Karin Wisiol, BSc BSc

Human-Activity Recognition

MASTER'S THESIS

to achieve the university degree of

Diplom-Ingenieurin

Master's degree programme: Geomatics Science

submitted to

Graz University of Technology

Supervisor

Ao.Univ.-Prof. Dipl.-Ing. Dr.techn. Manfred Wieser

Institute of Navigation

Graz, November 2014

Eidesstattliche Erklärung
Affidavit

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit identisch.

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources. The text document uploaded to TUGRAZonline is identical to the present master's thesis.

Datum / Date	Unterschrift / Signature

Acknowledgement

First of all, I would like to thank my supervisor, Univ.-Prof. Dr.techn. Manfred Wieser, for suggesting the topic of this thesis, the ground-breaking and constructive suggestions along the way and proofreading my work.

I also would like to thank Dipl.-Ing. Thomas Moder, who supervised me throughout my entire work. He had always an open ear for me and had patience with solving a variety of problems.

The helpful people and the pleasant and friendly atmosphere at the Institute of Navigation have made a great contribution to this work.

My special thanks go to my family and my partner Markus.
I would like to say how much I appreciate Markus for his encouragement and being a mental support.
I am grateful to my mother and grandmother for giving me the opportunity to complete my studies and for their great support during the past years.

Abstract

The present work deals with the recognition of human movements by means of low-cost inertial sensors. Simple activities such as *standing*, *sitting*, *walking*, *running*, *walking on stairs*, etc. are examined on the basis of different devices, such as Smartphone and Smartwatch, worn at various locations on the body. Inertial data, i.e. accelerations and angular velocities, of known activities are needed to train a classifier which then should be able to predict unknown data. These measured data must be processed in a suitable manner to obtain therefrom so-called features which serve as input for the classifier.

In this work, the suitability of a Smartphone for activity recognition is established first. Based on this, several features are selected as the input variables of the C4.5 decision tree preferred in this work. User-specific and speed-dependent activities are examined as well as the real-time capability. Several classifiers are compared and, finally, an algorithm for determining the current sensor position is presented.

The use of coordinates in the local level frame and a self-defined body-fixed frame contributes to the improvement of activity recognition. The evaluation of the created classifiers yields good results for all activities. Although the analysis of test data shows not such high accuracies, they are still satisfactory. The user-specific activities *upstairs* and *downstairs* are the hardest to detect, whereas all other activities can be predicted robustly.

Kurzfassung

Die vorliegende Arbeit befasst sich mit der Erkennung von menschlichen Bewegungen mittels low-cost Inertialsensoren. Einfache Aktivitäten wie *Stehen*, *Sitzen*, *Gehen*, *Laufen*, *Stiegen steigen* usw. werden auf Basis unterschiedlicher Geräte, wie Smartphone und Smartwatch, die an verschiedenen Positionen am Körper getragen wurden, untersucht. Inertialdaten, d.h. Beschleunigungen und Winkelgeschwindigkeiten, von bekannten Aktivitäten werden benötigt, um einen Klassifikator zu trainieren, der dann im Stande sein soll, aktuelle Daten auszuwerten. Diese Messdaten müssen auf geeignete Weise bearbeitet werden, um daraus sogenannte Features zu erhalten, die als Input für den Klassifikator dienen.

In dieser Arbeit wird zuerst die Eignung eines Smartphones für die Aktivitätserkennung festgestellt. Basierend darauf werden verschiedene Features als Eingangsgrößen des in dieser Arbeit bevorzugten C4.5 Entscheidungsbaumes gewählt. Benutzerspezifische und geschwindigkeitsabhängige Aktivitäten werden untersucht, sowie die Echtzeitfähigkeit getestet. Mehrere Klassifikatoren werden miteinander verglichen und schließlich ein Algorithmus zur Bestimmung der aktuellen Sensorposition vorgestellt.

Die Verwendung von Koordinaten im lokalen Horizontsystem und einem selbst definierten körperfixierten System trägt zur Verbesserung der Aktivitätserkennung bei. Die Evaluierung der erstellten Klassifikatoren liefert für alle Aktivitäten gute Ergebnisse. Die Analyse mit Testdaten zeigt zwar nicht so hohe Genauigkeiten, diese sind immer noch zufriedenstellend. Die benutzerspezifischen Aktivitäten *Stiegen hinauf* und *Stiegen hinunter* sind am schwierigsten zu detektieren, während alle anderen Aktivitäten robust prädiziert werden können.

Contents

Abbreviations

AAL	Ambient Assisted Living
ADL	Activities of Daily Living
ANN	Artificial Neural Network
AR	Activity Recognition
BF	Body Frame
BN	Bayesian Network
CART	Classification and Regression Tree
CFS	Correlation-based Feature Selection
DT	Decision Tree
FFT	Fast Fourier Transformation
FOG	Fiber Optic Gyroscope
HAR	Human Activity Recognition
IADL	Instrumental Activities of Daily Living
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
IQR	Interquartile range
k-NN	k-Nearest Neighbour
LLF	Local Level Frame
MAX	Maximum value
MEAN	Mean (averaged) value
MEMS	Micro-Electro-Mechanical Systems
MFC	Main Frequency Component
NB	Naive Bayes
PDR	Pedestrian Dead Reckoning
RLG	Ring Laser Gyroscope
RMS	Root Mean Square
SF	Sensor Frame
STFT	Short Time Fourier Transform
SVM	Support Vector Machine
UTC	Universal Time Coordinated
WEKA	Waikato Environment for Knowledge Analysis

1 Introduction

1.1 Motivation

Human Activity Recognition (AR) is an approach of monitoring human movements. Various activities such as walking, sitting, standing, running, cycling, falling down, etc. are to be detected to thereby obtain information, especially for Ambient Assisted Living (AAL). The EU runs a research program on AAL which has the aim of enhancing the living quality of elderly people by using Information and Communication Technologies. The constant and increased ageing in Europe induces challenges for the states' finance and health system, but also affords an "opportunity to live a long and better life after working life-time"[2]. Therefore, the main interest lies in developing platforms for AAL. In cause of the large number of devices, services, functionalities and the evaluation aspects to be compared, the EvAAL competition - Evaluating AAL Systems through Competitive Benchmarking - was launched [1]. For several years this competition takes place once a year, where an indoor positioning including AR has to work in real-time.

Thus, the field of applications for AR has a wide variety [22]: Healthcare and Assisted Living for elderly people aims to detect potentially dangerous situations in a person's life to start an automated call for external help, for example when a person has fallen. Employing long-term monitoring to detect changes or unusual patterns in a person's daily life that may indicate early symptoms of diseases like Alzheimer's is another class of utilization. The more popular usage of AR systems consists of motivating and promoting people to maintain a more active and therefore healthy life by, e.g., showing them how many calories they burned each day through performing dynamic activities [52].

AR currently works via image-based techniques, but also inertial sensors and combined methods. The expected rise of people owning Smartphones and growing of willingness to use Smartwatches, which include inertial sensors as well, increases the interest in research in this direction.

