Bachelor thesis
Computer Science

Radboud University

Supervised Learning in Human Activity Recognition based on Multimodal Body Sensing

Author:
Michel Botros
s4259564
michel.botros@student.ru.nl

First supervisor/assessor:
Prof. Dr. Tom Heskes
t.heskes@science.ru.nl

Second assessor:
Prof. Dr. Ir. A.P. de Vries
A.deVries@cs.ru.nl

August 22, 2017

Abstract

The task of determining the activity a subject is performing at a specific time by using sensor data is called Human Activity Recognition. This work aims to utilize supervised learning techniques with data from different sensors placed on the body, providing details about all stages of the classification process. Both the more practical aspects from body sensing, as well as the more theoretical aspects of machine learning, in particular random forests, are discussed. Our main goal is a high rate of correct predictions, making Human Activity Recognition systems more useful for monitoring behavior, benefiting the health care sector and medical sciences. Afterwards we reflect on the classification process by looking for pairs of activities that are confused with each other more often. As a third aim, this thesis attempts to identify the important sensors, which should be used for a better prediction. We reached an accuracy score of 0.87. In the rest of our findings we present an overview, showing the degree of confusion between activities and a ranking for features, which translates back to a ranking of sensors.

Contents

Chapter 1

Introduction

In our society today there is a shift in the distribution of age due to declining fertility rates and an increase in the average life expectancy. Combine this with the fact that the world population is also growing and we can see why the health care sector is rapidly growing. In order to reduce strain on health care staff and costs, a smart system that can support and monitor elderly at home is desirable. Recognizing behavior of patients is crucial in this process.

The task of determining the activity a subject is performing at a specific time by using sensor data is called Human Activity Recognition (HAR). Early attempts to recognize human activity date back to the 80's. HAR has become a broad and active research area since hardware has become cheaper, smaller and faster. HAR has two main issues. The first one is choosing the right setup and sensors for a set of activities. The second issue is going from raw data to a prediction using machine learning techniques. HAR has a wide range of applications, particularly in medical science. Automated HAR is desired for researching effects of lifestyle on overall health by giving insights into the physiological activity of a subject and can also aid in diagnosing some diseases like sleep apnea or Parkinson's disease [19] or can be used to alarm medical personnel in case of abnormal activities like falling, strokes, seizures and heart attack or find causes of abnormal activity [2, 24].

In fields as the military and sports there exist a high interest in HAR systems. Safety and performance of the human body are critical. In both fields health conditions and activities could be monitored, benefiting the safety and performance of the soldier or athlete.

Human interaction with portable devices is growing rapidly. Nearly every adult carries some kind of mobile device through a large portion of the day. Most people have a smart phone and in some cases a smart watch. Due to advances in technology these mobile devices are equipped with a variety of sensors. Some examples of sensors that can be found in modern phones are an accelerometer, fingerprint sensor, barometer, gyroscope, geomagnetic sensor, hall sensor, heart rate sensor and a proximity sensor. The availabil-

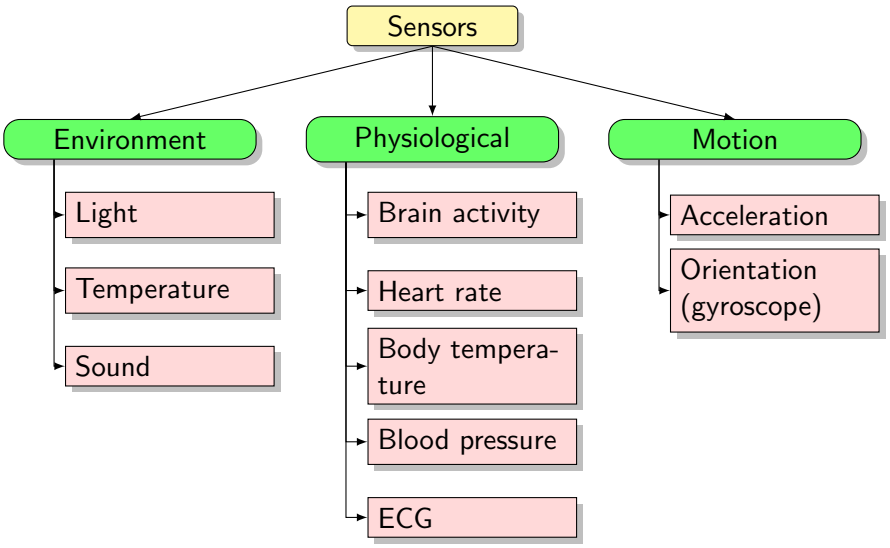

Figure 1.1: Taxonomy of sensors

ity of data provided by these sensors gives us an opportunity to monitor behavior of a user and his surroundings. Figure 1 shows a taxonomy of sensors [18].

One approach is placing sensors on the body of a subject. This is called body sensing. This has the advantage of collecting more specific data for areas of the body. Wearable sensors for example can be used to monitor physiological attributes like brain activity, heart rate, body temperature and blood pressure. Wearable sensors can also capture motion using accelerometers and gyroscopes placed on different body parts. Sensors can be placed in clothes, devices, glasses and shoes.

