For each activity a range of possible values is defined using the mean value of crossings and the corresponding standard deviation as presented in Figure 21. The mean of the number of crossings between the X and Y axes points out that *washing hands* and *flossing* are quite similar. On the contrary, the mean of the number of crossings between the X and Z axes highlights the difference between these two activities whereas for *flossing* there are almost no crossings, while for *washing hands* there are more than 40 crossings in average. Similarly, the difference between *writing* and *brushing teeth* can be concluded from the mean of the crossings between X and Y axes where the averages are near 10 and near 30, respectively. Additionally, the difference between *writing* and *brushing teeth* can be seen from the mean number of crossings between the Y and Z axes where the averages are above 50 and near 0, respectively. However, the means of the number of crossings between X and Z axes for the activities *writing* and *brushing teeth* are almost equal. The same can be applied for the other activities when the crossings between the pairs of axes are combined and investigated together.

Next, for each activity, the simple linear regression method is used for finding the slope of the regression line per axis, as described in Section 2.4.1. The slope ranges computed in degrees for each activity are presented in Figure 22. When the focus is on the slope ranges based on the X axis there is a distinction in the distributions between *writing*, *brushing teeth* and *eating a meal*. The Y axis slope distributions provide distinction between *walking* and *flossing* where the highest number of slopes are in the ranges from 0 to 30 degrees and -10 to 5 degrees, respectively. The Z axis slope distributions of

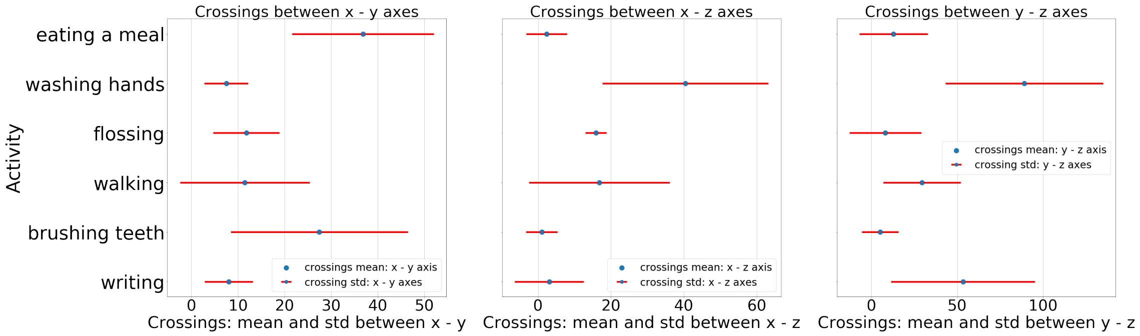


Figure 21. The mean value of the overall intersects between the pairs of axes values computed over the time windows along with their corresponding standard deviations for six different activities: *writing*, *brushing teeth walking*, *flossing*, *washing hands* and *eating a meal*.

writing and *brushing teeth* additionally confirm their difference. However, the differences in the slope ranges for X axis between the activities *writing* and *brushing teeth* are quite poor. In conclusion, discrimination can be achieved when the slope distributions over all axes are used together.

In summary, any traditional ML model will benefit from the artificially constructed features. Additionally, the custom classifiers point out that the accelerometer data is suitable for feature extraction. The aforementioned features are not definite and are solely made from a human's perspective by evaluating and observing the actual accelerometer readings in both, vector representation and graphical representation. The derived features represent the basis of the feature set used in this thesis.

4.2.3 Feature engineering

The feature engineering process is used to improve the discriminatory power of the machine learning model by providing the necessary information extracted from the data. In general, the feature extraction process represents the creation of artificial constructs that take part in both, the learning process and the evaluation process of the model. Additionally, this process is especially helpful for the practitioner to understand the basic concepts of the specific area in order to provide the meaningful constructs.

Feature engineering is especially needed if there is not enough information present in the initial data, and additional assumptions need to be made by creating new features out of the existing data. Furthermore, the generation of new features also improves the discriminatory power of many of the models which have limited capacity to utilize the data in a proper way. In addition, there are many hidden patterns which may only be visible to a human. Therefore, generating features enhances the model's ability to learn additional dependencies resulting in increased discriminatory capacity.

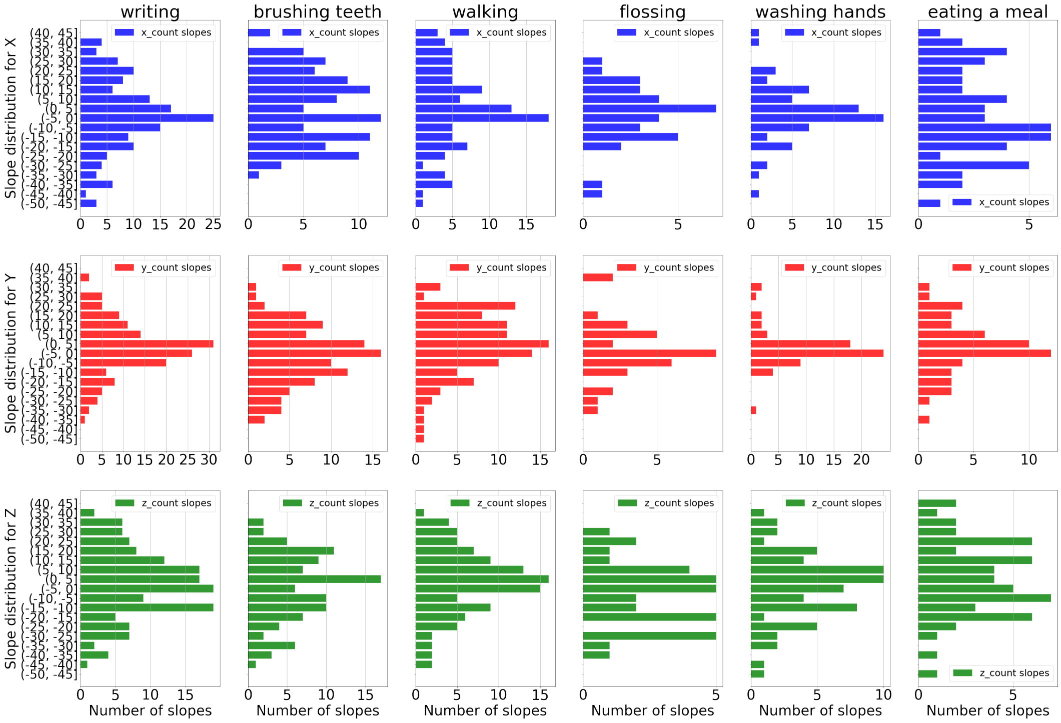


Figure 22. The slope distributions calculated in degrees for the X, Y and Z axes. The simple linear regression method is applied on the accelerometer data for six different activities: *writing*, *brushing teeth walking*, *flossing*, *washing hands* and *eating a meal*.

Following the guidelines and best practices presented in Section 3.3 two sets of features are derived for the purposes of the traditional ML model including the time-domain set and frequency-domain set.

The time domain feature set extends the used features in the custom classifiers in Section 4.2.2. All of the derived features are directly extracted from the sensor data without additional transformations. Furthermore, all of the features are computed over the defined time windows addressed in Section 4.2.1. A subset of the time domain features is presented below.

1. Mean

   This feature represents the mean value of the accelerometer readings per axis. In total, three mean features are constructed, given that there are measurements for three axis. The following expression is used for computing the mean over a time (sliding) window for one axis:

$$mean(a_1, a_2, ..., a_n) = \frac{1}{n}\sum_{i=1}^{n} a_i \tag{5}$$

where a_i is the accelerometer record in g units with respect to the X, Y and Z axes, and n is the number of accelerometer records in one sliding window(e.g. sliding window with up to 240 samples).

2. Standard Deviation

   The standard deviation is computed per sliding window, for each of the three axes of the accelerometer. The computation is presented by the following expression:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(a_i - mean(a_1, a_2, ..., a_n))^2} \tag{6}$$

   where a_i is the accelerometer record in g units with respect to the X, Y and Z axes, n is the number of accelerometer records in one sliding window.

   Furthermore, additional simple computations are performed in order to generate values such as the minimum value per axis, maximum value per axis, the median value, the mode, etc.

3. Overall magnitude.

   The overall magnitude of the X, Y and Z axes represents the acceleration in a sliding window. This feature is preferred especially when the orientation of the sensor is not known [7]. The following expression encapsulates the magnitude:

$$acc = \sqrt{(x^2 + y^2 + z^2)} \tag{7}$$

4. Correlation.

   The correlation feature is calculated between each pair of axes, thus, in practice three different features are defined. The usefulness of this feature is that it can be used to discriminate between periodic activities such as *walking* that have only translation from one to another dimension and ADL activities which include different movement patterns such as *washing hands* where the correlation values may point out to translations in more than one dimension [27]. The standard definition for the correlation is summed up in the following expression:

$$corr(x, y) = \frac{cov(x, y)}{\sigma_x \sigma_y} \tag{8}$$

5. Crossings of readings from different axes

   The computed ratio of the crossings between the pairs of axes generates three different crossing features. The idea for this feature is derived after the visual inspection of the accelerometer data as presented in Figure 20 while constructing the custom classifiers. The goal is to discriminate between the activities that produce many spikes while performing a specific activity, thus, the spikes produce many crossings between the axes. By capturing the interactions, the group of activities for which the accelerometers produce steady values with low standard deviation can be easily separated from those activities that produce high standard deviation.

6. Slope steepness

   This feature provides the steepness of the slope calculated in degrees in one time window with respect to the X, Y and Z axes. The slope is found by using the simple linear regression model presented in Section 2.4.1. The input for this model are the acceleration values for one axis, thus, three different slopes are created respectively. The slope steepness of the regression line follows a specific distribution for different activities as presented in Section 4.2.2.

7. Variance explained by the principal components

   This feature which is explained in greater details in Section 2.4.2 is computed by using the data readings for the X, Y and Z axes per time window as input to the principal component analysis machine learning model. The output of the the PCA model are the principal components and their variances. Usually, the first two components explain approximately 90% of the total variance, therefore, two different features representing the first two principal components with their variance values accordingly are derived by using the PCA model.

The frequency domain feature set is used for capturing the data periodicity. The first step in constructing the frequency feature set is a transformation from time-domain to frequency domain. The transformation is performed by utilizing the Fast Fourier transform (FFT). The FFT is addressed in greater details in Section 2.4.3.

The Fast Fourier transform for the purposes of this thesis is used in two different ways for each sliding window. The difference in the usage between the former and the later is the input to the FFT. The input in the former is the accelerometer 3D data matrix, while the input in the later is per axis, thus, the input is a 1D vector. Nevertheless, the output for both approaches is a 1D vector which represents the components (coefficients) of the FFT for a specific sliding window. The reason behind applying the joined approach is because it offers information from the data such as computed spectrums on a sliding window level that the separated approach is not able to capture. On the contrary, the separated approach is used because it produces frequency coefficient outputs for each axis, thus, providing the possibility to compute the correlation between the pairs of outputs respectively.

In Figure 23 the activities *writing* and *walking* are presented in the frequency domain, respectively. The frequency window representing the *writing* activity in Figure 23 has maximal power of $4Db$ (decibels) when the frequency is close to $1Hz$. Additionally, the frequency window of the *walking* activity in Figure 23 has its maximal peaks near $1.5Hz$, $5Hz$ and $8Hz$ with the powers of $5Db$, $2Db$ and $1.5Db$, respectively. In conclusion, visible differences in the power spectrums of the two activities are present. This is useful for further analysis from which additional features can be extracted.