1.2 Problem statement

The aim of this thesis is the analysis and development of algorithms for AR with low-cost sensors. The AR should be studied by a medium-priced Inertial Measurement Unit (IMU) and finally evaluated and tested with low-cost inertial sensors from a Smartphone or Smartwatch. In this context, the effect of individual sensor locations shall be investigated. In addition, a classification algorithm for the detection of the sensor's current body position is to be developed. In the future, AR should be possible on a Smartphone in real-time. Moreover, an intent is to use the AR result as support for a Pedestrian Dead Reckoning (PDR) algorithm.

1.3 Methodology

An AR system mainly consists of a classifier that gets as an input a vector of attribute values, called features and outputs a class, the activity. Features are extracted from inertial data that have been recorded during the performance of activities from a device. After a preprocessing, e.g., rotation into a different coordinate system, the data are divided into sections, called windows, and on the basis of the data points within this window a statistical value is calculated, e.g. the mean value, which represents one feature. A classifier usually has to be trained on labelled data, where the activity is known. Once a classifier was built and tested with training data, it can be used to predict unknown data.

1.4 Thesis Outline

This thesis includes another five chapters. Chapter 2 presents the state of the art in the field of AR. Different aspects to be considered are summarized and works on AR with inertial sensors and Smartphones respectively Smartwatches are presented. Chapters 3 contains background information that is important for the understanding of the sensors and algorithms that appear in this thesis. The development of the AR system and the procedure in the course of the definition of desired activities up to the evaluation of classification results is discussed in Chapter 4. In Chapter 5, various investigations are carried out: from the suitability of low-cost sensors via user-specific activities and different classifiers to the design of an algorithm to determine the current sensor location. Chapter 6 draws conclusions, that have been made in this thesis, and provides an outlook for future work.

2 State of the art

First research on Activity Recognition (AR) using accelerometers date back to the 1990s [17], when advanced hardware technology opened the possibility of wearing lightweight sensing-, computing- and display-equipment by a single person [22]. Since the development of small, lightweight and inexpensive Micro-Electro-Mechanical Systems (MEMS) inertial sensors, investigations regarding AR are increasing strongly.

In the past decade the use of MEMS-based Inertial Measurement Units (IMUs) for AR has been studied in several research projects. The increasing availability of inertial sensors such as accelerometers in consumer products, like smart phones, and the big potential of applications have attracted interest to AR as a promising research topic.

2.1 Various aspects

It depends on the final application what kind of sensors have to be used to fulfil the requirements of the desired system. Application examples have already been discussed in Chapter 1. In the context of designing a Human Activity Recognition (HAR) system, as it is described in the previous section, many aspects have to be encompassed:

Recognized activities Plenty of different activities have been studied in related work [22]:

- Human motion activities like walking, sitting, lying, walking on stairs or standing.
- Activities of Daily Living (ADL) such as dressing, bathing, toileting.
- Instrumental Activities of Daily Living (IADL) like food preparation, using the phone or shopping.
- Sports activities: cycling, callisthenics, running or rowing.
- Short-term activities (gestures) such as open the door or pulling the handbrake.

Depending on the activities, that should be recognized from an AR system, different types of sensors come into consideration. Activities related to human motion require the

usage of sensors related to motion, while for activities related to social interactions, like talking on the phone or attending a meeting, other kinds of sensors - that would be for example a microphone - might be required [39].

Types of sensors Besides accelerometers, which are the most commonly used sensors in related work, a wide range of different sensors has been employed for recognizing activities: physiological sensors like skin conductivity, heart rate and body temperature sensors, microphones, light sensors, humidity and barometric sensors, gyroscopes, video data or Radio-frequency identification (RFID) tag readers. These are just a few which are mentioned by Huynh [22]. There have been much more sensors in use yet.

Location of measurement devices Most researches on AR have used accelerometers in multiple locations of the body. There exists recent work where high recognition rates have been achieved by using several sensors in one single location [8]. Bao and Intille [6] observed that an accelerometer placed on the thigh or hip were the two most powerful to distinguish between their set of 20 activities including human motion activities, ADLs and IADLs. Based on this result, the research on IMUs placed on the hip [18, 15] or near the pelvic region [45], or Smartphones carried in a user's pocket or on the waist [27, 52, 44, 4, 24, 59] - the most convenient location (at least for men) - has grown.

Data set The quantity of data and the constraints when collecting them is different in each work. While most of the researchers collect their own data, there already exist online databases where data sets are made available to the public for further research [18]. The number of used subjects to collect data varies pretty much from one person - mostly the researcher himself - to a few tens of persons where the aim is to get a more realistic and representative data set. Another criterion to take into account are the conditions under which the data was collected [39]. Laboratory conditions are distinguished from semi-naturalistic and naturalistic conditions. The (self-)reporting of the performed activities poses quite a challenge and requires special auxiliary devices or software.

Feature extraction methods To obtain sufficient information to describe the performed activity, the signal is processed by windowing with some overlap. The window length in related work varies from 1 second [59] up to 10 seconds [27, 33]. In order to receive quantitative measures, statistical methods are used to extract features [28] in order to find useful hidden information and eliminate the noise in the raw data from data collection process or sensors [3].

The used sensor and its location designate the features to compute. Bin Abdullah et al. [3] grouped all features used by HAR researchers using a Smartphone into four categories:

1. Magnitude-based features rest upon the magnitude values of sensors and are mainly based on the raw values from sensors (x-axis, y-axis, z-axis, axes means, axes standard deviation, axes minimum, axes maximum, axes min minus max, axes max minus min, kurtosis, average absolute difference, zero crossing rate and 75% percentile).
2. Frequency-based features are based on the frequency values of sensors where the most commonly used feature is from Fast Fourier Transformation (FFT): frequency domain entropy, maximum frequency, FFT energy, FFT mean and FFT standard deviation.
3. Correlation features where correlations between axes or between features are used.
4. Other features that are not in the previous categories. Khan et al. [23] used autoregressive coefficient and signal magnitude area for their kernel discriminant analysis.

In order to enable an accurate and reliable system that works in real time with long battery life, the required feature computation should be studied. The often large set of features at the beginning should be reduced to fulfil the requirements of the system [39]. Some feature selection techniques are:

- Linear Discriminative Analysis [57].
- Feature Subset Selection [57].
- Forward and backward sequential search algorithms [42].
- Evaluation of information gain from attribute and correlation-based feature subset selection [36].

Learning algorithms AR is based on machine learning techniques. A supervised learning algorithm has to be trained on labelled data, i.e., the output is known and aims to generate a classifier. A classifier is a system that gets as an input a vector of feature values and outputs a single discrete value, the class [38].