Another approach is based on external sensors, which focus on the environment of the subject. External sensors are not placed on the body, but are placed on points of interest. Points of interest depend on the analysis being done. Frequently used points include fridge, bed, shower, kitchen, hallway and living room. External sensors capture environment attributes such as light, temperature and sound. Especially camera's and microphones are used a lot. A downside to this approach is that the sensors have to be placed beforehand and subjects might be out a sight too much for useful analysis. Consider the situation of multiple people in a room. Wearable sensors might have the advantage since camera's might have difficulty with hindered sight. Another downside is complexity. Constant analysis of camera recording might cost to much resources, especially in real time applications. An advantage of external sensors is that interaction between subject and environment can be monitored. On camera there is a distinction between a fridge door and a ordinary door, while this distinction is hard to make

with wearable sensors. In some cases this distinction might be relevant, for example in research on eating habits. The external sensor approach has applications particularly in smart homes [22, 21, 15, 25].

In this research we are taking the body sensing approach. There is a wide variety of sensors to choose from (figure 1.1). For a specific set of activities, a specific set of attributes is relevant. Based on those attributes a setup can be chosen. Mostly accelerometers are used for motion in combination with gyroscopes. Environment attributes, such as light, temperature and sound might be relevant for some activities. They might for example be used to decide whether an activity is an outside activity or an inside activity. Physiological sensors can monitor brain activity or vital signs such as heart rate, body temperature and blood pressure, giving a indication on the physiological state of the subject. When motion of two activities are similar, think of jogging and sprinting, these can make a distinction based on the intensity of an activity by monitoring heart rate. In the next chapter I will give more details about the sensor setup.

Another issue [18] in choosing wearable sensors is obtrusiveness. Wearable sensors each have a level of obtrusiveness. Wearing sensors can hinder the subject while performing activities. Some sensors might need wires or might be heavy, resulting in high obtrusiveness. Ideally, the subject doesn't notice that he's wearing sensors. Therefore a lot of HAR systems are based on phones and watches [17, 10, 23].

There is also a variety of machine learning techniques being used, both in the preprocessing phase (selection of features) and the actual classification phase. Some classifiers that are used are neural networks [17], Hidden Markov Model [22], Support Vector Machines [1], K-NN [16], Decision Trees [23, 17], Naive Bayes [23, 21], Logistic Regression [17] and Random Forests [23].

As we can see there are a lot of different approaches in the field of HAR. In this work we are providing a new framework for activity recognition using supervised learning, focusing on the machine learning aspect of HAR. We decided to use a Random Forest classifier. The MHEALTH dataset is used `http://archive.ics.uci.edu/ml/datasets/MHEALTH+Dataset` for which measurement was done with wearable sensors. As a second aim, we we will be focusing on which pairs of activities are confused more often with each other in this process. And as a third aim, this work will attempt to identify the most important sensors.

The remainder of this work is structured as follows:

Chapter 2 describes background information on HAR systems, classification in general and provides a theoretical background on Random Forests and feature extraction.

Chapter 3 describes the details of the data preparation. It includes details on the setup that was used to gather data, the set of activities and preprocessing steps.

In Chapter 4 summarizes the results of this work.

Chapter 5 summarizes this thesis and discusses its findings, limitations and future work.

Chapter 2

Preliminaries

2.1 HAR system architecture

In this work we consider the case of wearable sensors. In figure 2.1 a typical structure of such a HAR system is depicted [18, based on figure 2, p. 1194]. Wearable sensors attached to a person measure attributes. Typical sensors are ECG, accelerometer, magnetometer, gyroscopes and heart monitoring sensors. Output of these sensors is sent to a integration device such as a mobile phone or a laptop. The data streams of different sensors are bundled in this phase. Now that the data is stored on one device, the data can be pre-processed locally or could be used for real time monitoring. Larger datasets with data from multiple subjects can be acquired by sending the data over a communication channel such as Wi-Fi and Internet to a remote server. Note that there exists a lot of variety between setups. A simpler setup as depicted is sometimes used. For example a single unit [11], with multiple integrated sensors that can store aggregated data on an SD card, could be preferred. In that case the sensor unit is also the integration and storage device. These units are commercially available nowadays. The downsides to this setup are that pre-processing might be limited due to less processing power and a smaller storage capacity. Aggregating data from different subjects also has to be done afterwards if this is desired.

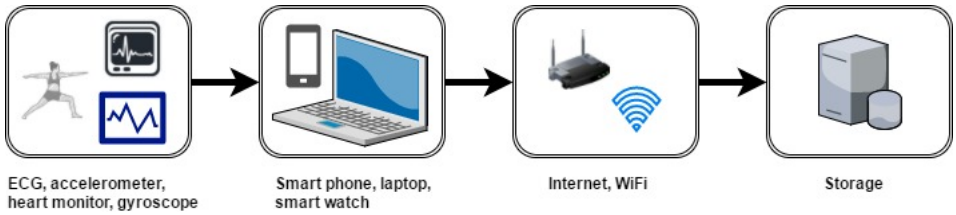

Figure 2.1: A typical structure of a HAR system

2.2 The classification pipeline

A classification task consists of multiple phases. A typical pipeline for classification is depicted in figure 2.2. It starts with raw sensor data, which can contain outliers due to measurement errors. In the pre-processing phase these outliers are removed. Analyzing data without removing outliers can cause misleading results.

Then features are extracted. For some attributes it is beneficial to use derived values instead of the actual values. Derived values could be simple statistical features, such as means, standard deviations, maximum values and minimum values. In time series data, feature extraction reduces data points by binning values in time windows. Multiple measurements taken in a window size are combined into one value resulting in a smaller, derived data set. This may also reduce overfitting in later stages of the classification pipeline. The reasoning behind binning samples is that a single sample provides information for a very small time frame, since sensors often have a high sampling rate. Activities take relatively long (multiple seconds, or multiple minutes), this means that information from a single sample doesn't tell us that much about the activity being performed and it thus makes sense to take a window size.