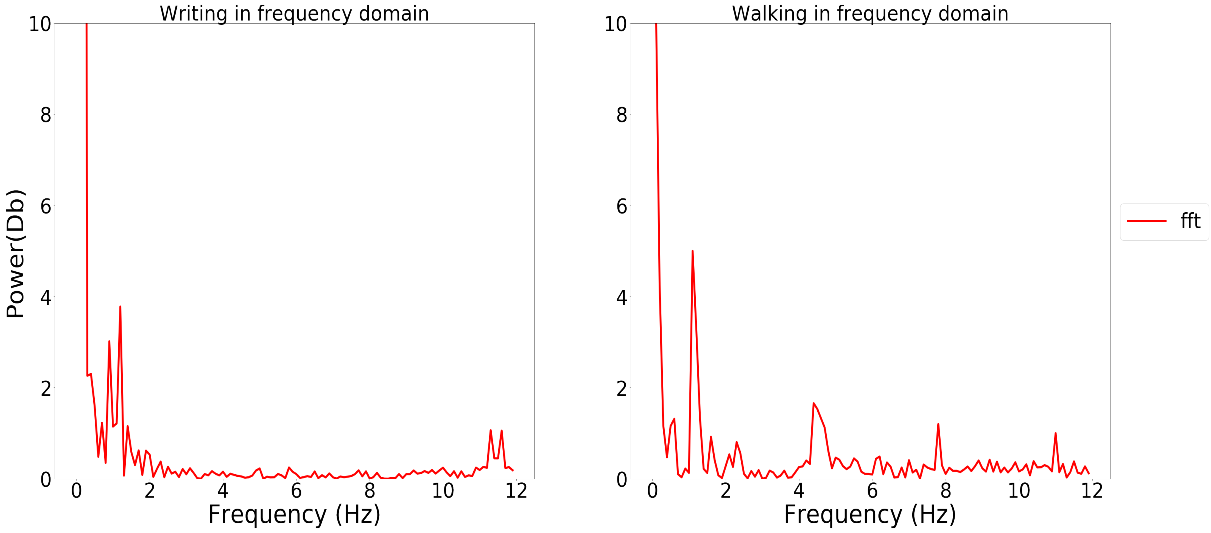


Figure 23. The left sub-figure represents the frequency window of one sample *writing* activity after the FFT is applied on a time window with a length of 10 seconds. The right sub-figure represents the frequency window of one sample *walking* activity following the same setup.

The next step, after applying the FFT, is to extract features from the FFT coefficients. Overall, the extracted features are derived from the outputs of the joined input and the per-axis inputs. A subset of the frequency domain features is presented below.

1. Frequency power spectrum when FFT is applied over the 3D accelerometer data window.

   The FFT coefficients point out to the frequency power spectrum. The human activities are always below $15Hz$. Therefore, for the purposes of the thesis, four peaks are taken into consideration, by splitting the coefficient vector in four equal quadrants. Furthermore, the computations in the aforementioned setup generate four features, where the values for each of them is the power spectrum in each quadrant with a size of $3Hz$, starting from $0Hz$. This can be seen in Figure 24.

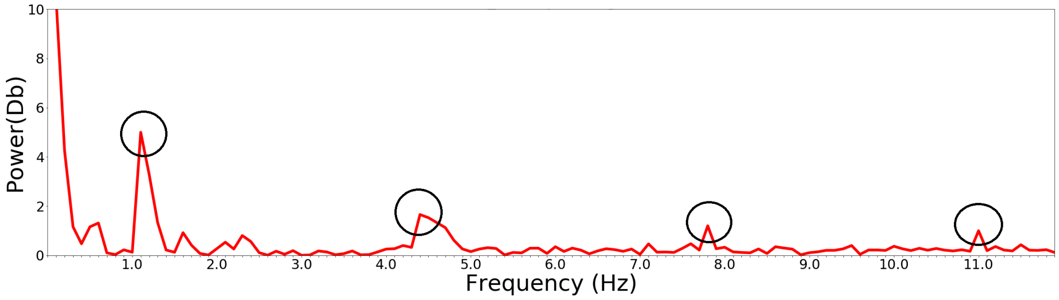


Figure 24. The power spectrums present in one frequency window of the activity *walking*.

2. Frequency power spectrum when FFT is applied over the values for each axis separately.

   Furthermore, the FFT coefficients received when the FFT is performed separately for each axis point out to the frequency power spectrums in each vector. All of the rest follow the same approach explained in the aforementioned feature extraction technique.

3. Correlation in frequency domain

   This feature is representing the pair-wise correlation computations between the 1D FFT vectors respectively. Therefore, in practice three different features are defined.

4. Energy.

   Additionally, the energy feature is computed for both, the combined input (when the FFT is applied over the 3D accelerometer data), and the separated input (when the FFT is applied on the values of each axis separately). In order to generate the value for this feature first the sum of the squared discrete FFT coefficients is computed [27]. Next, the sum is divided by the length of the time window for normalization. The following expression represents the energy in frequency domain for one time window:

$$Energy = \frac{1}{n} \sum_{i=1}^{n} x_i^2 \tag{9}$$

   where $x_1, x_2, ...$ are the FFT coefficients and n represents the number of recordings in a specific time and frequency window.

In conclusion, the extracted features are useful because additional information can be inferred from the accelerometer data. Additionally, the feature extracted values replace the long time series, making the training process faster and more efficient. According to the presented work in Section 3 and given the various feature extraction possibilities it can be concluded that a traditional ML model using extracted features can partially mitigate its limitations.

4.2.4 Random forest classifiers

For the purposes of this thesis one of the used models for recognition of activities is the random forest classifier, addressed in greater details in Section 2.5.1. The majority of the presented articles in Section 3 utilize the random forest algorithm for classification purposes. Additionally, most of these articles also highlight that among the many traditional ML models, the random forest usually outperforms the rest. Therefore, by following the best practises and guidelines from Section 3 the random forest algorithm is selected as the most appropriate for addressing the human activity recognition tasks.

The final step before training the random forest model is removing all of the unnecessary data. After applying the windowing approach and extracting the defined features, the initial acceleration values for the X, Y and Z axes need to be removed. Furthermore, the FFT coefficients also need to be removed. After removing these parts of the final data, each sliding window has samples with identical values. Additionally, the input for the random forest is extended with the room locations gained as part of the data preparation process explained in Section 4.1.2. The room locations are in a string format, therefore, one-hot encoding is performed in order to receive numeric values. The room locations are considered as part of the input because it is assumed that these locations can also be provided from other sensors as part of a data fusion activity recognition solution. In addition, instead of using a data window with approximately 240 samples, each data window is filtered out of the duplicates, thus, the final data windows carry only a single sample.

Due to the imbalance data set, while building the random forest model, first oversampling is performed on the final data set in order to increase the number of sliding windows representing the minority classes. Given that the data windows carry exactly one sample, the training time of the random forest after oversampling is still significantly fast.

The main focus is utilizing the random forest model to discriminate between the chosen set of activities highlighted in Figure 17, while presented with only one part of the data set of a single person. For the training process approximately 70% of the sliding windows are selected, while the remaining 30% are used for testing the built classifier. The train test split is defined in a way that for each activity a specific splitting time point is chosen, thus, all of the preceding data windows are part of the training process, while everything after is part of the test process. The splitting point in time for each activity differs because there are 9 different activities, and some of them occur earlier in the time range of one week, while others later in the same week.

Furthermore, during the training process, additional split on the training data is performed in order to define evaluation process during which the random forest will be tuned. The evaluation process includes 80% of the sliding windows present in the training set, while the remaining 20% are used as an evaluation set. The chosen evaluation measure is accuracy, even though some of the presented articles in Section 3 propose working with F-measure. Nevertheless, the main reason why accuracy is chosen as a evaluation measure for the purposes of this thesis is due to the performed oversampling. By including the oversampling strategy as part of the training process the limitations of the imbalanced dataset are partially mitigated. The chosen parameters after the evaluation are presented in Table 2. The maximum depth parameter is evaluated using the values from 12 to 45, and the minimum sample split parameter is evaluated with the values from 15 to 46. The number of estimators (decision trees) in the random forest is fixed to 1000. The evaluation process itself is presented in Figure 25 where two different parameters are being tuned concurrently.

Table 2. The random forest selected parameters after the parameter tuning for maximal depth and minimum sample split is performed for both hand wearables.

Random forest parameters	**sensor c0(right hand)**	**sensor c1(left hand)**
n estimators	1000	1000
max depth	41	26
min sample split	33	16

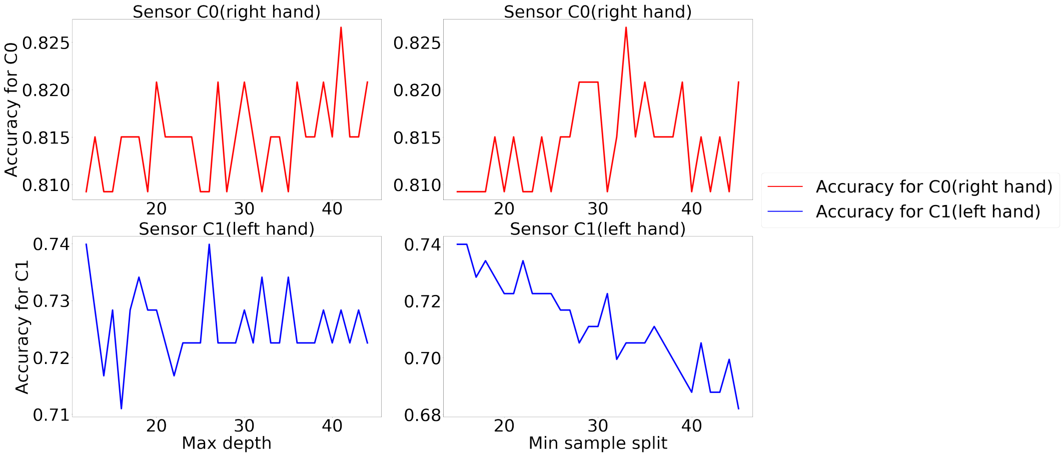


Figure 25. Two different parameters are concurrently tuned for the random forest classifier. The first row represents the tuning of the maximum depth parameter, and the minimum split size parameter when the data provided by the C0(right hand) sensor is used. The second row represents the same parameters when C1(left hand) sensor is used.

4.3 Long short-term memory(LSTM)

This section highlights the used approach for building and evaluation of the long short-term memory neural network. Similarly as for the random forest, there are additional processes such as visualization, time segmentation and hyper-parameter tuning aiding the development of the LSTM architecture. By following the summary of the presented related work in Section 3 it can be concluded that the Neural Networks (CNNs and LSTMs) usually do not need additional features to capture the dependencies present in the accelerometer data. In addition, following the results presented in Section 3 both, the CNNs and LSTMs achieve great classification results, and the two of them are suitable to address problems that arise in the HAR task. Nevertheless, to the author's best knowledge there are much more articles relating to CNNs than to LSTMs regarding accelerometer data. Therefore, the main reasons why LSTMs are chosen over CNNs is that they provide satisfactory results, and are not fully covered and investigated when applied on accelerometer data.

Furthermore, because the main constraint related to feature extraction present in the traditional ML models is alleviated, in this part a greater focus is given in choosing the suitable architecture followed by extensive hyperparameter tuning. In continuation to the previous, the LSTM model is built on the raw accelerometer data, and later evaluated. The overall architecture of this component is part of the Appendix.

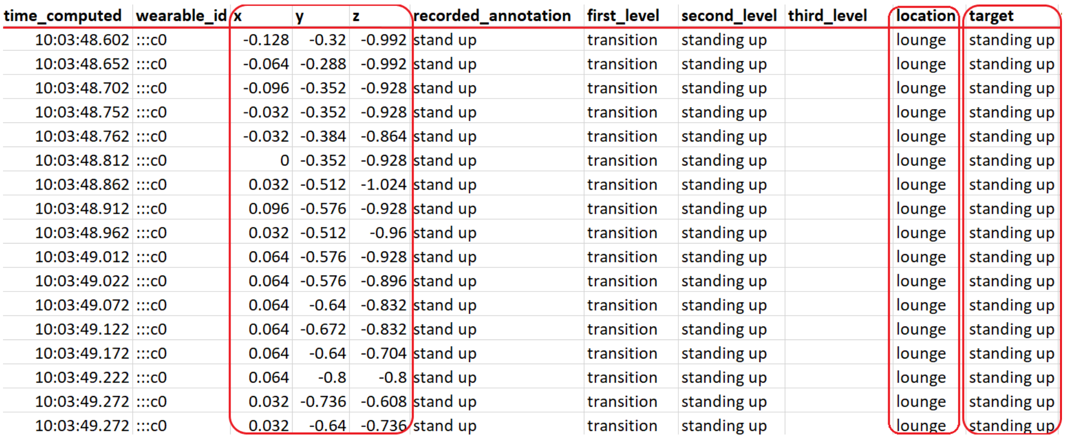
The subsections that follow expand on the previously mentioned topics in greater details.