The effort of labelling the data can be avoided by using unsupervised approaches. These imply the construction of models from unlabelled data, where groups of similar examples are discovered which is called clustering [22]. Semi-supervised learning has been implemented in relatively few approaches. It can be applied when parts of the available

data are labelled, while for other parts no labels exist [22]. The most commonly used supervised learning methods are [28]:

- Decision Trees [6, 49, 13, 27, 33, 59, 36]
- Bayesian methods (mostly Bayesian Network (BN) and Naive Bayes (NB)) [6, 5, 13, 33, 59, 36]
- Instance-Based-Learning (e.g. k-Nearest Neighbour (k-NN)) [6, 33, 44, 59, 36]
- Support Vector Machine (SVM) [44, 4, 7, 24]
- Artificial Neural Network (ANN) [27, 33, 44, 59, 58]
- Classifier Ensembles (e.g. bagging, boosting, stacking) [45, 7]

Evaluation methods The majority of researchers use the cross validation technique to compare the performance of classifiers. Therefore the features are divided into a training and a test set, the classifier learns from the training set, and the classification performance of the trained classifier on the test set is evaluated. Commonly these steps are repeated with different partitioning into training- and test set, where the results are averaged [22]. This kind of evaluation provides better estimates of the classifier performance especially in case of a limited data set. The classification results are usually stated by the classifier accuracy. But this kind of evaluation metrics doesn't provide a detailed interpretation, hence precision and recall for every activity are the favoured quantities. An extensive presentation of the results can be given by the confusion matrix where each column shows the instances in a predicted class, while each row displays the instances in an actual class. All other evaluation parameters can be calculated from it. A detailed description on evaluation methods is given in Section 3.3.2.

The results depend on the performed tests as well as on the type of model. Lockhart and Weiss [33] define three types of models: impersonal, personal and hybrid models. They differ in which people were taken for the creation of models and the model used for activity prediction. The authors conclude that personalized models perform best whereas impersonal models perform much worse, because they cannot effectively distinguish between certain activities which are user-specific.

Besides the above mentioned aspects further requests come along regarding design of an HAR system such as real-time capability, low energy consumption or unobtrusiveness. In the following chapters particular efforts on AR related to the present work are given.

2.2 Activity Recognition using inertial sensors

One type of inertial sensors, namely accelerometers, have been used most often in related work. Earliest works in AR using accelerometer concentrated on using multiple accelerometers placed on several parts of the subject's body. The work of Bao and Intille [6] in 2004 - now counting more than 1500 citations [19] - contributed significantly to the field of AR. Five bi-axial accelerometers worn on the user's ankle, hip, wrist, arm and thigh were used to collect data from 20 users, who were performing daily activities such as watching TV, brushing teeth, and working at the PC. Time- and frequency domain features along with several classifiers such as C4.5 Decision Tree, decision table, IBL and NB were used. The Decision Tree classifier showed the best performance with an overall accuracy of 84%.
Other research also applying five accelerometers placed on various body locations was made by Mannini and Sabatini [35], Krishnan and Panchanathan [26] and Foerster and Fahrenberg [16]. Some of them tried to distinguish between postures and movements using data collected by ten to thirty volunteers. Tapia et al. [50] incorporated an additional heart monitor observing a slight increase in performance.

Some researchers examined the use of combining accelerometer and other types of sensors. One work of using a single tri-axial accelerometer in addition with an image sensor worn at the subject's waist to recognize nine activities comes from Cho et al. [11]. Choudhury et al. [12] made use of a device containing seven different sensors: accelerometer, microphone, visible light photo transistor, visible+IR light sensor, humidity/temperature sensor, compass and barometer. The aim was to identify the activities *sitting*, *standing*, *walking*, *ascending* and *descending stairs* and *moving up* and *down in an elevator*. Parkka et al. [41] utilized twenty different sensors to recognize daily human movements and sports activities.

Furthermore, tests were carried out using only one accelerometer for recognizing activities. Long et al. [34] collected data in a naturalistic way from twenty-four users wearing an accelerometer on the waist while performing activities like *walking*, *running*, *jogging*, *cycling* and other sports. The orientation of the sensor was irrelevant. Lee et al. [30] chose the left waist as a location for the tri-axial accelerometer. Five users were performing the activities *sitting*, *lying*, *standing*, *walking* and *running*. Ravi et al. [45] used a single device including an accelerometer to collect data from two users. Eighteen different classifiers were compared especially classifier ensembles like bagging and boosting algorithms.

There is also several work based on the use of IMUs where accelerometers and gyro-

scopes are combined. Frank et al. [18] used an Xsens IMU to receive acceleration and angular velocity and benefited from the provided attitude information to rotate them from sensor frame to global frame. The device was mounted on the user's belt while he was performing the activities *sitting*, *standing*, *walking*, *running*, *jumping*, *falling* and *lying*. Nineteen features were computed which are mainly based upon signals in the body frame associated with the human body (z-axis towards the head, xy-plane orthogonal to it, see Chapter 3.2), four different classification algorithms (all Bayesian techniques) were tested and the best one achieved recalls and precisions between 93 and 100% except for falling. Florentino-Liano et al. [15] utilized the database provided from [18] for generating a hierarchical dynamic model with hidden Markov Models. This classification algorithm delivered higher recalls and precisions than the previous work and was supposed to be computationally less expensive. Susi et al. [49] presented an algorithm for the detection of quasi-static instants from handheld MEMS devices. A decision-tree classifier, able to detect activities typical for mobile phone users (*phoning*, *texting*, *walking with swinging hand* or *carrying the device in a bag*) with probability bigger than 90% was designed. Furthermore they investigated if the use of quasi-static instants for detecting pedestrian's step would be possible. Bancroft et al. [5] developed a foot mounted device which combines IMU and GPS measurements. They aimed at classifying the environment (*indoor*, *outdoor*), activities like *stationary*, *crawling*, *walking* or *biking*, and the vertical movement (*level*, *up* or *down elevator*, *up* or *down stairs*). With the use of NB probabilistic models they received excellent results for activity classification, whereas environment classification was less reliable because of the shading of GPS signals.

2.3 Activity Recognition using Smartphones

The use of widely-available mobile devices like cell phones for AR was considered by several researchers. Instead of taking advantage of the sensors included in the cell phones, a separate device was used to measure the signals which were sent via a wireless connection to the mobile phone for processing. Gyorbiro et al. [20] used the MotionBand - including accelerometer, magnetometer and gyroscope - attached to the dominant wrist, hip or ankle to collect data from human motion as well as gesture activities. The data was transmitted via a Bluetooth connection to a smart phone carried by the user. Lester et al. [31] achieved an accuracy rate of 90 % in classifying common human movement. Instead of using a cell phone they collected the data with an external accelerometer-based device worn attached to the wrist, waist or shoulder. Lara and Labrador [29] designed a mobile application for real-time HAR using the Android platform called Vigilante.

They employed a Bluetooth chest sensor strap to measure acceleration and physiological signals like heart rate and skin temperature. Using time and frequency-domain features and the C4.5 Decision Tree classifier they accomplished 96.8 % accuracy in recognizing 5 activities.