After we extracted the features, train and test subsets are chosen. The training data is then used to fit a model. The model uses learning algorithms to fit a model using the training set. A fitted model can make a prediction, if given a sample. The test data is used to evaluate the fitted model. In the next section we will give a more detailed explanation on a popular supervised machine learning classifier, a random forest.

Figure 2.2: HAR pipeline using sensor data

2.3 Feature extraction

As explained above, it is beneficial to use derived values of signal data. Determining how to derive these values is an important step in the pre-processing phase. The discriminative power of these derivations influence the classification. Typically used in HAR are the statistical functions mean, standard deviation, maximum and minimum, because they discriminate well in the acceleration domain [6, 20, 13, 17].

Note that making features increases computational cost, which means the number of features should be kept to a minimum if performance is an issue. Choosing the relevant features from a set of features will be discussed in subsection 2.4.2

2.4 Random forest classifier

Classification in supervised learning is the problem of identifying to which category a sample belongs to by learning from a training set with correctly labeled data. In HAR this means that we want to identify the activity the subject is performing in a given time window with a model that learned from correctly labeled training data. There are a lot of classification methods to choose from when encountering a classification problem. They arose from different fields of computer science, statistics and mathematics and a lot of these have implementations available. There is no classifier that performs best on every problem given. To determine which classifier is suited for our problem we will take a look at the work "Do we need hundreds of classifiers to solve real world classification problem" [12], which evaluated 179 classifiers from 17 families with over 100 different data sets. This work concludes that classifiers most likely to be the best for real world classification problems are random forest classifiers. In the next section we will explain the concept of a random forest classifier, which will be used later on in this work.

2.4.1 Algorithm

Random forest classifiers are based on random classification trees incorporated into an ensemble. Leo Breiman modified a technique called bootstrap aggregation (bagging) [8] to build a collection of de-correlated trees and called the resulting technique random forests. There are a lot of variations and different implementations, but here we will explain the general concept. A number of decision tree classifiers are fit on sub-sets of the data set. After growing the ensemble of trees, there is a poll amongst the trees for the most popular class. Predictive accuracy is increased by averaging results from these trees. We are going to assume the reader is familiar with decision trees and take a look at a random forest algorithm (figure 2.3) for classification.

The algorithm starts with selecting the number of trees. Each tree is constructed on a random subset of the data. In the next step each tree is grown using the bootstrapped data for that tree. This is done by selecting m features from the set of features in the data, **at random**. Recommendation from the inventors states that the default value for m should be $\left\lfloor \sqrt{(}p) \right\rfloor$, but optimal values depend on the problem and therefore the parameters of the algorithm might need some tuning. For every of those m selected features,

Algorithm 15.1 *Random Forest for Classification.*

1. For $b = 1$ to B:
   (a) Draw a bootstrap sample $\mathbf{Z}^*$ of size N from the training data.
   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.
      i. Select m variables at random from the p variables.
      ii. Pick the best variable/split-point among the m.
      iii. Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point x:

Classification: Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest tree. Then $\hat{C}_{\text{rf}}^B(x) = \textit{majority vote}\ \{\hat{C}_b(x)\}_1^B$.

Figure 2.3: Random forest algorithm, from [14, p. 588]

a split is done and from those splits the best is selected. The quality of the splits is assessed using impurity functions such as Gini and the entropy, which both are a measure of misclassification.

Gini [7, p. 42]:

$$\phi(p) = \sum_j p_j(1 - p_j)$$

j denotes a class and p_j is the probability of items labeled with j.

Entropy [7, p. 42]:

$$\phi(p) = -\sum_j p_j \log p_j$$

In the next step of the algorithm the node is split, using the best split found in the previous step. Note that the algorithm is recursive in terminal nodes of the trees. It stops when the minimum node size n_min is reached. The set of all the trees formed is the ensemble. Classification of a test point is done, by pushing it through all the trees in the ensemble until a leave is reached and then a majority vote is held amongst the trees.

2.4.2 Feature importance

To determine which sensors are most important for HAR we are going to take a look at the extracted features of that sensor. Importance of features can be measured in multiple ways.

One approach to measure feature importance is by performing statistical permutation tests. Breiman came up with this approach as a first step, but said that more research would be necessary to understand this mechanism [9]. The idea is to measure the decrease in accuracy, using out-of-the-bag

data. After the trees are constructed, one feature of the out-of-the-bag samples is randomly permuted and run down the corresponding tree. The predicted classes for each sample is stored. This procedure is done for every feature and afterwards the votes are compared to the true class label. The result is a measurement of misclassification when a feature noised up.

Another approach is based on Gini feature importance, often called MeanDecreaseGini. In the process of training each tree, splits are assessed using impurity functions such as Gini. Every time a node is split the Gini impurity of the two descendant nodes is smaller than the parent node.

$$D = \text{Gini}_{\text{parent}} \text{ - } \text{Gini}_{\text{child1}} \text{ - } \text{Gini}_{\text{child2}}$$

where D denotes the decrease in Gini from the split performed

The importance of a variable is measured by taking all the splits the variable is involved in across the whole forest and average the decreases in Gini. From these values a ranking on importance can be made. The highest value (highest decrease in Gini) belongs to the most important feature for classification. Results from these two approaches are often consistent. But the Gini approach is a relatively cheap option by means of performance.