4.3.1 Data preparation for LSTM

Before the raw accelerometer data can be fed in the LSTM neural network, initially a specific pre-processing should occur on the raw accelerometer data in order to define the suitable input. The data pre-processing can be split in three steps.

The first step is to get all of the candidate data segmented in windows with equal lengths of 10 seconds, and exactly 240 samples. The input of the LSTM has to have fixed-length sequences as training data, therefore, every time window that has less than 240 samples is extended to exactly 240 samples. The extension is done by duplicating randomly chosen samples from the same window and appending these at the end of the time window. In the end of this step there will be multiple segments with equal size of 240 samples.

The second step is to define the input for the LSTM network. The raw accelerometer readings for the X, Y and Z axes and the room location is used as input for the network, while the activities represent the target variable as shown in Figure 26. It is assumed that the room locations will be provided from other sensors when the accelerometers will be part of a data fusion activity recognition solution. The LSTM only accepts numeric values, therefore, one-hot encoding is performed on the room location values and the



time_computed	wearable_id	x	y	z	recorded_annotation	first_level	second_level	third_level	location	target
10:03:48.602	:::c0	-0.128	-0.32	-0.992	stand up	transition	standing up		lounge	standing up
10:03:48.652	:::c0	-0.064	-0.288	-0.992	stand up	transition	standing up		lounge	standing up
10:03:48.702	:::c0	-0.096	-0.352	-0.928	stand up	transition	standing up		lounge	standing up
10:03:48.752	:::c0	-0.032	-0.352	-0.928	stand up	transition	standing up		lounge	standing up
10:03:48.762	:::c0	-0.032	-0.384	-0.864	stand up	transition	standing up		lounge	standing up
10:03:48.812	:::c0	0	-0.352	-0.928	stand up	transition	standing up		lounge	standing up
10:03:48.862	:::c0	0.032	-0.512	-1.024	stand up	transition	standing up		lounge	standing up
10:03:48.912	:::c0	0.096	-0.576	-0.928	stand up	transition	standing up		lounge	standing up
10:03:48.962	:::c0	0.032	-0.512	-0.96	stand up	transition	standing up		lounge	standing up
10:03:49.012	:::c0	0.064	-0.576	-0.928	stand up	transition	standing up		lounge	standing up
10:03:49.022	:::c0	0.064	-0.576	-0.896	stand up	transition	standing up		lounge	standing up
10:03:49.072	:::c0	0.064	-0.64	-0.832	stand up	transition	standing up		lounge	standing up
10:03:49.122	:::c0	0.064	-0.672	-0.832	stand up	transition	standing up		lounge	standing up
10:03:49.172	:::c0	0.064	-0.64	-0.704	stand up	transition	standing up		lounge	standing up
10:03:49.222	:::c0	0.064	-0.8	-0.8	stand up	transition	standing up		lounge	standing up
10:03:49.272	:::c0	0.032	-0.736	-0.608	stand up	transition	standing up		lounge	standing up
10:03:49.272	:::c0	0.032	-0.64	-0.736	stand up	transition	standing up		lounge	standing up

Figure 26. The columns used for defining the input for the LSTM network. The input is consisted of the accelerometer readings for the X, Y and Z axes, the room location and the target activities. Furthermore, the room location and target activities need to be converted to integer values using one-hot encoding.

target variable values. At the end of this step, the formatted data for the LSTM will be placed in a three dimensional input array carrying the information for each segment, the samples per segment, and the expected input features.

The third step is splitting the data in train and test sets following the same distribution as for the random forest (70% train, 30% test). The segmentation process includes activity specific time splits because the distribution over the entire week varies for each activity.

4.3.2 Long short-term memory architecture

The initial architecture of the LSTM neural network contains 2 LSTM layers (stacked on each other) and 2 fully-connected layers with 64 neurons each. Greater details about the LSTM are given in Section 2.5.2. All of the neurons in the fully-connected layer incorporate the ReLU activation function. The rectified linear is an activation function defined as the positive part of its argument given with the following expression:

$$f(x) = x^{+} = max(0, x) \tag{10}$$

The chosen loss function in the network is the cross-entropy. In general, as presented in the study by John E. Shore et al. [37] the cross-entropy quantifies the difference between the true probability distribution and the predicted probability distribution. In other words, in the case of neural networks, the true probability distribution is given with the target labels, while the predicted distribution is the actual prediction of the neural network. Therefore, during the training process of the network the cross-entropy loss needs to be minimized. Given that the training process is continuous and repetitive, the need for minimization of the cross-entropy loss forces constant re-adjusting of the network's weights.

Furthermore, the imposed regularization of the loss function in the network is the L2 regularization (Ridge regularization). The L2 regularization as presented in study by Andrew Y. Ng [38] forces all of the weights present in the network to be as small as possible without making them zero. Additionally, it has non sparse solution. There are two main reasons why L2 is chosen over L1(Lasso regularization). First, if the output variable represents a combination of all of the input features it gives better predictions when compared to L1 [38]. Second, the L2 regularization has the ability to learn complex data patterns such those present in the accelerometer data [38].

The Adam optimizer is chosen for the minimization of the network's loss. This optimizer is chosen over the classical stochastic gradient descent procedure (SGD) mainly because of its unique approach of maintaining the learning rate in a neural network. More specifically, as presented in the study by Diederik P. Kingma [39] the Adam optimizer maintains a unique learning rate for each weight in the network, thus, updating the weights according their specific learning rates. On the contrary, the SGD procedure maintains a single learning rate for the network's weights, therefore, all of the weights are updated following the same learning rate.

4.3.3 Parameter tuning of the LSTM network

The LSTM network's architecture can be modified significantly and the discriminatory power can vary accordingly. Finding the optimal set of hyper-parameters is always a difficult challenge, especially when there are multiple options. Nevertheless, in order to increase the overall performance, hyper-parameter tuning is necessary. Therefore, a systematic approach is defined exploring the effect on the predictive power of the network of four different hyper-parameters such as the number of epochs, batch size, number of hidden units and number of hidden layers. The approach is presented below.

Following the data preparation from Section 4.3.1, the final data is already split into training and test data sets. Furthermore, the training data is additionally split into training and evaluation data sets. Each of the aforementioned hyper-parameters are evaluated in isolation while repeating the evaluation in five runs. The evaluation of each hyper-parameter using the same configuration occurs five times because the random initial conditions of the network may significantly affect the final results.

The first hyper-parameter that is evaluated is the number of epochs. The same configuration is run five times and the number of epochs is up to 550. The train and evaluation loss values when the data from the c0 (right hand) sensor is used are presented in Figure 27. According to the evaluation after epoch 150, the evaluation (test) loss starts again to increase. On the contrary, the training loss continues to decrease. Additionally, after this epoch, the test accuracy is steady, while the train accuracy continues with its

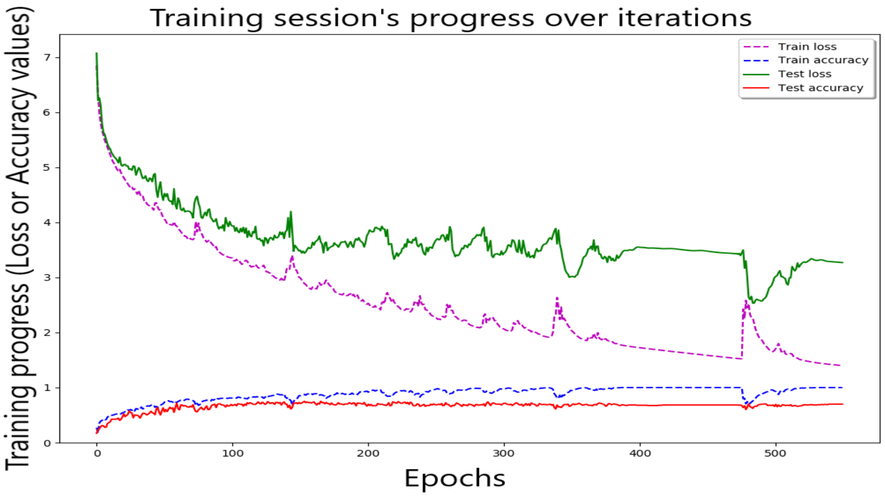


Figure 27. The train and evaluation loss of the network for different epoch values when the data from the c0 (right hand) sensor is used.

Table 3. Selected hyper-parameters after the hyper-parameter tuning of the LSTM network.

LSTM hyper-parameters	**sensor c0(right hand)**	**sensor c1(left hand)**
epochs	150	150
number of hidden units	64	64
number of hidden layers	2	3
batch size	180	90

increasing trend. Epoch 150 is stored as an optimal value for the epoch hyper-parameter, because, soon after this epoch the network starts to overfit. The LSTM built using the c1 (left hand) sensor data reports similar results.

The next three hyper-parameters, the batch size, the number of hidden units and the number of hidden layers are evaluated using the same approach as explained for the number of epochs.

The hyper-parameter evaluation highlights that increasing the number of hidden units and the number of hidden layers usually leads to higher loss regardless of the position of the sensor. The optimal number of hidden units for the LSTM using the data from the c0 (right hand) sensor is stored at 64, while only two hidden layers are sufficient as presented in Figure 28, respectively. The optimal number of hidden units for the LSTM using the data from the c1 (left hand) sensor is 64, and the number of layers is set to 3. During the evaluation process of the LSTM networks the number of hidden units is selected from the following list: 64, 128, 256, and the number of hidden layers varies

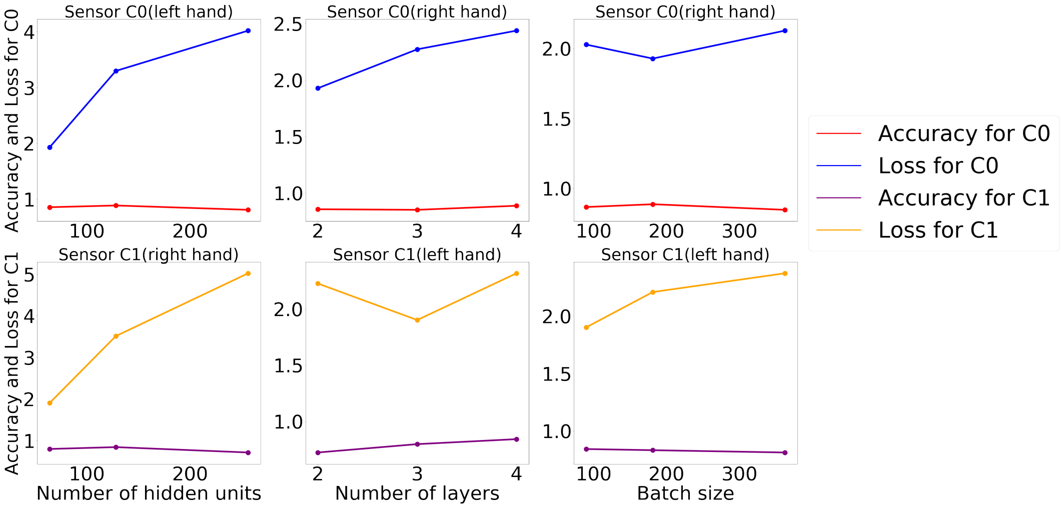


Figure 28. The first row represents the hyper-parameter tuning for the number of hidden units, the number of hidden layers and the batch size when the LSTM model is trained using the data provided by the right hand sensor(c0). Additionally, the second row represents the same hyper-parameter tunning when the model is trained using the data provided by the left hand (c1) sensor.

between 2, 3 and 4.