In recent years many researchers used cell phones to collect data for AR. The advantage is that they are unobtrusive and do not require additional equipment. Brezmes et al. [9] examined the use of a conventional mobile phone equipped with an accelerometer to recognize human movements in real time. Additionally he laid the focus on the activities *stand-up*, *sit-down* and *falling*. All computations are executed on the phone whereupon an own model has to be trained for each user. Yang [56] made use of orientation-independent features extracted from acceleration magnitude to predict human motion activities. Moreover this work inferred physical activity diaries from classification models. Miluzzo et al. [37] integrated several sensors available in Smartphones such as microphone, GPS, accelerometer, camera and Bluetooth into their system. It should automatically infer people's sensing presence and share it through social network portals such as facebook. Data from ten users was collected to build a model by a C4.5 Decision Tree classifier which had difficulty to differ sitting and standing activities.

Anguita et al. [4] implemented an adaption of the standard SVM to predict human motion activities including *walking upstairs* and *downstairs* which received recall values from 72 to 79 %. Kwapisz et al. [27] combined these two activities in order to receive higher accuracies. Wu et al. [54] used the iPod Touch which includes the same sensors as the iPhone to record data from 16 users wearing the device in an armband for *jogging* and in the front shorts pocket for *sitting*, *walking* and *stair climbing*. They were instructed to perform these activities at normal and brisk paces. The researchers tested several classifiers whereas the k-NN performed best and received benefits when adding gyroscope measurements to the accelerometer readings. Dernbach et al. [13] discovered the benefits of using both accelerometer and gyroscope as well but they failed in identifying complex activities such as *cooking* and *cleaning* besides simple ones. A promising new service called Actitracker was developed by Weiss et al. [52] which is a Smartphone-based activity monitoring service with the aim of helping people to maintain proper health. Assuming that the subject carries the phone in the trouser pocket, five activities *standing*, *walking*, *jogging*, *stairs (up or down)*, and *sitting/lying down* are identified using an impersonal model initially. After the completion of a simple training phase, a personal model is generated automatically which improves recognition rate significantly. Currently they are using only accelerometers but are planning to include gyroscopes.

2.4 Activity Recognition using Smartwatches

The idea of using the wrist as a position for a device to record data and use it for AR was picked up by some researchers several years ago. With the rise of Smartwatches, which are equipped with functionalities in addition to the traditional display of time, there is nothing to stand in the way of the permanent monitoring of activities.

Yang et al. [58] used a tri-axis accelerometer mounted on the dominant wrist to identify eight common domestic activities. They performed a data pre-processing step to separate static and dynamic activities and a feature subset selection to remove redundant information. Using a neural classifier, they obtained an overall accuracy of 95 %. Chernbumroong et al. [10] incorporated an accelerometer sensor in a sport watch to recognize five daily living activities, i.e. *walking*, *running*, *sitting*, *standing* and *lying*. When comparing the two classifiers Decision Tree and ANN, the Decision Tree performed better with 94 % accuracy. Both studies indicate that the use of an on the wrist-worn accelerometer is often not enough to detect more complex activities. Furthermore, activities involving upper body movements by can be distinguished mainly thereby [6].

The online AR system eWatch by Maurer et al. [36] embedding accelerometer, light sensors, microphone and a micro-controller in one device can be worn as a sports watch and is used to recognize six primary activities like *walking*, *standing*, *sitting*, *running*, *ascending* and *descending stairs*. Comparing four types of classifiers, the accuracy and computational costs of the Decision Tree were in a good balance. The overall accuracy was 92.5 % and the execution time for feature computation and classification was less than 0.3 ms.

A final, important work to be mentioned is from Bieber et al. [8]. They examined the capabilities of the sensors provided by Smartwatches and showed the requirements on them to detect activity and inactivity as well as sleep phases and heart rate, when the Smartwatch is held to the chest. A new measurement quantity, the activity unit, which is described in detail in Section 4.5.1, is introduced.

3 Background Theory

This chapter includes detailed information about inertial sensors, the coordinate systems used in this work and classification techniques.

3.1 Inertial Sensors

Inertial sensors measure acceleration and angular velocity and are usually part of Inertial Measurement Units (IMUs) which contain three-axis accelerometers and gyroscopes. They are grouped into two categories which differ in the frame of reference in which the sensors operate - stable platform and strapdown systems. IMUs form the main part of inertial navigation systems (INS) which are self-contained and independent from external disturbances or weather. INS serve to determine position, velocity and attitude of a moving object relative to a known starting point, position and orientation, called dead reckoning, and are used among others in aircraft, spacecraft, missiles, submarines, robotics and image stabilization.

Since the development of small, lightweight and inexpensive Micro-Electro-Mechanical Systems (MEMS) inertial sensors are used in Smartphones for example to re-orient the screen as a user moves the device or for gesture recognition functions.

In the following paragraphs the functionality and different types of inertial sensors (based on [21],[51]) are presented.

Accelerometers The principle of accelerometers basically lies in measuring the forces acting upon a proof mass. Generally there exist two types of sensor architectures: open-loop and closed-loop. On the one hand external forces acting on the sensor, the specific force, cause a displacement of the proof mass which is measured by open-loop accelerometers. On the other hand the proof mass stays in a state of equilibrium as closed-loop accelerometers generate a force that opposes the specific force.

Besides the division concerning the sensor architecture there exist different forms of construction or design: mechanical and "solid-state" accelerometers. The first consist of

a proof mass suspended in a case and confined to a zero position by springs or pendulums. The deflection of the proof mass is measured using a pickoff mechanism like a mechanical scale or an optical detector and is proportional to the specific force acting along the input axis. Solid-state accelerometers can be subdivided into various categories including surface acoustic wave, silicon, vibratory and quartz devices, and constitute sensors which are small, reliable and rugged.

Gyroscopes Gyroscopes, or briefly gyros, sense the angular rate of turn about some defined axis. Traditionally two types are distinguished: mechanical gyros, which can be sub-grouped into rotational and vibratory gyroscopes, and optical ones, which can be split into Ring Laser Gyros (RLG) and Fiber Optic Gyros (FOG). Mechanical gyros operate in gimbaled platforms and optical ones in analytical systems. Rotational gyroscopes principally consist of a rotational-symmetric body such as a wheel mounted on two gimbals and spinning at high speed. The gravity field of the earth has no effect on the wheel because it is suspended with minimal friction symmetrical to its centre of mass. The spinning wheel tends to maintain the spin axis in inertial space. If external forces are acting on the gimbals, the wheel remains at its global orientation and only the angles between the gimbals change. The rotations of the gimbal frames relating to the spin axes can be measured using angle pick-offs. Optical gyroscopes make use of the Sagnac effect, where the light paths of two laser beams travelling in opposite direction around an enclosed ring differ in length when the ring is rotated around the axis orthogonal to the light path. This difference can be measured in two ways. The resonator principle utilizes the standing wave that stays fixed in space when the path length is chosen appropriately and is used in RLGs. The interferometric principle is based upon a recombination of the laser beams, which are interfered due to the phase shift when leaving the ring coil. The resulting intensity depends on the angular velocity which can be measured there from. This principle is used in FOGs.

Magnetometer A magnetometer is a sensory device for measuring magnetic flux densities which are given in units of Tesla (T). It determines the direction to the magnetic north pole, whereby the angle between the Earth's magnetic field and a certain axis of the vehicle or device, where the sensor is attached to, can be calculated. Magnetometers are usually used together with accelerometers and gyroscopes to contribute the direction to the north pole.