Chapter 3

Data Preparation

In this chapter we will go over the details of the data preparation (also called preprocessing in figure 2.2). It provides practical information on how the research was conducted and provides insights on the decisions, that had to be made along the way. First we will start of with the origin of the data.

3.1 Experimental set-up

The MHEALTH (Mobile health) is a dataset available at http://archive.ics.uci.edu/ml/datasets/MHEALTH+Dataset [4]. Motion and vital signs of volunteers were measured using wearable sensors while performing several physical activities with periods of rest between them. Ten volunteers of diverse profile were asked to perform 12 basic physical activities (table 3.1) to the best of their abilities. Sensors were placed on the subject's chest, right wrist, and left ankle using elastic straps, as depicted in figure 3.1. Accelerometers, gyroscopes and magnetometers for X,Y and Z axis were used to capture three dimensional movement. These are embedded in Shimmer2 sensors (the white boxes in figure 3.1) [11]. The blue sensors attached to the chest and stomach are used for ECG measurements.

The units of measurement for the accelerometers, gyroscopes and magnetometers are in m/s^2, deg/s and local respectively. Additional ECG measurements were taken with sensors on the chest in mV. This information can potentially be used for simple heart monitoring or effects of the activities on the ECG signal. All the sensors used have a sampling rate of 50 Hz. Each session was recorded using a video camera. The video recordings were used to label the data and to check for anomalies in the signal.

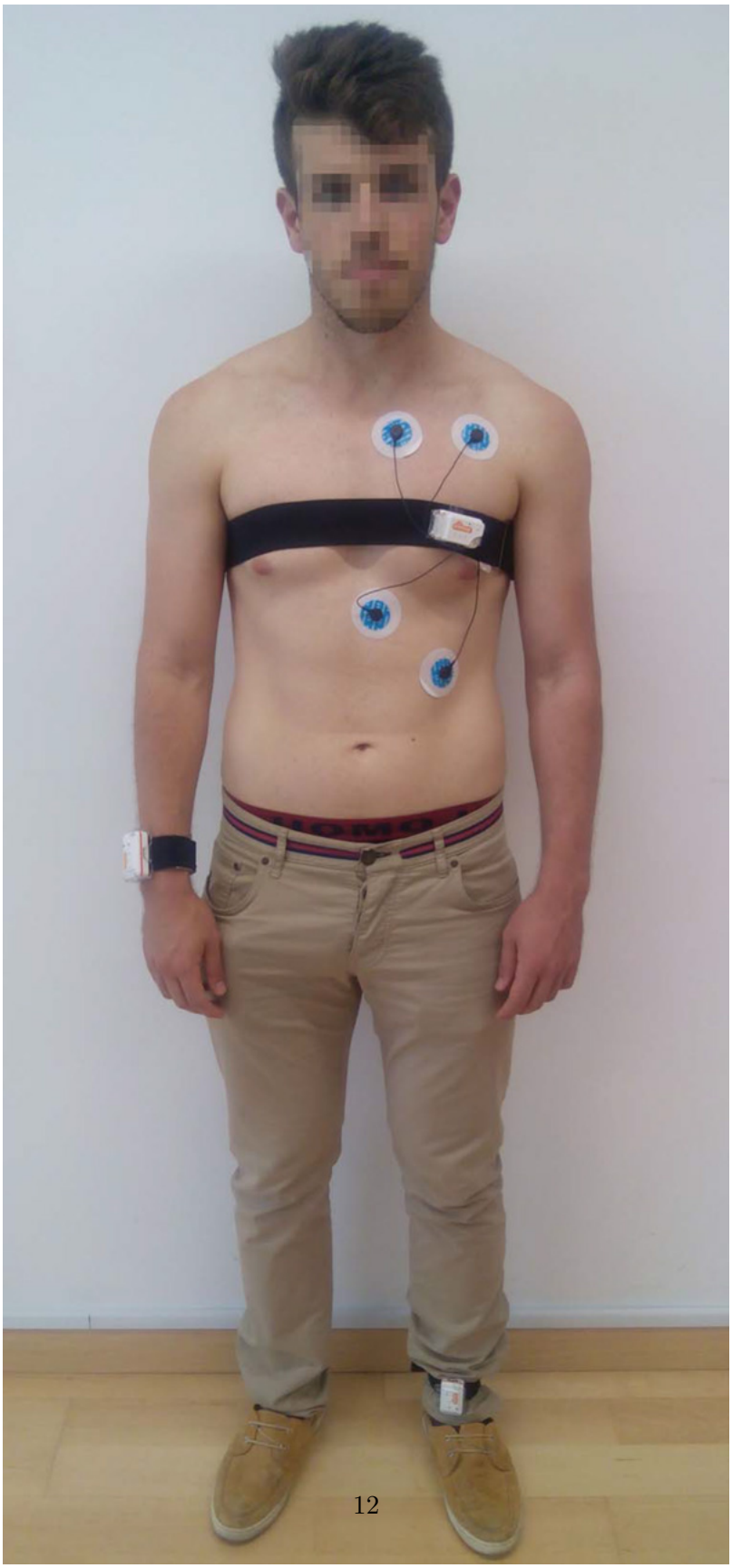

Figure 3.1: The sensors placed on the body of a subject. From [5, p13, figure 5]

3.1.1 Activities

The activities consist of simple daily life things. The activities have different intensities, speeds and motions and therefore should be distinguishable.