The evaluation highlights that significantly small values for the batch size up to 100 impacted the performance of the LSTM network using the data from the c0 (right hand) sensor, thus small batch sizes lead to higher loss. Furthermore, large values for the batch size also lead to higher loss. Therefore, the optimal batch size according to the experiments is 180 when the LSTM model is trained using the data provided by the right hand wearable (c0 sensor) as presented in Figure 28. On the contrary, the batch size of 90 is chosen when the LSTM model is trained using the data provided by the left hand wearable (c1 sensor) as provided in Figure 28. The evaluation process for the batch size include values such as 90, 180 and 360. In conclusion, the values of the hyper-parameter tuning that significantly minimize the loss function are presented in Table 3.

The main drawback in the performed work for building the LSTM network is that the effect of other hyper-parameters is not evaluated. Further evaluation may be performed on the dropout rate, the type of optimization algorithm, the regularization of the loss function, etc. Nevertheless, this may take significant part of a future work related to these topic.

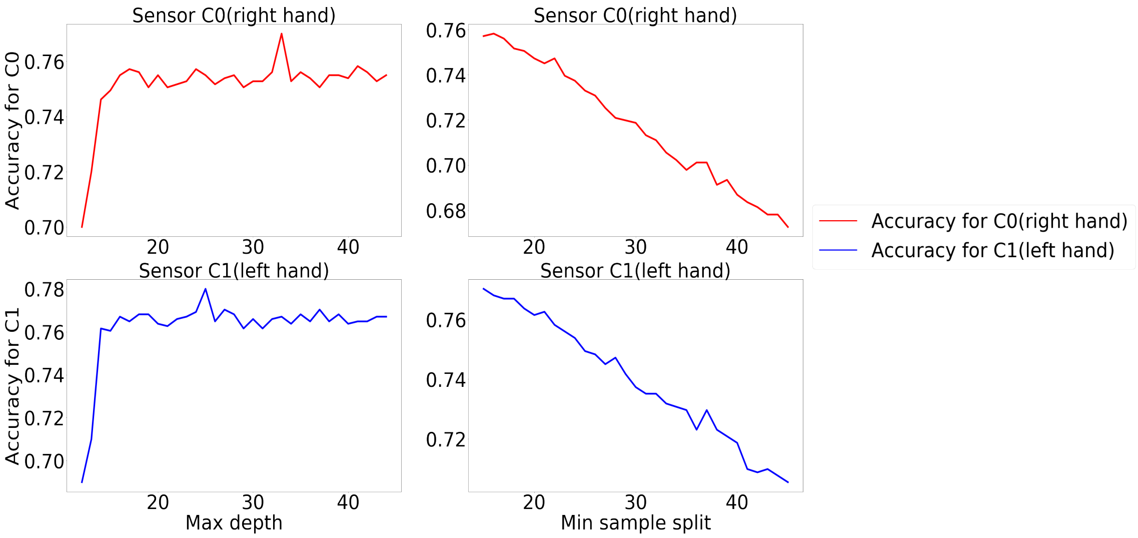


Figure 29. Random forest parameter tuning of the maximal depth and minimum sample split parameters when the *unknown* activity class is included.

Table 4. Random forest chosen parameters when the set of target classes is extended with the *unknown* activity class.

Random forest parameters	sensor c0(right hand)	sensor c1(left hand)
n estimators	500	500
max depth	34	26
min sample split	16	16

4.4 Activity recognition with *unknown* activities

The final part of the pipeline utilizes the defined processes for building models for recognition of activities in the unannotated SPHERE's data segments. The main focus is to introduce the unknown activity class as part of the existing processes for building models for recognition of activities. The new models need to classify whether the given data is part of the activities presented in Figure 17, or represents some unknown activity. Furthermore, all of the data labeled as an *unknown* activity represents activities that are not included in Figure 17. The goal is to provide models for annotation of activities in the SPHERE's unannotated data segments.

4.4.1 Random forest with the *unknown* activity class

For the training purposes of the models, the data that is not part of the activities presented in Figure 17 is annotated as unknown. Following the same principles and techniques described in Section 4.2, the data is split for training and testing, for both random forest classifiers, where the former uses the data provided by the right hand sensor, while the later is trained on the data provided by the left hand sensor. In addition, the test data that is given to the models is never present in the training process of the models.

Furthermore, the models are additionally evaluated using the same evaluation techniques for parameter tuning as in Section 4.2, respectively. The maximum depth parameter is evaluated in the range of values between 12 and 45. The minimum sample split parameter is evaluated using the values between 15 and 46. The overall tuning process for both parameters is shown in Figure 29. Additionally, the final parameter values that are chosen after the parameter tuning are presented in Table 4.

4.4.2 LSTM with the *unknown* activity class

For the training purposes of the LSTM models based on the data provided by the hand wearables (c0 and c1) the same principles and techniques described in Section 4.3 are applied.

The two derived models are additionally evaluated using the evaluation techniques for hyper-parameter tuning, respectively. In a continuation, the hyper-parameter tuning process of the created LSTM models is given in Figures 30 and 31 while the selected hyper-parameters for the two LSTMs are presented in Table 5.

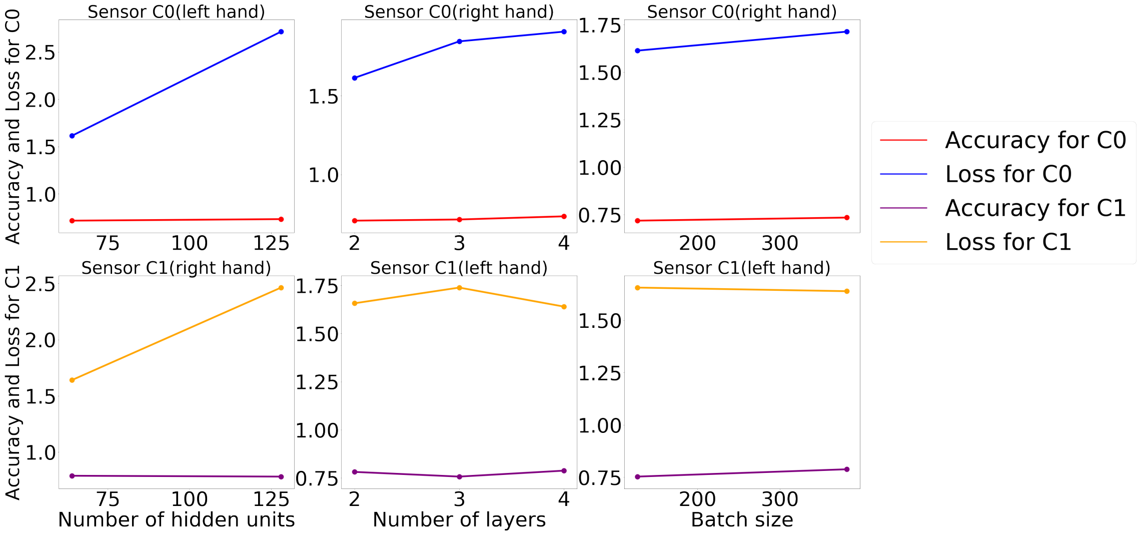


Figure 30. The first row represents the hyper-parameter tuning for the number of hidden units, the number of hidden layers and the batch size when the LSTM model is trained using the extended data with the *unknown* activity class provided by the right hand sensor(c0). Additionally, the second row represents the same hyper-parameter tunning when the model is trained using the extended data with the *unknown* class provided by the left hand (c1) sensor.

Table 5. LSTM chosen hyper-parameters when the set of target classes is extended with the *unknown* activity class.

LSTM hyper-parameters	**sensor c0(right hand)**	**sensor c1(left hand)**
epochs	260	260
number of hidden units	64	64
number of hidden layers	2	4
batch size	180	360

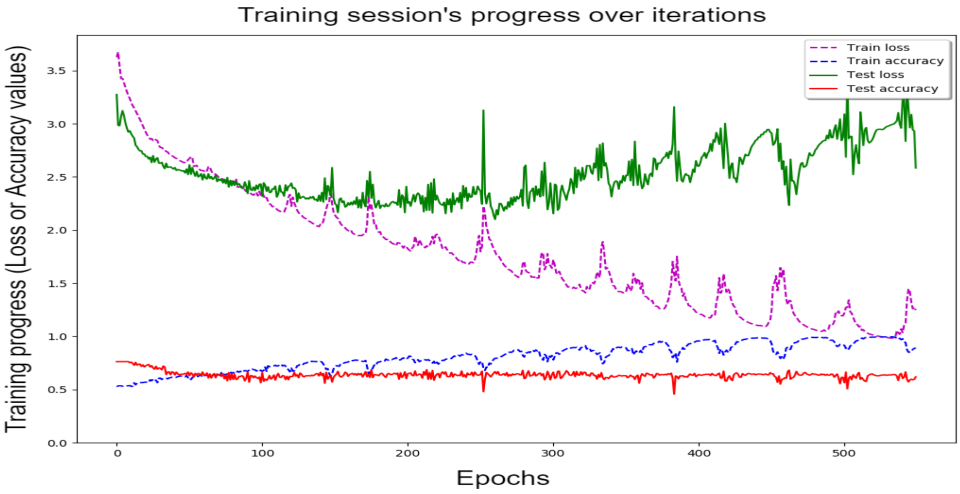


Figure 31. LSTM hyper-parameter tuning for number of epochs. The 260 epoch is chosen because there is no decrease of the test loss as the number of epochs increases, and additionally the test accuracy remains the same. Additionally, after the 260 epoch there is clear sign of overfitting due to the continuous drop of the train loss, and increase of the train accuracy.

5 Results

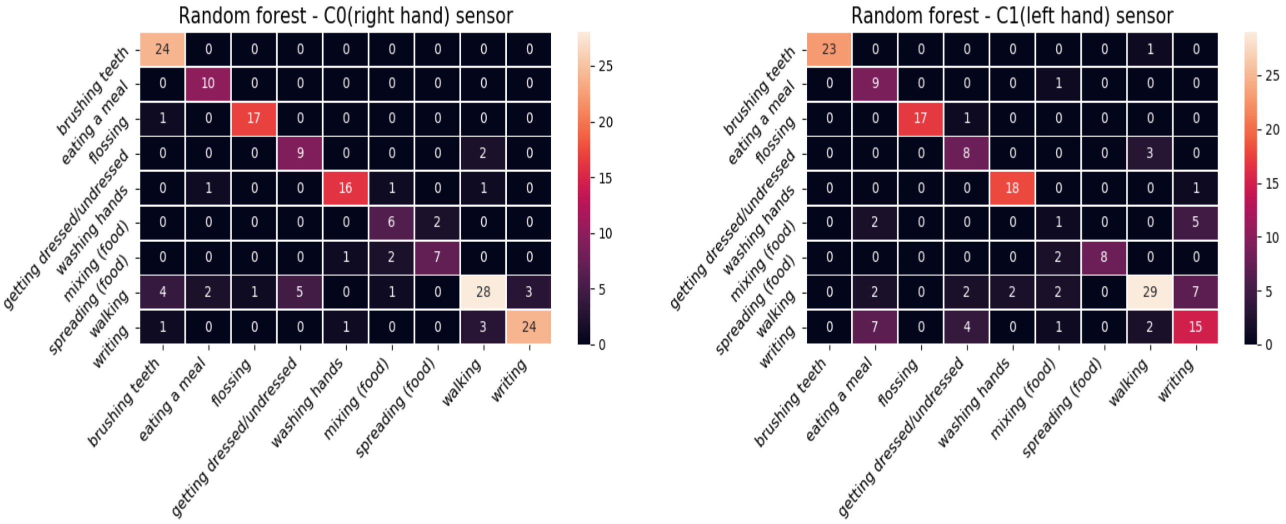
This section highlights the main findings when the built models are applied on test data. More specifically, an overview of the test results is given for the models (random forest and LSTM) that discriminate among the *known* activities provided in Figure 17 followed by the models that use the extended data with the *unknown* class. From this point forward the time windows used for testing purposes are referred as test instances, thus, each instance consists many accelerometer readings.