MEMS inertial sensors MEMS are miniaturized components which combine logic elements and micro-mechanical structures in a chip usually made of silicon. The structures may be less than one micron and thanks to the miniaturization they can be produced cheaply and in high volume. They are rugged, low weight, have a low power consumption and a short start-up time. The main disadvantage is that MEMS inertial sensors cannot match the accuracy of sensors manufactured using traditional techniques.

Errors In traditional as well as in MEMS inertial sensors errors may occur, in the worst case systematic sensor errors. A bias of the accelerometer leads to an offset in the data and a bias in the rotation rate causes a drift. Low-cost inertial sensors lie in the lowermost accuracy class.

3.2 Coordinate Systems

When measuring acceleration and angular rate with three-axis inertial sensors the coordinates refer to a three-dimensional coordinate system associated with the sensor, called sensor frame (SF), the origin of which is situated within the object.

Another important frame is the local-level frame (LLF). Its origin is defined anywhere on or near the earth's surface, called topocentre, and its axes are, e.g., pointing to north (x^{LLF}), to west (y^{LLF}) and to local zenith (up, z^{LLF}), thus it is defined as a right-handed reference frame. In the literature, there exist right-handed and left-handed systems, with the z-axis either pointing local downwards or upwards. To avoid mistakes, it should carefully be defined if a right-handed or left-handed system is used.

The parameters for describing the attitude of the object in the SF relative to the LLF are the rotation-angles about the three axes, named roll, pitch and yaw.

The third and final coordinate system which is used in this work is the so-called body frame (BF), which is fixed to the subject's body. Its centre of mass is approximately situated at the waist and the directions of the axes, relative to the body, are forward (x^{BF}), left (y^{BF}) and up (z^{BF}).

Florentino-Liano et al. [15] proposed an algorithm to transform the coordinates from the SF to the defined BF. The assumptions for doing this are that the sensor is placed in a fixed location near the waist, e.g., on a belt or in the trouser pocket and that it does not move except for a few millimetres during the signal recordings. In periods where there is no linear acceleration (i.e., no movement of the body), the orientation of the SF in relation to the LLF can be determined. The roll and pitch angles can be calculated

with the following formulas:

$$roll = \arctan\left(\frac{\overline{\alpha}_y^S}{\overline{\alpha}_z^S}\right) \tag{3.1}$$

$$pitch = \arcsin\left(\frac{-\overline{\alpha}_x^S}{\sqrt{(\overline{\alpha}_x^S)^2 + (\overline{\alpha}_y^S)^2 + (\overline{\alpha}_z^S)^2}}\right), \tag{3.2}$$

where $\overline{\alpha}_i^S$ is the mean acceleration of axis i in the sensor frame.
When the subject is standing in an upright position for a few seconds, the LLF can be equalized with the BF, whereupon small deviations are negligible as they are in the magnitude order of the measurement errors. Therefore the above estimated values describe the inclination of the sensor in relation to the BF. The yaw angle is not required, because it is irrelevant in which direction the subject is moving. The aim is to use the norm, a vertical component (the z-coordinate) and a horizontal component, derived from x- and y-coordinate, which is independent from the yaw angle. Therefore the reduced rotation matrix from SF to BF can be introduced with the yaw angle assumed to be zero:

$$R_{SF}^{BF}(roll, pitch) = \begin{bmatrix} \cos(pitch) & \sin(roll)\sin(pitch) & \cos(roll)\sin(pitch) \\ 0 & \cos(roll) & -\sin(roll) \\ -\sin(pitch) & \sin(roll)\cos(pitch) & \cos(roll)\cos(pitch) \end{bmatrix} \tag{3.3}$$

Under the assumption that the sensor position does not vary over time, the rotation matrix stays constant and can be applied to the measured signals which are given in the SF.

To obtain coordinates in the LLF in general for an arbitrary posture of the body, i.e., while performing different activities, a Kalman-filter-based estimation can be applied. Considering the basic observation equation

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k, \tag{3.4}$$

where $\mathbf{x}_k$ represents the parameters and $\mathbf{z}_k$ and $\mathbf{v}_k$ represent the measurements and the observation noise, roll and pitch angles are calculated with the formulas (3.1) and (3.2) and put into the measurement vector $\mathbf{z}_k$. Furthermore they are used for the rotation of angular rates delivered from the gyro, which are interpreted as derivations of roll and pitch angles, also included in the measurement vector $\mathbf{z}_k$:

$$\mathbf{z}_k = \left[roll, \dot{roll}, pitch, \dot{pitch}\right]^T = \mathbf{x}_k. \tag{3.5}$$

The state vector $\mathbf{x}_k$ includes the same quantities as the measurement vector. Therefore the design matrix $\mathbf{H}_k$ is simply the identity matrix. The transition matrix in the dynamic

model

$$\mathbf{x}_{k+1} = \mathbf{\Phi}_k \mathbf{x}_k + \mathbf{w}_k \tag{3.6}$$

is built under the assumption of a uniform motion, as carried out in [21, chap. 13.4]. The parameters in the measurement and system noise have to be adjusted in order to get reliable results. The estimated roll and pitch angles are then used in a rotation matrix with the same structure as in Eq. (3.3) to rotate the accelerations measured in the SF into the LLF.

An overview of the introduced coordinate frames is given in Fig. 3.1.

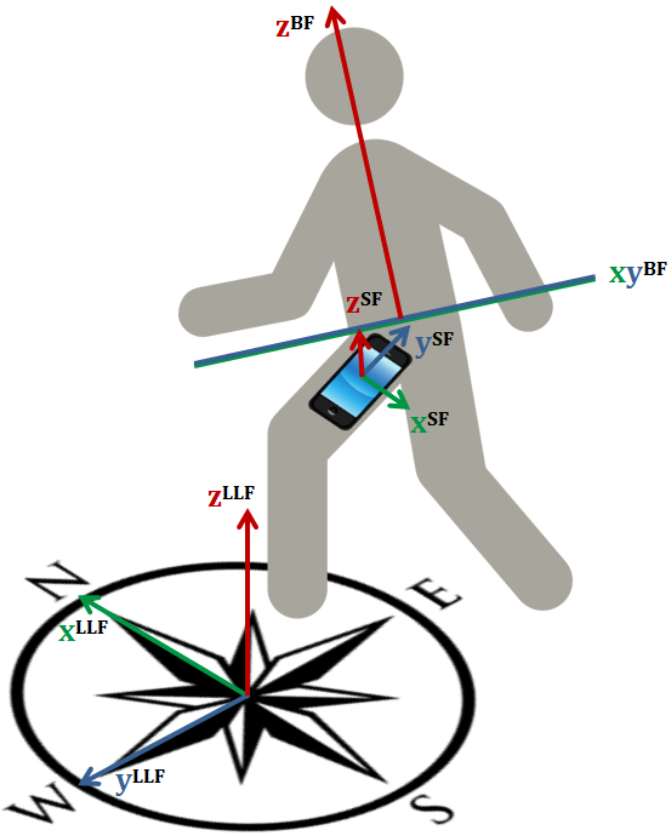


Figure 3.1: Coordinate frames - SF, LLF, BF.