Activity	Description	Duration
L1	Standing still	1 min
L2	Sitting and relaxing	1 min
L3	Lying down	1 min
L4	Walking	1 min
L5	Climbing stairs	1 min
L6	Waist bends forward	20x
L7	Frontal elevation of arms	20x
L8	Knees bending (crouching)	20x
L9	Cycling	1 min
L10	Jogging	1 min
L11	Running	1 min
L12	Jump front and back	20x

Table 3.1: The activity set

3.1.2 Storage format

The data is collected separately for each of the subjects and stored in a log file. Each row in this file is a sample with the different sensors as column (table 3.2). The last column is the label. Labels range from 0 to 12. 0 for the null class and for the activities the numbers from table 3.1 are used. The null class label was used as a transition phase between activities, as for example ECG signals might still be elevated from the previous activity.

3.2 Preprocessing

Now we know how the data was gathered and stored, we can start preparing it for use in a random forest classifier. We chose to do our experiment in Python using scikit-learn, a free machine learning library. Scikit-learn has various features for data analysis. For our research, features for preprocessing as well as classification, evaluation and visualizing data using plots would come in handy.

The raw data of each subject is split on lines to retrieve a sample. Sensor values in a sample were tab-separated. After splitting on tabs and casting

Column	Sensor	Bodypart	Axis
0	acceleration	chest	X
1	acceleration	chest	Y
2	acceleration	chest	Z
3	electrocardiogram signal	chest	lead 1
4	electrocardiogram signal	chest	lead 2
5	acceleration	left-ankle	X
6	acceleration	left-ankle	Y
7	acceleration	left-ankle	Z
8	gyro	left-ankle	X
9	gyro	left-ankle	Y
10	gyro	left-ankle	Z
11	magnetometer	left-ankle	X
12	magnetometer	left-ankle	Y
13	magnetometer	left-ankle	Z
14	acceleration	right-lower-arm	X
15	acceleration	right-lower-arm	Y
16	acceleration	right-lower-arm	Z
17	gyro	right-lower-arm	X
18	gyro	right-lower-arm	Y
19	gyro	right-lower-arm	Z
20	magnetometer	right-lower-arm	X
21	magnetometer	right-lower-arm	Y
22	magnetometer	right-lower-arm	Z
23	Label (0 for the null class)		

Table 3.2: Dataset file

to floats, we end up with a matrix, where each row is a sample and in each column the values for one of the sensors (table 3.1).

3.2.1 Segmentation

We started of by making segments of samples for each subject. This was done separately for each subject, because it wouldn't make sense to make a segment that combines samples from two different subjects. Segmentation is a preparation step before feature extraction. The data was ordered in chunks of the same class. From those chunks a few of the first samples were removed so that in the next step, segments where every class label is identical, would be formed. Segments were chosen to be the size of 100 samples, which is equal to a window of 2 seconds. This specific window size is thought to be a good trade-off between speed and accuracy [3]. By making features from the newly formed segments the number of data points is reduced by a factor of roughly 100, which will speed up the classification process. Performance is not our main concern, but it should always be taken into account. Note that a bigger window size has the advantage of increasing the accuracy (see explanation given in 2.2). Afterwards data from different subjects were combined to one matrix. In our implementation we kept track of which sample belongs to which subject, as this is needed in the evaluation phase.

3.2.2 Feature extraction

We chose to extract features using the statistical functions mean, standard deviation, maximum and minimum. As explained in section 2.3 features made from these functions seem to be typically used in HAR, because they discriminate well in the acceleration domain [6, 20, 13, 17]. This means we will end up with $23 * 4 = 92$ features. The basic idea is to take slices, with the size of the window size, of the original matrix and then form a new smaller derived matrix from the results of these functions.

Mean

The mean is the sum of each element divided by the total number of elements, given by:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

Standard deviation

The standard deviation is the square root of the average of the squared deviations from the mean. We used the implementation of NumPy, with the degrees of freedom parameter set to 1, resulting in:

$$s = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2}$$

Max

The maximum is the highest value of each element.

Min

The minimum is the lowest value of each element.

3.2.3 Class distribution

After extracting features we noticed that the null class had significantly more data points (figure 3.2). An unbalanced dataset might lead to a model that will predict the null class a lot, since it occurs so much in the dataset. We came up with two different strategies to handle with this. The first one is simply removing some instances of the null class. We chose to remove the first 95 percent of the data points with class null per subject to make the classes balanced (figure 3.2). We chose to do this in the preprocessing phase. Later on 10 fold cross validation will be used to test the classifier. Note that because we remove instances of the null class before making a test and train split, the test data will contain significantly less instances of the null class as well. In the solution, we provide next, we will use weights on classes to make the two solutions comparable. By using weights we can make the null class less important, which is comparable to removing instances of the null class.

The second solution is classifying in two steps. In the first step a binary classifier is used to distinguish between the null class and the rest of the classes. In the second step a multi-class classifier is used to predict the exact class, if in the previous step it was predicted to not be from the null class. Note that these are two separate random forest classifiers. As mentioned above, we had to weigh classes differently to make it comparable to the first solution. Our main goal was to adjust the weights in a manner that is similar to having a test set with an even distribution in classes, as in the previous solution. We chose to weigh not null classes 3 times as hard as the null class in step 1, since not null classes combined only had 4 times more instances as the null class and in the previous step this was roughly 12 times. In the second step we weighed the not null classes 30 times as hard as the null class to mimic a balanced test set, since each class occurred 30 times less than the null class. This seemed to be simplest fix for this issue. We will discuss the results of these two solutions in chapter 4.