5.1 Random forest results

The random forest is part of the traditional machine learning approach where the most important prerequisites are the time-series segmentation and feature engineering. Based on these techniques the random forest increases its ability for discrimination between activities.

Following the confusion matrices presented in Figure 32 it can be concluded that the random forest model that uses the accelerometer records generated from the wearable attached on the right hand provides better prediction results than the model using the data records from the left hand wearable. Furthermore, the activities that can be discriminated without any classification mistakes include *brushing teeth* and *eating a meal*. Additionally, the rest of the activities of daily living (ADL) have only few classification mistakes. This points out that the random forest classifier using the data from the right hand sensor is able to discriminate between similar activities of daily living such as *brushing teeth* and *flossing*, between different activities of daily living such as *eating a meal*, *writing* and *getting dressing/undressed*. However, the activity *walking* has 16 misclassified instances distributed among activities such *brushing teeth*, *getting dressing/undressed* and *writing*. The misclassification highlights several different possibilities. First, for these four activities the current set of features may not be the best option. Second, additional features may be necessary that can distinguish walking from the rest. Third, there might not be any existing approach for defining a greater distinction between them given that activities such as *brushing teeth* and *getting dressing/undressed* can also be performed while executing the activity *walking*. And last, the training data might not be sufficient for *walking*.

In conclusion, given the SPHERE's real world environment settings, the custom adaption of data annotations, and additionally the 9 different activity classes, the scored accuracy of approximately 81% using the random forest model represents a satisfactory achievement. Furthermore, the received results point out that the model based on the right hand sensor slightly outperforms the model based on the left hand sensor. The main difference in the classification provided by the two models occurs when the activities *writing* and *mixing (food)* need to be recognized. As expected the right hand sensor is able to recognize the activities *writing* and *mixing (food)* because they are performed



(a)

(b)

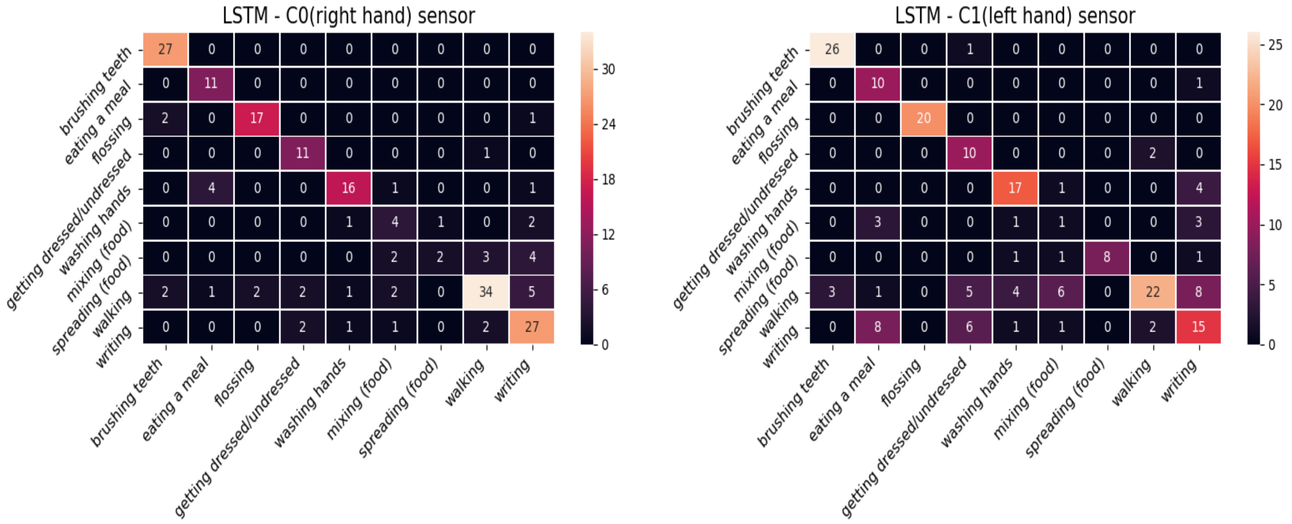
Figure 32. Confusion matrices for the random forest models when only the known activities presented in Figure 17 are part of the train and test data sets. (a) The random forest model trained and tested using the data from the wearable attached on the right hand (c0 sensor). (b) The random forest model trained and tested using the data from the wearable attached on the left hand (c1 sensor).

mostly by the dominant hand (in this example it is the right hand). In addition, another satisfactory result is that the constructed random forest models discriminate well on the pairs of similar activities (e.g. *flossing* and *brushing teeth*, *mixing (food)* and *spreading (food)*). The discrimination ability of the these models proves that using the feature extraction approach is indeed the right strategy when using a traditional machine learning model. The accuracy results of the random forest models are presented in Table 6.

5.2 LSTM results

The LSTM's most important prerequisites are the time-series segmentation and hyper-parameter tunning. The LSTM's ability for discrimination increases when the proper hyper-parameters are selected along with the correct data segmentation strategy.

The confusion matrices presented in Figure 33 highlight that the LSTM model built using the data from the right hand wearable provides more accurate results when compared to the LSTM built on the data from the left hand wearable. Additionally, it can be seen in Figure 33 that a similar group of activities can be discriminated with only few classification mistakes. The LSTM models provide great distinction between the pairs of similar activities such as *brushing teeth* and *flossing*, and between totally different



(a)

(b)

Figure 33. Confusion matrices for the LSTM models when only the known activities presented in Figure 17 are part of the train and test data sets. (a) The LSTM model trained and tested using the data from the wearable attached on the right hand (c0 sensor). (b) The LSTM model trained and tested using the data from the wearable attached on the left hand (c1 sensor).

activities such as *eating a meal* and *getting dressed/undressed*. The main shortcoming of the LSTM model built using the data from the right hand sensor represents the missclassifcation of the activity *spreading (food)*, while the LSTM model built using the data from the left hand sensor confirms the random forest missclassifcation results of the activities *writing* and *mixing (food)*. Also, the activity *walking* has a bigger number of missclassified instances as shown in Figure 33. Both of the LSTM models fail to provide satisfactory difference between *walking* and the activities of daily living similarly as in the traditional ML approach. One possibility is that the provided data for the activity *walking* is not enough given that both of the models (random forest and LSTM) highlight the inability to correctly classify $1/3$ of the *walking* test instances.

In summary, the LSTM models perform slightly worse than the random forest models, respectively. The main reason behind the shortcomings of the LSTM models can be detected in the current LSTM architecture, thus, additional hyper-parameter tuning may be necessary. In addition, the small training set can further decrease the classification capabilities of the LSTM. However, the presented results in Figure 33 highlight the LSTMs ability to provide almost the same classification capabilities as the random forest models by only using the raw accelerometer data as an input, thus, evading the need of the long and exhausting feature extraction process. The provided LSTM architecture suits as a baseline for the architecture of the models used in the activity recognition task when the *unknown* class is present. The accuracy results of the LSTM models are presented in Table 6.

5.3 Activity recognition when the *unknown* activity class is present

The HAR pipeline provides models which have the ability for recognition of activities in the unlabeled segments of the SPHERE's accelerometer data. Initially, these models recognize whether the specific accelerometer segment carries readings which belong to a *known* activity presented in Figure 17, or an *unknown* activity. Furthermore, the accelerometer data from each segment which is classified as *known* activity is recognized as one of the activities presented in Figure 17.

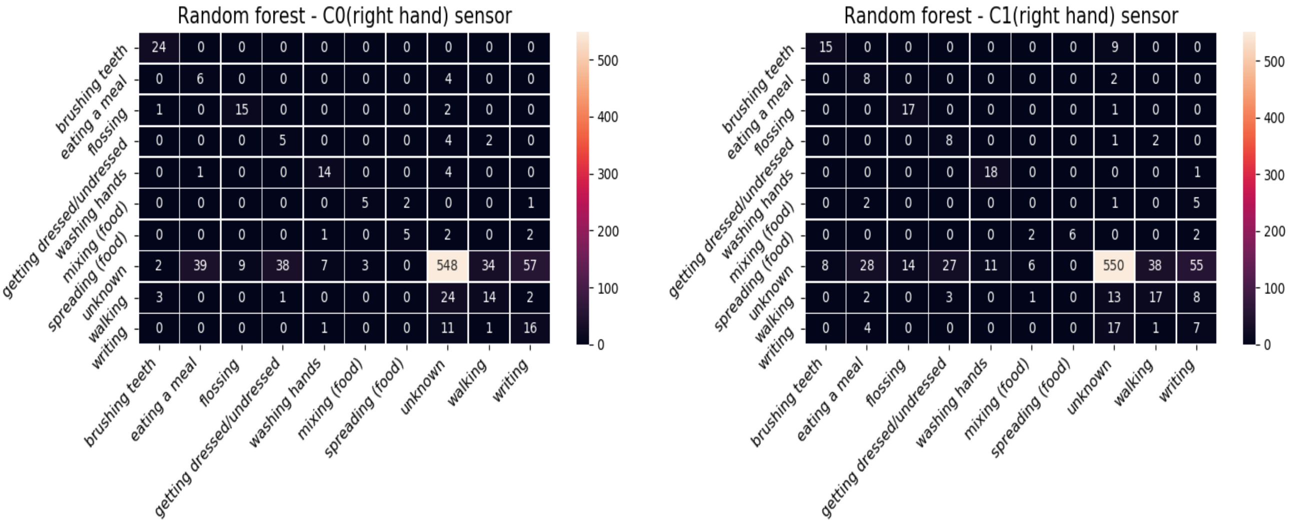
However, the proposed strategy for building the activity recognition models in Section 4.4 provides solution closely coupled with the models ability to discriminate between the *unknown* majority class and the *known* minority classes. Therefore, the results are expected to be biased towards the majority class. Nevertheless, there are many possible alternative solutions and many possible variations of the proposed solution. The test results of the models for recognition of activities when the *unknown* class is included are presented below.

5.3.1 Results of the random forest models when the *unknown* activity class is present

The confusion matrices received after using the random forest model for both of the hand sensors presented in Figure 34 provide the classification results between the *known* activities and the *unknown* activities.

The testing process is extended with the *unknown* activities and all of the instances are labeled as *unknown*. From the received results in Figure 34 it can be concluded that the random forest models are incapable of classifying all of the instances that carry records of the *known* activities correctly. The confusion matrix in Figure 34 derived from the right hand sensor points out that all of the activities except *brushing teeth*, and *mixing (food)* have part of their instances classified as *unknown*. The biggest discrimination problem occurs for the activity *walking*. Nevertheless, the number of instances that are actually *known* activities and are classified correctly is significantly higher when compared to the number of instances that are *known* activities but are classified as *unknown* as shown in Figure 35 given that there are 650 test instances of the *unknown* activity class.

There are also *unknown* activities which are classified as a *known* activity. Most of the missclassified *unknown* test instances are distributed among the activities such as *writing*, *getting dressed/undressed*, *walking* and *eating a meal*. Again, the number of correctly classified *unknown* activities is significantly higher than the *unknown* activities which are classified as *known*.



(a) (b)

Figure 34. Confusion matrices for the random forest models when the list of target activities presented in Figure 17 is extended with the *unknown* activity class as part of the train and test data sets. (a) The random forest model trained and tested using the data from the wearable attached on the right hand (c0 sensor). (b) The random forest model trained and tested using the data from the wearable attached on the left hand (c1 sensor).