3.3 Classification Techniques

AR is based on machine learning techniques. Depending on the availability of labelled data (i.e., the output is known) different algorithms are distinguished [28].

- Supervised learning algorithms are trained on labelled data with the aim of learning a general rule to map input to output and generate a classifier.
- Unsupervised learning, called clustering, is carried out if the features have no labels, for example when labelling the data is not feasible. Sometimes it is used to discover hidden patterns in data and therefore detect new classes.
- In semi-supervised learning both labelled and unlabelled features are combined to generate an appropriate classifier.

The labelled data which are used for supervised learning are called training set. For testing the generated classifier a part of the data set may be used as test set. More details on the evaluation of classifiers is given in Section 3.3.2. The following chapter includes detailed information about some classifiers whereupon the focus is directed on those who have been used in this work.

3.3.1 Classifiers

Some of the most commonly used supervised learning methods, as stated by Labrador and Yejas [28], will be explained briefly below. Decision Trees (DT), k-Nearest Neighbour (k-NN), Support Vector Machine (SVM) and Naive Bayes (NB) are treated. The section on DTs is based on Quinlan [43] and is very detailed as these are used as the main classifiers in the present work.

Decision trees

A DT is a classifier with a special structure containing

- leafs, which correspond to classes,
- decision nodes, specifying a test to be carried out on a single feature of an instance, with
- one branch and subtree for each possible outcome of the test [43].

It can be used for classifying an instance by starting at the root node of the tree and moving through it by means of its feature values until a leaf is reached.

Construction The process of generating a DT from a training set is based on the original idea from Hoveland and Hunt in the late 1950s. Their divide and conquer method for the construction of a DT from a training set T involving the classes $\{C_1, C_2, \ldots, C_k\}$ distinguishes three possibilities [43]:

- T contains instances, all belonging to a single class $C_j \rightarrow$ the DT is a leaf,
- T contains no instances $\rightarrow$ the DT is a leaf but the class must be determined from information other than T,
- T contains instances belonging to a mixture of classes $\rightarrow$ idea of refining T into subsets of instances which belong to single classes. A test, based on a feature, is chosen with one ore more exclusive outcomes $\{O_1, O_2, \ldots, O_n\}$. Thereby T is separated into n subsets where the subset T_i contains all instances in T which have the outcome O_i as a test result. This procedure is repeated successively to each subset of the training set until the subsets consist of instances of the same class or with a reasonable number, the default minimum is 2.

Example The construction process is illustrated by an exemplary data set "weather" whose objective is to estimate the likelihood of playing golf based on given weather conditions. The four features of 14 instances in the training set are given in Tab. 3.1. Each instance either belongs to class *Play* or *Don't Play*. Since there are instances in the hole set T belonging to different classes, the algorithm splits it into subsets. Under the assumption that *outlook* has been chosen as test feature, it has three outcomes (*sunny*, *overcast* and *rainy*) building three subsets T_i, stated in Tab. 3.1. The second group is the only one including instances with one class, hence the other two subsets are further divided using the test *humidity* and *windy*, as can be seen in Fig. 3.2.

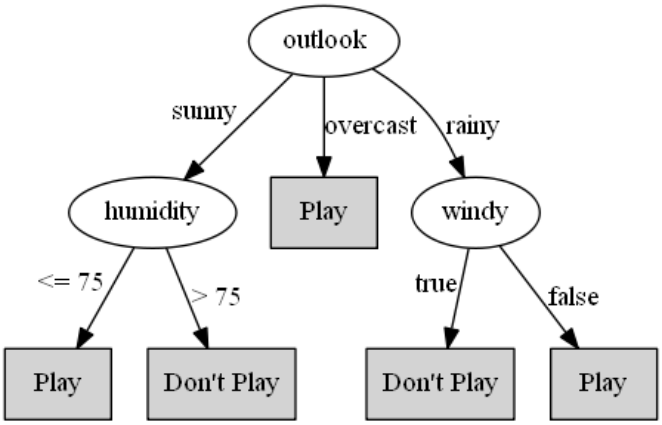


Figure 3.2: Constructed decision tree for the example "weather".

Table 3.1: Small training set "weather".

subset	Outlook x1	Temp (°F) x2	Humidity (%) x3	Windy? x4	Class y
T1	sunny	75	70	true	Play
	sunny	69	70	false	Play
	sunny	80	90	true	Don't Play
	sunny	85	85	false	Don't Play
	sunny	72	95	false	Don't Play
T2	overcast	72	90	true	Play
	overcast	83	78	false	Play
	overcast	64	65	true	Play
	overcast	81	75	false	Play
T3	rainy	71	80	true	Don't Play
	rainy	65	70	true	Don't Play
	rainy	75	80	false	Play
	rainy	68	80	false	Play
	rainy	70	96	false	Play

This is not the only way to split the data into subsets. Two questions arise: What is the justification for taking for example *outlook* as the first test feature? Why not inspect all possible trees?

The reason is that the tree-building process does not intend to find any partition of the data set, but to construct a tree with predictive power and with as few blocks as possible [43]. Moreover the problem of finding the smallest tree is NP-complete. Therefore DT algorithms are non-backtracking, greedy algorithms, where the impacts of alternative choices are not analysed. Nevertheless a criterion has to be found by which is decided what test is applied.

Gain (ratio) criterion There exist many implementations of DTs, for example Classification and Regression Tree (CART) systems from Breiman, or ID3 and C4.5 from Quinlan. Hunt recommended an entry via information theory, so the ID3 algorithm uses the so-called gain criterion. There is one statement of the information theory [43]: "*The information conveyed by a message depends on its probability and can be measured in bits as minus the logarithm to base 2 of that probability.*" This leads to a definition as it

is given in Eq. 3.7:

$$H(S) = -\sum_{j=1}^{k} P(s_j) \cdot \log_2 P(s_j). \tag{3.7}$$

The *information entropy* $H(S)$ of a set S is defined as the average number of information needed to determine the class of an instance in S. $P(s_j)$ describes the probability of the information "message" and can be computed with

$$P(s_j) = \frac{|\{\mathbf{x} \in S | y = C_j\}|}{|S|}, \tag{3.8}$$

which is the number of instances in S, which belong to class C_j divided by the number of all instances in set S. The instances have the form $(\mathbf{x}, y) = (x_1, x_2, x_3, ..., x_p, y)$ where x_a is the value of the a^{th} feature of the instance $\mathbf{x}$ and y is the corresponding class label. If T is partitioned with n outcomes $(O_1, \ldots, O_n)$ because of test (feature) a, the required information by using this test is

$$H(T|a) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \cdot H(T_i), \quad \text{with} \quad T_i = \{\mathbf{x} \in T | x_a = O_i\}. \tag{3.9}$$

The *gained information* IG by using the test a to partition the set T is calculated with the quantity

$$IG(T, a) = H(T) - H(T|a), \tag{3.10}$$

that measures the *information relevant to classification*. The gain criterion finds the test which maximizes the information gain.