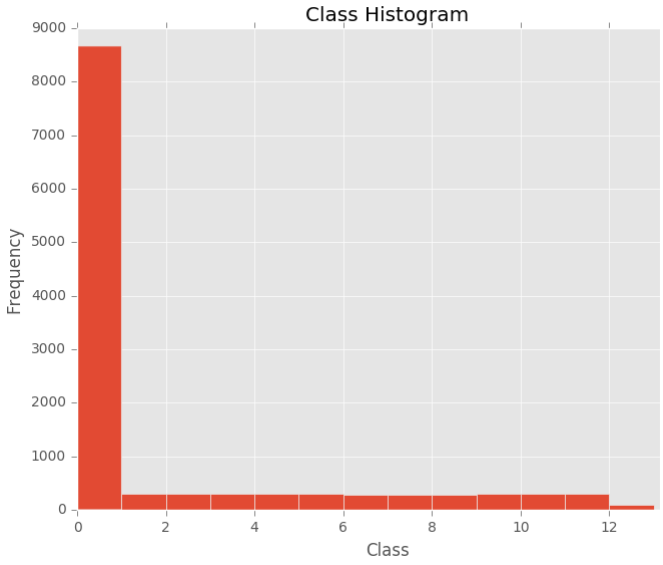

Figure 3.2: Class histogram before removing samples from the rest class

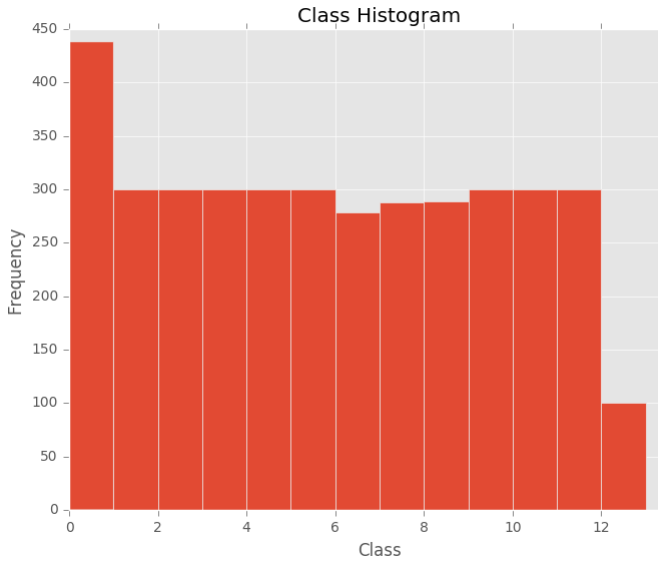

Figure 3.3: Class histogram after removing samples from the rest class

Chapter 4

Results

4.1 Classification performance

Table 4.1 depicts the results for both solutions. The accuracy score of classifying in two steps was generally worse comparing to the score of the solution with removed samples. We chose to use 10 fold cross validation to test classifiers.

Solution	**Accuracy score**
Removing samples	0.87
Two step classification	0.75

Table 4.1: Comparison between the two solutions

4.2 Analysis on the first solution

In this section we will analyze the first solution, where we removed samples of the null class. Figure 4.1 shows the confusion matrix of the classification. It shows the number of correct predictions on the diagonal. The rest of the matrix shows the incorrectly predicted instances and their true label.

The first thing we noticed by looking at the confusion matrix was that null class and the activities with a low intensity were confused a lot. This is mostly seen when the true label is null. This seems logical, because in the time frame between activities indicators of intensity might still be elevated. A second thing we noticed, is a lot of confusion between running and jogging. These activities are much alike with regard to movements, but are different in intensity. Jogging is more confused for running than the other way around, which indicates that the intensity of jogging might be a bit overestimated.

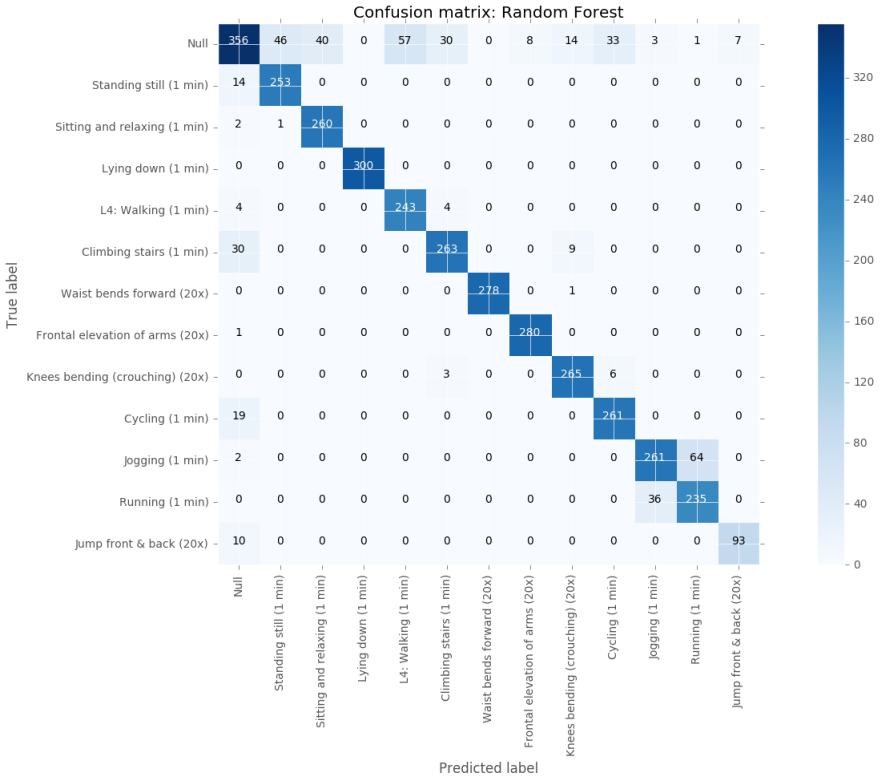

Figure 4.1: Confusion matrix of classification using random forest.