The confusion matrices in Figure 34 point out that the random forest model that uses the data provided by the right hand wearable sensor slightly outperforms the other model which is based on the left hand wearable. Additionally, the high number of *unknown* activities classified as *known* is mainly due to the possibility of some wrong annotations in the initial data set. On the contrary, the main reason why the *known* activities are classified as *unknown* is due to the ratio between the *known* and *unknown* instances of $1 : 8$ in the training data set, or in total, approximately 90.000 and 700.000 accelerometer readings, respectively. However, the ability of the models to recognize *known* activities, and additionally to classify them correctly as presented in Figure 35 is on a satisfactory level. The accuracy results of the random forest models that discriminate between *known* and *unknown* activities are presented in Table 6.

5.3.2 Results of the LSTM models when the *unknown* activity class is present

The presented results in Figure 36 point out that the discrimination between *known* and *unknown* labeled accelerometer data follows a similar classification distribution as for the classification results of the random forest models presented in Section 5.3.1. The main difference in the classification results between the two LSTM models (right and left hand-based) occurs with respect to the activities *writing* and *mixing (food)*.

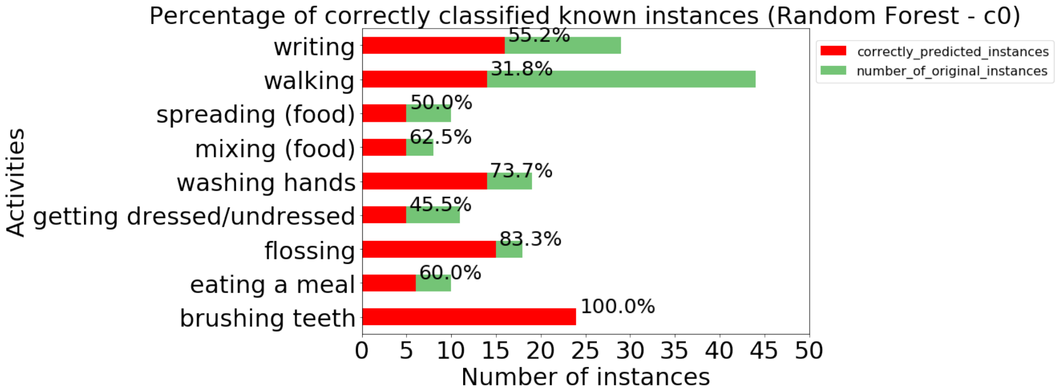
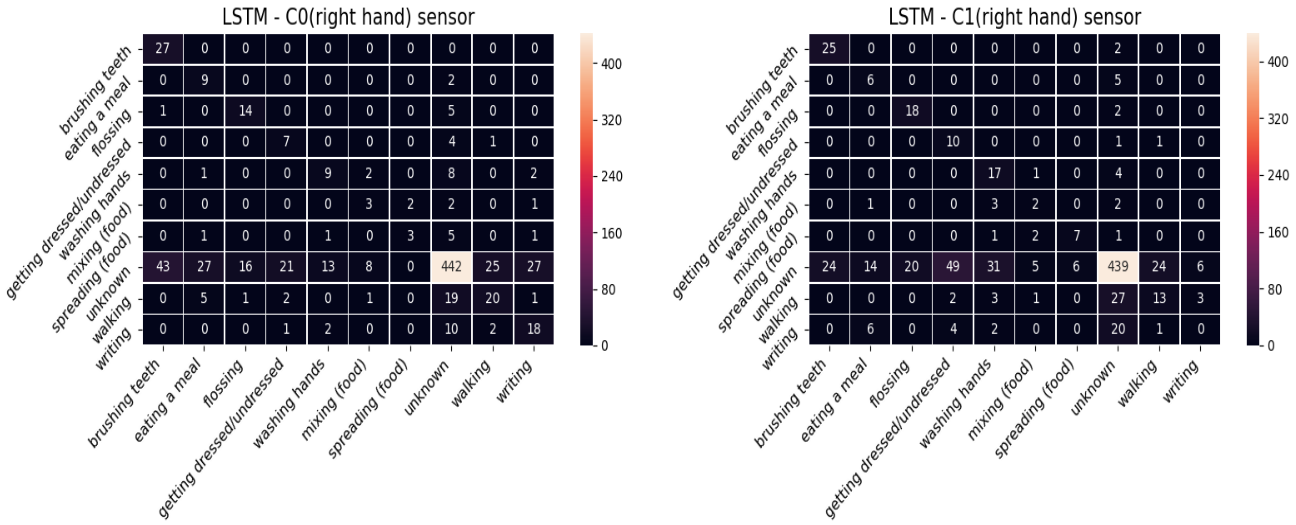


Figure 35. Distribution of correctly classified *known* activities when the *unknown* activity class is included in the training and testing processes for the random forest model using the data provided by the right hand sensor(c0).



	brushing teeth	eating a meal	flossing	getting dressed/undressed	washing hands	mixing (food)	spreading (food)	unknown	walking	writing
brushing teeth	27	0	0	0	0	0	0	0	0	0
eating a meal	0	9	0	0	0	0	0	2	0	0
flossing	1	0	14	0	0	0	0	5	0	0
getting dressed/undressed	0	0	0	7	0	0	0	4	1	0
washing hands	0	1	0	0	9	2	0	8	0	2
mixing (food)	0	0	0	0	0	3	2	2	0	1
spreading (food)	0	1	0	0	1	0	3	5	0	1
unknown	43	27	16	21	13	8	0	442	25	27
walking	0	5	1	2	0	1	0	19	20	1
writing	0	0	0	1	2	0	0	10	2	18

(a)

	brushing teeth	eating a meal	flossing	getting dressed/undressed	washing hands	mixing (food)	spreading (food)	unknown	walking	writing
brushing teeth	25	0	0	0	0	0	0	2	0	0
eating a meal	0	6	0	0	0	0	0	5	0	0
flossing	0	0	18	0	0	0	0	2	0	0
getting dressed/undressed	0	0	0	10	0	0	0	1	1	0
washing hands	0	0	0	0	17	1	0	4	0	0
mixing (food)	0	1	0	0	3	2	0	2	0	0
spreading (food)	0	0	0	0	1	2	7	1	0	0
unknown	24	14	20	49	31	5	6	439	24	6
walking	0	0	0	2	3	1	0	27	13	3
writing	0	6	0	4	2	0	0	20	1	0

(b)

Figure 36. Confusion matrices for the LSTM models when the list of target activities presented in Figure 17 is extended with the *unknown* activity class as part of the train and test data sets. (a) The LSTM model trained and tested using the data from the wearable attached on the right hand (c0 sensor). (b) The LSTM model trained and tested using the data from the wearable attached on the left hand (c1 sensor).

The LSTM confusion matrices in Figure 36 highlight the LSTM model based on the right hand sensor data as somewhat better compared to the left hand-based LSTM model. Similarly as in the case of the random forest models, the high number of *unknown* activities which are classified as *known* is mainly due to the possibility that

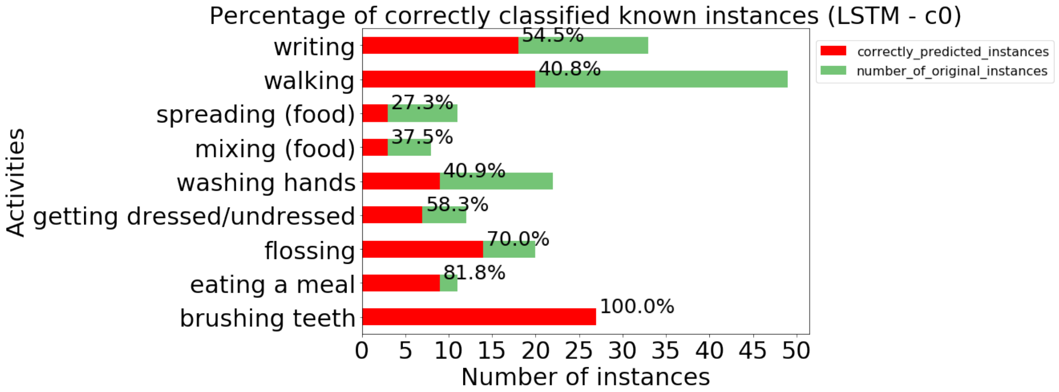


Figure 37. Distribution of correctly classified *known* activities when the *unknown* activity is included in the training and testing processes for the LSTM model using the data provided by the right hand sensor(c0).

Table 6. Summary of all of the used models and their accuracy performances.

Models	**right hand (c0)**	**left hand (c1)**
Random forest performing on 9 different classes	0.81	0.74
LSTM performing on 9 different classes	0.77	0.67
Random forest - second layer (9 different classes vs *unknown*)	0.71	0.70
LSTM - second layer (9 different classes vs *unknown*)	0.67	0.66

an underlying similar activity is performed or due to a wrongly adapted annotation. Additionally, the imbalanced data set has a significant role in the wrong predictions of the *known* activities as *unknown*. In summary, the models ability to recognize the *known* activities and discriminate between them correctly is on a satisfactory level. The proportion of correctly classified *known* activities when the *unknown* activity is included is shown in Figure 37. For the training purposes of the LSTM models approximately 90.000 *known* and 700.000 *unknown* accelerometer readings have been used, while 622 *unknown* instances are included in the testing process. The accuracy results of the LSTM models that discriminate between *known* and *unknown* activities are presented in Table 6.

5.4 Summary of the results

In summary, the random forest models slightly outperform the LSTM models when the discrimination is done only between the *known* activities presented in Figure 17. In addition, the random forest models also provide better activity recognition results when the *unknown* activity class is part of the training and the testing process. In Table 6 a summary of the accuracy results of the 8 aforementioned models is presented.

6 Conclusion

The presented HAR pipeline provides in-depth overview of the process of recognition of activities given the accelerometer data. The main challenges such as time series segmentation, feature engineering and building suitable models for recognition of activities are highlighted and described along with the necessary undertaken technical and theoretical decisions.

More specifically, a thorough overview of the SPHERE data is given in Section 2 together with the technical and theoretical limitations. Furthermore, the corresponding approach for alleviating these constraints and providing adapted data sets is presented in Section 4. In addition, as the initial limitations of the accelerometer data are surpassed, the next major steps include practical implementation of the time series segmentation and feature extraction by following the guidelines and best practises in Section 3.

Furthermore, two different types of machine learning models are presented in Section 4. In greater details one possible approach is proposed on how to train and evaluate the random forest models and the LSTM models, respectively. In total 8 different models are provided as part of this thesis, using accelerometer data received from two wearables attached on the person's wrists, respectively. The initially provided models are defined for discrimination between a group of *known* activities. Furthermore, the data is extended with an *unknown* activity, extending the possibility of the models to recognize known activities in unlabeled data segments. The provided results point out that the random forest models slightly outperform the LSTM models when the discrimination process is done only between the known activities. Also, the models using the data provided from the right hand sensor usually outperform the models using the data provided from the left hand sensor. Additionally, the random forest models outperform the LSTM models when the activity recognition task is extended with the *unknown* activity class. However, the LSTM models provide almost equal results as the random forest, thus, highlighting that the accelerometer data can also be used in its raw format excluding the long feature extraction process.

The constructed and easily adaptable HAR pipeline is the main outcome of this thesis along with the theoretical overview of the accelerometer based Human Activity Recognition.

In the end, the provided results support the two main hypothesis: the accelerometer data is suitable for recognition of activities when a proper data pre-processing strategy and feature extraction are employed, and also suitable as an input to a neural network (LSTM) in its raw format. Nevertheless, the proposed architecture, chosen strategies, chosen models and chosen activities can be altered and some potential shortcomings can be alleviated in future work.

7 Future work

The described approach in this thesis has a potential for different types of future work. The constructed HAR pipeline can be altered in different ways, thus, resulting in increased efficiency and accuracy.

The presented synchronization of the accelerometer data can be part of a further investigation. It may prove to be useful to define a strategy that can remove the one second gap between the actual record and the annotation. It is worth investigating because it may lead to increased accuracy during the recognition of activities.