In the C4.5 Decision Tree algorithm, the criterion has been refined and extended by the *intrinsic value* IV, which shows the potential *information achieved by splitting* T into n subsets:

$$IV(T, a) = -\sum_{i=1}^{n} \frac{|T_i|}{|T|} \cdot \log_2 \left(\frac{|T_i|}{|T|} \right) \tag{3.11}$$

Thus the *information gain ratio* IGR relates the information apparently helpful for classification to that one generated by the split:

$$IGR = \frac{IG}{IV}. \tag{3.12}$$

Based on this criterion the classification tree is created, namely it is used to evaluate a set of tests, which may take one of the forms [43]:

- "standard" test on a discrete feature, with one outcome and branch for each possible value.

- binary test on a continuous feature, with outcomes $A \leq Z$ and $A > Z$, where the feature value A is compared to a threshold Z.

In the second case, the training set T is sorted on the finite values of feature A, $\{v_1, v_2, \ldots, v_m\}$. Any threshold between v_i and v_{i+1} divides the values into $\{v_1, v_2, \ldots, v_i\}$ and $\{v_{i+1}, v_{i+2}, \ldots, v_m\}$. So there are only $m - 1$ possible values for the threshold and they are either chosen as the midpoint of each interval

$$Z_i = \frac{v_i + v_{i+1}}{2}, \tag{3.13}$$

or, as the C4.5 algorithm does, as the largest value of A not exceeding the midpoint above, ensuring that all threshold values appear in the data set.

Overfitting A learning algorithm is said to overfit relative to a simpler one if it is more accurate in fitting known data but less accurate in predicting new data. In cases where learning was performed too long or where the training dataset is small, the performance on the training set still increases while it becomes worse on unknown data. There are two common approaches to avoid overfitting of training data [25]:

- stop the training algorithm before it reaches a point at which it perfectly fits the training data, or
- prune the DT.

The second approach is slower but more reliable.

k-Nearest Neighbour (k-NN)

k-NN is an instance-based learning algorithm which requires less computation time during the training phase than eager-learning algorithms (like DTs) but more computation time during the classification process [25]. The principle of this classifier is that instances within a dataset with similar properties will be located close to each other. This algorithm interprets every instance as a point in a n-dimensional space where each feature corresponds to one dimension. The relative distance between the k nearest instances and the query instance is measured using distance metrics such as the Euclidian, Hamming or Chebychev distance. The most frequent class label of the k nearest instances is then identified as the class of the query instance.

Support Vector Machine (SVM)

A SVM constructs a hyperplane or set of hyperplanes in a high-dimensional space, which linearly separates data of two different classes. The hyperplanes are constituted by so-called support vector points and the solution is only a linear combination of these. They are part of the training data that lie on the hyperplane's margin, other points of the training dataset are ignored. Hence, the number of training instances is unimportant as well as the number of features in relation to the size of the training set. The maximum margin allows the SVM algorithm to select one of the candidate hyperplanes. When the data can not be separated linearly, they are mapped into a higher-dimensional space, where the separating hyperplane is defined. Multi-class problems must be reduced to a set of multiple binary classification problems [25].

Naive Bayes (NB)

NB is a special kind of Bayesian Network (BN). It is a simple probabilistic classifier based on Bayes' rule with strong independence assumptions between the features, i.e., it naively assumes that the value of a particular feature is unrelated to the presence or absence of any other feature, where the class variable is given.

The following formula describes the probability model over a dependent class variable C conditional on several feature variables F_1 through F_n using Bayes' rule:

$$p(C|F_1, F_2, \ldots, F_n) = \frac{p(C)p(F_1, F_2, \ldots, F_n|C)}{p(F_1, F_2, \ldots, F_n)} \tag{3.14}$$

The numerator corresponds to the joint probability which can be further subdivided into conditional probabilities

$$\begin{aligned} p(C)p(F_1, F_2, \ldots, F_n|C) &= p(C, F_1, F_2, \ldots, F_n) \\ &= p(C)p(F_1|C)p(F_2|C, F_1)\ldots p(F_n|C, F_1, \ldots, F_{n-1}) \end{aligned} \tag{3.15}$$

Assuming the conditional independence of features F_i and F_j $(i, j = 1, \ldots, n,\ i \neq j)$ it simplifies to

$$p(C)p(F_1, F_2, \ldots, F_n|C) = \frac{1}{Z}p(C)\prod_{i=1}^{n} p(F_i|C). \tag{3.16}$$

The denominator $Z = p(F_1, F_2, \ldots, F_n)$, called evidence, is constant because it does not depend on C and the feature values F_i are given. The NB classifier combines this model with a decision rule, for example picks the most probable hypothesis, the class C, where the model is maximized.

Despite the naive design, NB classifiers have worked quite well when tested on actual datasets and only require a small amount of training data to estimate the parameters (means and variances) necessary for classification, linear to the number of features [53].

3.3.2 Evaluation

For the evaluation of a classifier the entire dataset is split into a *training* and a *test set*. The classifier learns from the training set whereas the test set is presented to the classifier without labels, i.e. without the *ground-truth* which then predicts the class of each instance.

Partitioning the data set is a critical issue. It may occur that instances are concentrated in a particular feature space subregion, then the classifier performance would not be reflected by the evaluation metrics [28]. For this reason a *k-fold cross validation* is preferred. Thereby the dataset is split into k equally-sized parts, where one part is used for testing and the $k-1$ parts for training. This procedure is repeated k times, so that each fold is used as test set once. The validation results are averaged over the iterations. A 10-fold cross validation is commonly used. The selection of data samples should be done in stratified manner, so that the class distribution in the test set is similar to the training set.

The evaluation results can be presented in a *confusion matrix* $M_{n\times n}$, where n is the number of classes. M_{ij} is the number of instances that are belonging to class i (the ground-truth) and were classified as class j. An example is given in Tab. 3.2.

Table 3.2: Example for a confusion matrix with three classes

		Classification output		
		apple	cherry	banana
Ground truth	apple	30	3	2
	cherry	2	25	0
	banana	4	2	27

In a binary classification problem with two classes there are four parameters that can be defined and computed from the confusion matrix.

- *True Positives* (TP): the number of positive instances (i.e., relating to the considered class) classified as positive (that means correctly).
- *True Negatives* (TN): the number of negative instances (i.e., relating to the disregarded class) classified as negative (correctly).

- *False Positives* (FP): the number of negative instances classified as positive (i.e., incorrectly classified as the considered class).
- *False Negatives* (FN): the number of positives instances classified as negative (i.e., incorrectly classified as the disregarded class).

These quantities can be generalized for problems with n classes, whereat an instance is positive according to a class, e.g., *apple* and negative in line with all other classes except for *apple*. For better understanding see Tab. 3.3.

Table 3.3: Definition of True Positives (TP), True Negatives (TN), False Negatives (FN) and False Positives (FP) considering class *apple*, number of instances in brackets.