4.2.1 Feature importance

Figure 4.2 shows the ranking in importances of all created features. Min, max, std and mean stand for the statistical function used and the number afterwards stands for the specific sensor, placed on a specific body part, capturing a specific angle (see table 3.2). For example max11, the highest ranked feature, stands for the maximum value for magnetometer placed on left-ankle (X axis).

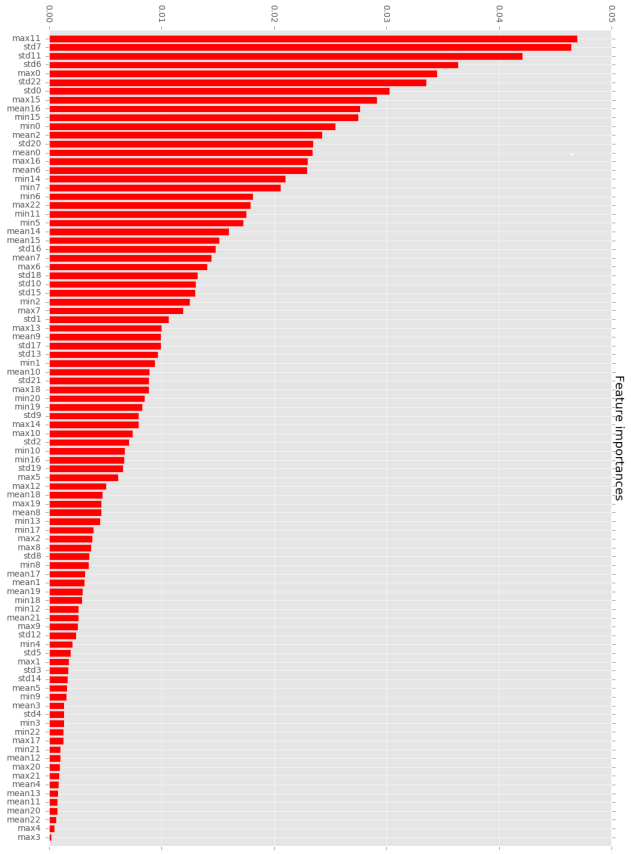

Figure 4.2: Feature importance ranking.

The translation of the plot in figure 4.2 results in the ranking depicted in table 4.2.

Rank	**Feature**
1	maximum magnetometer left-ankle sensor (X axis)
2	standard deviation acceleration left-ankle sensor (Z axis)
3	standard deviation magnetometer left-ankle sensor (X axis)
4	standard deviation acceleration left-ankle sensor (Y axis)
5	maximum acceleration chest sensor (X axis)

Table 4.2: Top 5 features

Table 4.2 indicates that features derived from the magnetometer on the left ankle in X axis is the most effective sensor for recognizing activities, followed by features derived from the acceleration sensor on the left-ankle in as-well Z as Y direction. Note that top four features are all derived from

sensor data from the left-ankle. This indicates the sensor placed on the foot is critical for this specific set of activities.

Chapter 5

Discussion

5.1 Findings

5.1.1 Classification performance

As table 4.1 shows, we reached a accuracy of 0.87 in the case where we removed samples in the data preparation phase. Related work [5] shows results in the same order of magnitude, if we consider the fact that in that work they only evaluated the activity classes, excluding the null class. Excluding evaluation on the null class is something we consider for future work, since it is not regarded as an activity. For HAR it might be more relevant to exclude it.

Confusion between activities

In the previous chapter we considered the confusion matrix in table 4.1 and found that the most confused activities were jogging and running. Jogging was more confused for running than the other way around. Related work [5] also reported the most confusion between these activities. Jogging and running are much alike activities. In most humans, motions of these activities are roughly the same. A distinction can be made by looking at the intensity. Table 3.2 depicts all sensors that were placed. We will consider standard deviation from the acceleration of the left ankle in Z-axis as a feature that could indicate levels of intensity and then compare the means of this feature where running is predicted, but jogging was performed, to means of where it actually is jogging. In table 5.1 we can see that these values are quite similar. Similarity indicates that this feature is a factor in the confusion between these activities.

Case	Mean std acceleration left ankle (Z-axis)
Predicts running, is jogging	9.4
Jogging	8.9

Table 5.1: Comparison between mean values

5.1.2 Feature importance

In the previous chapter we concluded that the acceleration sensor and the magnetometer on the left ankle were critical for recognizing activities, which indicates foot sensors are important, since no sensors were placed on the right foot of a subject. An explanation for the importance of the acceleration sensor on the left ankle might be that a lot can be told from non moving feet versus moving feet. Moving feet has to mean a subject is either walking, climbing stairs, cycling, jogging, running or jumping front and back. Non moving feet indicates a subject is either standing still, sitting and relaxing, lying down, doing waist bends, performing frontal elevation of arms or crouching. This split results in a lot of information gain.