Furthermore, the time segmentation strategy presented in this thesis is one among the many. Future work may also focus on defining time windows with different time length, while additionally allowing overlap between them. Such work will provide useful information about the advantages and shortcomings of the chosen segmentation strategies.

The chosen set of activities is not final and definite, thus, different activities can be added. Introducing new activities as part of a future work may prove to be important because it may highlight essential information for the feature extraction process. In continuation, it would push the research further if additional features are explored and tested using a different set of activities. Ultimately, it will be worth investigating the relation between specific features and specific activities.

Another possible future work can include a selection of different models used for activity recognition. One particular work can focus on building convolutional neural networks (CNNs) for recognition of activities on the SPHERE accelerometer data. Additionally, in another work a greater focus can be given on the hyper-parameter tuning of the LSTMs when working with accelerometer data. After an exhaustive hyper-parameter tuning the results may be used as a solid baseline when similar work has to be performed.

Overall, the presented solution tries to encapsulate as much as possible following the best practises and guidelines. The different configuration possibilities lead to a different implementation setup which can directly influence on the chosen evaluation measure and the performances of the pipeline. Therefore, the current HAR pipeline can serve as a baseline for many different types of future work.

8 Acknowledgments

I am particularly grateful for the assistance given by my supervisor Meelis Kull for his valuable and constructive suggestions throughout this research work. His willingness to assist me has been very much appreciated, while his criticisms and guidelines laid the foundation of this thesis.

References

[1] Elsts, Atis, et al. "A Guide to the SPHERE 100 Homes Study Dataset." *arXiv preprint arXiv:1805.11907* (2018).

[2] Avci, Akin, et al. "Activity recognition using inertial sensing for healthcare, wellbeing and sports applications: A survey." *23th International conference on architecture of computing systems 2010*. VDE, 2010.

[3] Chang, Keng-Hao, Mike Y. Chen, and John Canny. "Tracking free-weight exercises." *International Conference on Ubiquitous Computing*. Springer, Berlin, Heidelberg, 2007.

[4] Akl, Ahmad, Babak Taati, and Alex Mihailidis. "Autonomous unobtrusive detection of mild cognitive impairment in older adults." *IEEE transactions on biomedical engineering* 62.5 (2015): 1383-1394.

[5] Khusainov, Rinat, et al. "Real-time human ambulation, activity, and physiological monitoring: Taxonomy of issues, techniques, applications, challenges and limitations." *Sensors* 13.10 (2013): 12852-12902.

[6] Woznowski, Przemyslaw, et al. "SPHERE: A sensor platform for healthcare in a residential environment." *Designing, Developing, and Facilitating Smart Cities*. Springer, Cham, 2017. 315-333.

[7] Borio, Daniele. "Accelerometer signal features and classification algorithms for positioning applications." *Proceedings of the 2011 International Technical Meeting of The Institute of Navigation, San Diego, CA, USA*. 2011.

[8] Twomey, Niall, et al. "A comprehensive study of activity recognition using accelerometers." *Informatics*. Vol. 5. No. 2. Multidisciplinary Digital Publishing Institute, 2018.

[9] Kwapisz, Jennifer R., Gary M. Weiss, and Samuel A. Moore. "Activity recognition using cell phone accelerometers." *ACM SigKDD Explorations Newsletter* 12.2 (2011): 74-82.

[10] Williams, Ronald J., and David Zipser. "A learning algorithm for continually running fully recurrent neural networks." *Neural computation* 1.2 (1989): 270-280.

[11] Hochreiter, Sepp. "The vanishing gradient problem during learning recurrent neural nets and problem solutions." *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6.02 (1998): 107-116.

[12] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.

[13] Nissl, Norbert. "Analog accelerometer." U.S. Patent No. 4,226,120. 7 Oct. 1980.

[14] Tang, William C. "Digital capacitive accelerometer." U.S. Patent No. 5,447,068. 5 Sep. 1995.

[15] Devices, Analog. "Micropower, 3-axis, 2 g/4 g/8 g digital output mems accelerometer." *Analog Devices: Norwood, MA, USA* (2016).

[16] Fafoutis, Xenofon, et al. "Designing wearable sensing platforms for healthcare in a residential environment." *EAI Endorsed Transactions on Pervasive Health and Technology* 3.12 (2017).

[17] Micro, S. T. "LSM6DSM iNEMO Inertial Module." (2016).

[18] Flach, Peter. *Machine learning: the art and science of algorithms that make sense of data*. Cambridge University Press, 2012.

[19] Rencher, Alvin C. *Methods of multivariate analysis*. Vol. 492. John Wiley & Sons, 2003.

[20] James, John Francis, et al. "A student's guide to Fourier transforms with applications in physics and engineering." *Computers in Physics* 10.1 (1996): 47-47.

[21] Breiman, Leo. "Random forests." *Machine learning* 45.1 (2001): 5-32.

[22] Breiman, Leo, et al. "Classification and regression trees. Wadsworth Int." *Group* 37.15 (1984): 237-251.

[23] Haykin, Simon. *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

[24] Sak, Haşim, Andrew Senior, and Françoise Beaufays. "Long short-term memory recurrent neural network architectures for large scale acoustic modeling." *Fifteenth annual conference of the international speech communication association*. 2014.

[25] Huynh, Tâm, and Bernt Schiele. "Analyzing features for activity recognition." *Proceedings of the 2005 joint conference on Smart objects and ambient intelligence: innovative context-aware services: usages and technologies*. ACM, 2005.

[26] Bao, Ling, and Stephen S. Intille. "Activity recognition from user-annotated acceleration data." *International conference on pervasive computing*. Springer, Berlin, Heidelberg, 2004.

[27] Ravi, Nishkam, et al. "Activity recognition from accelerometer data." *Aaai*. Vol. 5. No. 2005. 2005.

[28] Ward, Jamie A., et al. "Activity recognition of assembly tasks using body-worn microphones and accelerometers." *IEEE transactions on pattern analysis and machine intelligence* 28.10 (2006): 1553-1567.

[29] Ha, Sojeong, and Seungjin Choi. "Convolutional neural networks for human activity recognition using multiple accelerometer and gyroscope sensors." *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2016.

[30] Guan, Yu, and Thomas Plötz. "Ensembles of deep lstm learners for activity recognition using wearables." *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1.2 (2017): 11.

[31] Ordóñez, Francisco, and Daniel Roggen. "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition." *Sensors* 16.1 (2016): 115.

[32] Banos, Oresti, et al. "Window size impact in human activity recognition." *Sensors* 14.4 (2014): 6474-6499.

[33] Bao, Chenglong, et al. "Dictionary learning for sparse coding: Algorithms and convergence analysis." *IEEE transactions on pattern analysis and machine intelligence* 38.7 (2016): 1356-1369.

[34] Xie, Lingyue, Han Zhang, and Feng Duan. "A feature extraction method based on dictionary learning for EEG." *2015 11th International Conference on Natural Computation (ICNC)*. IEEE, 2015.

[35] Gupta, Piyush, and Tim Dallas. "Feature selection and activity recognition system using a single triaxial accelerometer." *IEEE Transactions on Biomedical Engineering* 61.6 (2014): 1780-1786.

[36] Tonkin, Emma L., and Przemyslaw R. Woznowski. "Activities of Daily Living Ontology for Ubiquitous Systems." *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, 2018.

[37] Shore, John E., and Robert M. Gray. "Minimum cross-entropy pattern classification and cluster analysis." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1 (1982): 11-17.

[38] Ng, Andrew Y. "Feature selection, L 1 vs. L 2 regularization, and rotational invariance." *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004.

[39] Kingma, Diederik P., and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014).

[40] "Welcome to the MongoDB Docs." MongoDB Documentation, docs.mongodb.com/.

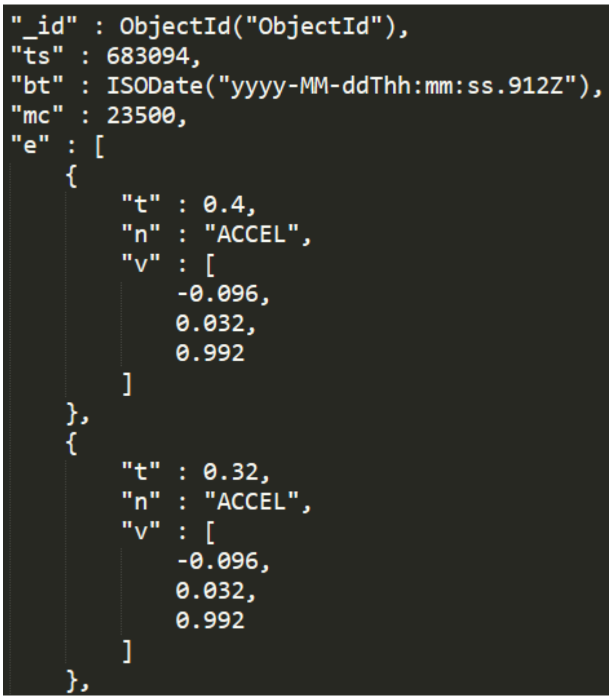
[41] "PyMongo 3.8.0 Documentation." PyMongo 3.8.0 Documentation - PyMongo 3.8.0 Documentation, api.mongodb.com/python/current/.

[42] "Python 3.6." DevDocs, devdocs.io/python 3.6/.

Appendix

I. SPHERE data packages

Part of one accelerometer package is given in Figure 38 where the main components are the bt field, and the e field. The bt field represents the absolute timestamp when this package is sent from the accelerometer, and the e field is consisted of six accelerometer readings. Each accelerometer reading has a t field which represents the relative timestamp against the bt recorded time, and a v array carrying the x, y and z values for the reading respectively. Further information for the entire document's structure is given in the study by Atis Elsts et al. [1].



```
"_id" : ObjectId("ObjectId"),
"ts" : 683094,
"bt" : ISODate("yyyy-MM-ddThh:mm:ss.912Z"),
"mc" : 23500,
"e" : [
    {
        "t" : 0.4,
        "n" : "ACCEL",
        "v" : [
            -0.096,
            0.032,
            0.992
        ]
    },
    {
        "t" : 0.32,
        "n" : "ACCEL",
        "v" : [
            -0.096,
            0.032,
            0.992
        ]
    },
```

Figure 38. Sample MongoDB document with 6 elements in array e, where each element is an acclerometer reading for the X, Y and Z axes. In addition the field bt includes the absolute time of recording of this 6 readings, while the relative field t shows the time difference between each of the readings in this document.