We found it hard to find an explanation for the importance of the magnetometer, as we lack the knowledge about the physics of magnetism. Our theory is that values from the magnetometer also indicate movement, since a magnetic field might be different at a different location.

Bibliography

[1] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge L Reyes-Ortiz. Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine. In *International Workshop on Ambient Assisted Living*, pages 216–223. Springer, 2012.

[2] Omar Aziz and Stephen N Robinovitch. An analysis of the accuracy of wearable sensors for classifying the causes of falls in humans. *IEEE transactions on neural systems and rehabilitation engineering*, 19(6):670–676, 2011.

[3] Oresti Banos, Juan-Manuel Galvez, Miguel Damas, Hector Pomares, and Ignacio Rojas. Window size impact in human activity recognition. *Sensors*, 14(4):6474–6499, 2014.

[4] Oresti Banos, Rafael Garcia, Juan A Holgado-Terriza, Miguel Damas, Hector Pomares, Ignacio Rojas, Alejandro Saez, and Claudia Villalonga. mhealthdroid: a novel framework for agile development of mobile health applications. In *International Workshop on Ambient Assisted Living*, pages 91–98. Springer, 2014.

[5] Oresti Banos, Claudia Villalonga, Rafael Garcia, Alejandro Saez, Miguel Damas, Juan A Holgado-Terriza, Sungyong Lee, Hector Pomares, and Ignacio Rojas. Design, implementation and validation of a novel open framework for agile development of mobile health applications. *Biomedical engineering online*, 14(2):S6, 2015.

[6] Ling Bao and Stephen S Intille. Activity recognition from user-annotated acceleration data. In *International Conference on Pervasive Computing*, pages 1–17. Springer, 2004.

[7] Leo Breiman. Technical note: Some properties of splitting criteria. *Machine Learning*, 24(1):41–47, 1996.

[8] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[9] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[10] Tomas Brezmes, Juan-Luis Gorricho, and Josep Cotrina. Activity recognition from accelerometer data on a mobile phone. *Distributed computing, artificial intelligence, bioinformatics, soft computing, and ambient assisted living*, pages 796–799, 2009.

[11] Adrian Burns, Barry R Greene, Michael J McGrath, Terrance J O'Shea, Benjamin Kuris, Steven M Ayer, Florin Stroiescu, and Victor Cionca. Shimmer^{TM}–a wireless sensor platform for noninvasive biomedical research. *IEEE Sensors Journal*, 10(9):1527–1534, 2010.

[12] Manuel Fernández-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems. *J. Mach. Learn. Res*, 15(1):3133–3181, 2014.

[13] Davide Figo, Pedro C Diniz, Diogo R Ferreira, and Joao MP Cardoso. Preprocessing techniques for context recognition from accelerometer data. *Personal and Ubiquitous Computing*, 14(7):645–662, 2010.

[14] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[15] Ahmad Jalal, Md Zia Uddin, and T-S Kim. Depth video-based human activity recognition system using translation and scaling invariant features for life logging at smart home. *IEEE Transactions on Consumer Electronics*, 58(3), 2012.

[16] Sahak Kaghyan and Hakob Sarukhanyan. Activity recognition using k-nearest neighbor algorithm on smartphone with tri-axial accelerometer. *International Journal of Informatics Models and Analysis (IJIMA), ITHEA International Scientific Society, Bulgaria*, 1:146–156, 2012.

[17] Jennifer R Kwapisz, Gary M Weiss, and Samuel A Moore. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82, 2011.

[18] Oscar D Lara and Miguel A Labrador. A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys and Tutorials*, 15(3):1192–1209, 2013.

[19] Benoit Mariani, Mayté Castro Jiménez, François JG Vingerhoets, and Kamiar Aminian. On-shoe wearable sensors for gait and turning assessment of patients with parkinson's disease. *IEEE transactions on biomedical engineering*, 60(1):155–158, 2013.

[20] Nishkam Ravi, Nikhil Dandekar, Preetham Mysore, and Michael L Littman. Activity recognition from accelerometer data. In *Aaai*, volume 5, pages 1541–1546, 2005.

[21] Emmanuel Munguia Tapia, Stephen S Intille, and Kent Larson. Activity recognition in the home using simple and ubiquitous sensors. In *International Conference on Pervasive Computing*, pages 158–175. Springer, 2004.

[22] TLM Van Kasteren, Gwenn Englebienne, and Ben JA Kröse. Activity recognition using semi-markov models on real world smart home datasets. *Journal of ambient intelligence and smart environments*, 2(3):311–325, 2010.

[23] Gary M Weiss, Jessica L Timko, Catherine M Gallagher, Kenichi Yoneda, and Andrew J Schreiber. Smartwatch-based activity recognition: A machine learning approach. In *Biomedical and Health Informatics (BHI), 2016 IEEE-EMBS International Conference on*, pages 426–429. IEEE, 2016.

[24] Jie Yin, Qiang Yang, and Jeffrey Junfeng Pan. Sensor-based abnormal human-activity detection. *IEEE Transactions on Knowledge and Data Engineering*, 20(8):1082–1090, 2008.

[25] Huiru Zheng, Haiying Wang, and Norman Black. Human activity detection in smart home environment with self-adaptive neural networks. In *Networking, Sensing and Control, 2008. ICNSC 2008. IEEE International Conference on*, pages 1505–1510. IEEE, 2008.