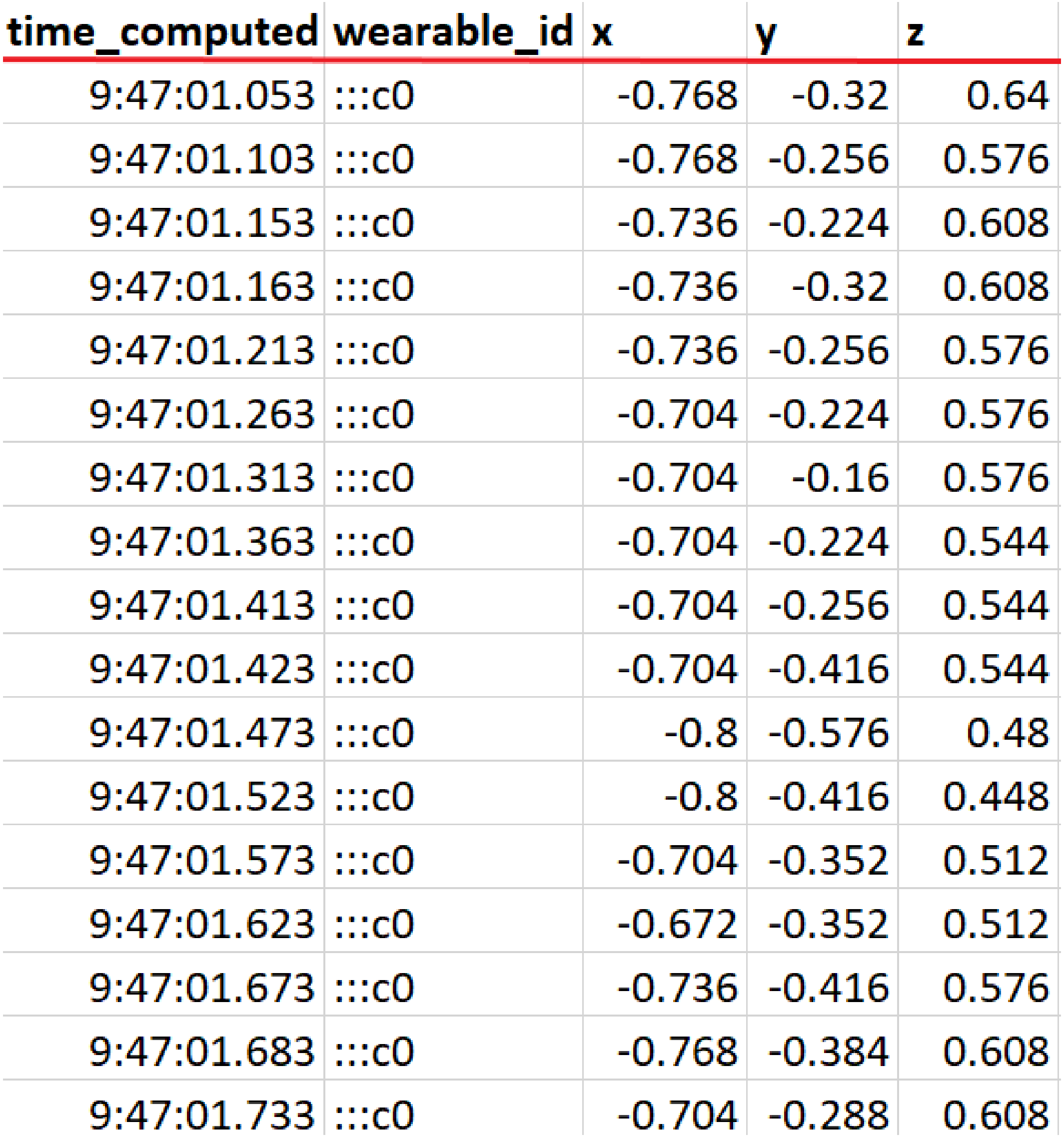
II. HAR Pipeline - technical details

Data storage and manipulation. The data that is recorded from the accelerometers has a size of approximately 20 GB for the period of one month. The data is stored in a .BSON format in a MongoDB database [40]. This format is the binary representation of the well-known JSON format. MongoDB Database is a NoSql database which operates with two basic structures, documents and collections. Each document is following a BSON format, thus, it is composed out of pairs of field and value. In each document there are exactly six accelerometer reading records for the X, Y and Z axes. A set of BSON documents represents a collection. To store the accelerometer readings in MongoDB, one collection of over 20 million documents is used.

Furthermore, a connection between the MongoDB and Python is needed in order to fetch the stored collections and manipulate with the documents within. This connection is enabled by PyMongo [41] which is a Python distribution containing all the necessary tools for proper communication between Python and MongoDB.

After the proper initialization of communication the data can be easily retrieved from MongoDB and used for further manipulations using Python 3.6 [42]. First, the manipulations are focused on combining the annotation files and accelerometer data retrieved from MongoDB. Second, the combined data sets are transformed accordingly for visualization, feature extraction and training and testing the models.

Generating the accelerometer data from database. The start of the process for producing raw accelerometer data files in a structured and easily reusable format is triggered after the finish of the synchronization process. The goal of this process is to transform all of the accelerometer readings associated with annotations. The transformation is needed because each document in the database has six readings for the X, Y and Z axes and each consecutive accelerometer record within the document has a fixed time difference as shown in Figure 38. The time offset is one of the following: 0, 0.05, 0.1, 0.15, 0.2 or 0.25. Additionally, there is only one absolute timestamp per document. The time difference is subtracted from the absolute timestamp of the document in order to compute the absolute timestamp for each of the six records, thus, there will not be always only one time difference between two consecutive readings. The specific structure of the documents makes the usage of the accelerometer readings difficult for additional needs such as concatenation with annotations, visualization, feature extraction, model building, etc. Therefore, in order to generate suitable files for the aforementioned needs each of the documents is transformed into six different samples. At the end of this process each sample carries a computed absolute timestamp, a name of the wearable that produced the record and the values for the X, Y, and Z axes as presented in Figure 39.



time_computed	wearable_id	x	y	z
9:47:01.053	:::c0	-0.768	-0.32	0.64
9:47:01.103	:::c0	-0.768	-0.256	0.576
9:47:01.153	:::c0	-0.736	-0.224	0.608
9:47:01.163	:::c0	-0.736	-0.32	0.608
9:47:01.213	:::c0	-0.736	-0.256	0.576
9:47:01.263	:::c0	-0.704	-0.224	0.576
9:47:01.313	:::c0	-0.704	-0.16	0.576
9:47:01.363	:::c0	-0.704	-0.224	0.544
9:47:01.413	:::c0	-0.704	-0.256	0.544
9:47:01.423	:::c0	-0.704	-0.416	0.544
9:47:01.473	:::c0	-0.8	-0.576	0.48
9:47:01.523	:::c0	-0.8	-0.416	0.448
9:47:01.573	:::c0	-0.704	-0.352	0.512
9:47:01.623	:::c0	-0.672	-0.352	0.512
9:47:01.673	:::c0	-0.736	-0.416	0.576
9:47:01.683	:::c0	-0.768	-0.384	0.608
9:47:01.733	:::c0	-0.704	-0.288	0.608

Figure 39. Raw accelerometer data in a structured format where each sample has its computed timestamp, the wearable id and one acceleration reading for the X, Y and Z axes.

III. HAR Pipeline - components

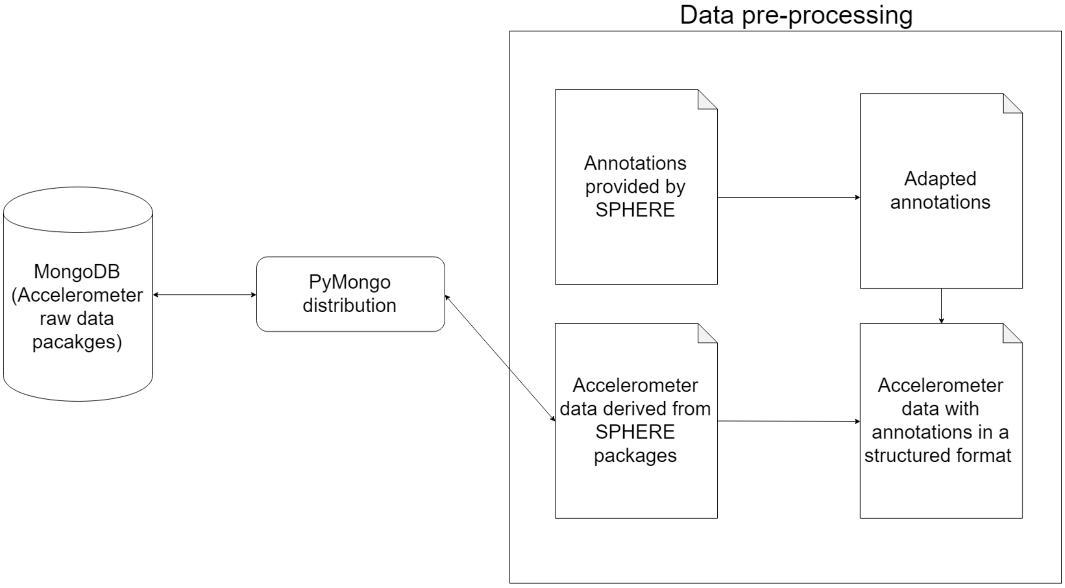

Data pre-processing
MongoDB (Accelerometer raw data pacakges)
PyMongo distribution
Annotations provided by SPHERE
Adapted annotations
Accelerometer data derived from SPHERE packages
Accelerometer data with annotations in a structured format

Figure 40. A broad overview of the system for data pre-processing. The initial leftmost side represents the database storage. The rightmost figure represents the starting unit of the HAR pipeline. Furthermore, the middle part is the established communication between the database and the HAR pipeline.

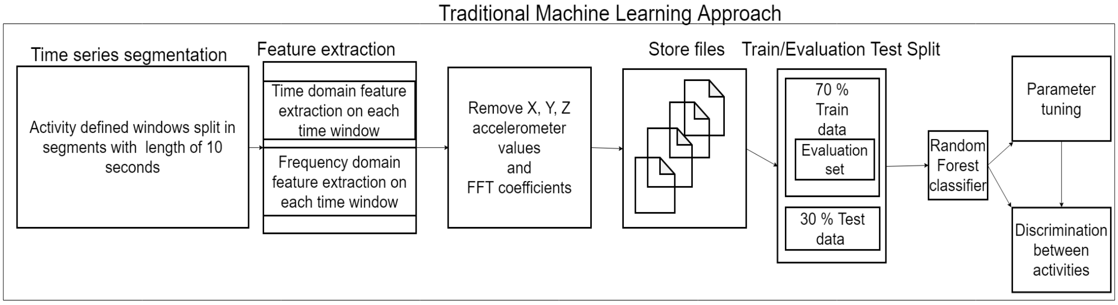

Traditional Machine Learning Approach
Time series segmentation
Activity defined windows split in segments with length of 10 seconds
Feature extraction
Time domain feature extraction on each time window
Frequency domain feature extraction on each time window
Remove X, Y, Z accelerometer values and FFT coefficients
Store files
Train/Evaluation Test Split
70 % Train data
Evaluation set
30 % Test data
Random Forest classifier
Parameter tuning
Discrimination between activities

Figure 41. A broad overview of the traditional ML approach.

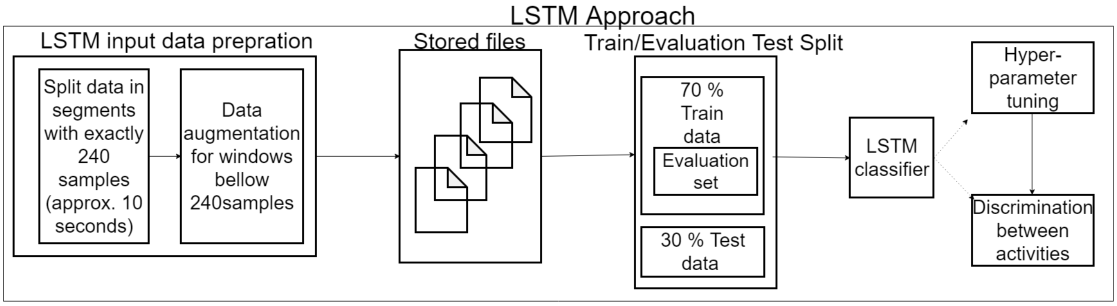

LSTM Approach
LSTM input data prepration
Split data in segments with exactly 240 samples (approx. 10 seconds)
Data augmentation for windows bellow 240samples
Stored files
Train/Evaluation Test Split
70 % Train data
Evaluation set
30 % Test data
LSTM classifier
Hyper-parameter tuning
Discrimination between activities

Figure 42. A broad overview of the LSTM approach.

IV. HAR Pipeline - code distribution

The python code for the HAR pipeline is provided as a supplementary zip along with the digital version of the thesis.

V. Glossary

SPHERE Sensor Platform for HEalthcare in a Residential Environment

HAR Human activity recognition

ML Machine Learning

LR Linear Regression

PCA Principal Component Analysis

FFT Fast Fourier Transform

RF Random Forest

LSTM Long Short-Term Memory

VI. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Hristijan Sardjoski**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Activity recognition using accelerometers,

   supervised by **Meelis Kull**.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Hristijan Sardjoski
Tartu, 16.05.2019