		Classification output	
		apple	others
Ground truth	apple	TP (30)	FN (5)
	others	FP (6)	TN (54)

Various metrics can be calculated when combining these parameters [28]. The most important are:

- ***Accuracy***
  is the most standard quantity to show the overall classification performance considering all k classes and is defined through the number of correctly classified instances (situated in the diagonal of the confusion matrix M_{ij}) compared to the total number of instances:

$$Accuracy = \frac{\sum_{i=1}^{k} M_{ii}}{\sum_{i=1}^{k} \sum_{j=1}^{k} M_{ij}} \tag{3.17}$$

- ***Precision*** (or Positive Predictive Value)
  is the ratio of correctly classified instances relating to a specific class to the total number of instances classified as this class:

$$Precision = \frac{TP}{TP + FP}. \tag{3.18}$$

- ***Recall*** (or True Positive Rate, Sensitivity, Hit Rate)
  is the ratio of correctly classified instances relating to a specific class to the total number of instances belonging to this class:

$$Recall = \frac{TP}{TP + FN}. \tag{3.19}$$

In addition to these, there exist, e.g., the *classification error*, the *F-measure* which combines precision and recall, the *False Positive Rate* and the *False Negative Rate*.

Comparison of classifiers In order to choose the most accurate classifier, a 5×2-fold cross validation with a paired t-test is suggested ([28],[14]). Thus, five replications of a 2-fold cross validation are performed, where the dataset is randomized for each replication before it is partitioned into two equal-sized sets. This can be achieved by using different seeds for the random number generator. When comparing two classification algorithms, a statistical paired t-test can be applied. The null hypothesis is that both algorithms have the same accuracy or error rate. Defining $p_i^{(j)}$ as the difference between the accuracies (or error rates) of both classifiers in the i-th iteration and j-th fold, the test statistic $\tilde{t}$ of the t-test is defined as:

$$\tilde{t} = \frac{p_1^{(1)}}{\sqrt{\frac{1}{5}\sum_{i=1}^{5} s_i^2}}, \tag{3.20}$$

with $s_i^2 = (p_i^{(1)} - \bar{p})^2 + (p_i^{(2)} - \bar{p})^2$ as the variance of the i-th iteration and $\bar{p} = \frac{p_i^{(1)} + p_i^{(2)}}{2}$. $\tilde{t}$ is approximately Student-t distributed with 5 degrees of freedom.

4 Development

In the following chapter, the individual steps are explained, which were necessary in order to perform AR. First, the desired activities, the used equipment and the data recording are described in detail. Then the explanation of the data processing follows including feature calculation. The final step consists in the selection of the features and the actual classification. All calculations were performed in MATLAB, version R2012b.

4.1 Activities

The activities which were desired to recognize are human motion activities. They are listed below. The abbreviations used in several graphics and tables in this work are stated in brackets.

- Walking on level ground or merely *walking* (WA)
- Walking downstairs or in short *downstairs* (DS)
- Walking upstairs or briefly *upstairs* (US)
- *Running* (RU)
- *Standing* (ST)
- *Sitting* (SI)
- *Lying* (LY)

Depending on the location of the measurement device, not all activities have been used in each investigation in Chapter 5. In addition, the motion activities were performed in fast and slow manner. The abbreviations are then extended by a letter "F" or "S", e.g. *walking fast* is shortened by WAF. In the first essential investigation, two additional activities were examined, which were furthermore disregarded because they were considered too specific:

- *Jumping* (JU)
- *Falling* (FA)

4.2 Measurement Devices

AR was investigated on the basis of a medium-priced IMU, the Xsens MTi-G, and evaluated and tested with low-cost sensors, Smartphone Samsung Galaxy Nexus and Smartwatch Omate TrueSmart. The three devices are explained in more detail below.

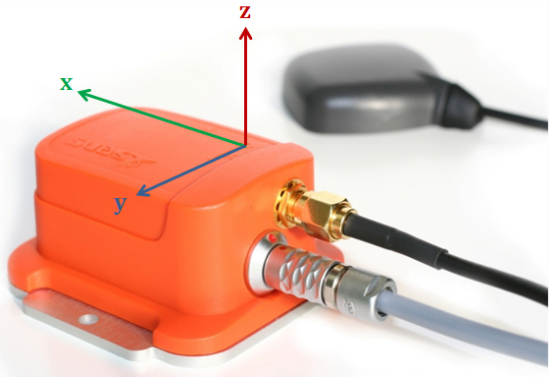


Figure 4.1: Xsens MTi-G with marked sensor axes.

Xsens The MTi-G is an integrated GPS and IMU with a Navigation and Attitude and Heading Reference System (AHRS) processor manufactured by Xsens Technologies [55]. It includes a miniature GPS receiver, MEMS inertial sensors and an additional 3D magnetometer and pressure sensor. The MTi-G provides 3D position and velocity estimated by sensor fusion using a real-time Kalman filter. In addition, 3D orientation estimates, as well as 3D acceleration, 3D angular rate, 3D earth-magnetic field data and static pressure are delivered. The 3D data are expressed in coordinates of a right-handed coordinate frame, the sensor frame (SF), which is aligned to the housing of the device, see Fig. 4.1. The units for acceleration and angular rate output are m/s^2 respectively rad/s. The output unit for the Magnetic field is an arbitrary unit normalized to earth field strength. The orientation between the SF and the local level frame (LLF) is output as Euler angles roll, pitch and yaw. The default sampling frequency of 100 Hz is retained in this work.

Smartphone The Samsung Galaxy Nexus is a touch-screen Smartphone co-developed by Google and Samsung Electronics with Android 4.0 as operating system. It was released in 2011 and includes an accelerometer as well as a gyroscope and magnetometer, for example, for rotating the screen and as a compass. The coordinate axes of the sensors are aligned with the housing and are defined as in Fig. 4.2.

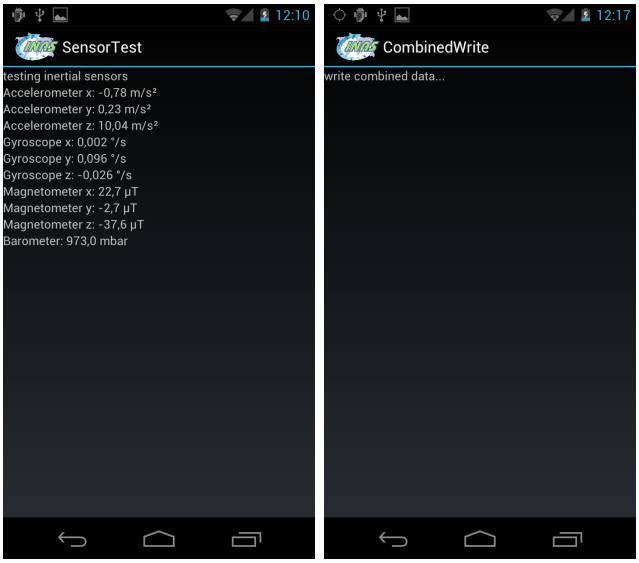
Figure 4.2: Smartphone Samsung Galaxy Nexus with marked sensor axes.

According to a survey in 2014, approximately 72 % of mobile phone users in Austria currently own a Smartphone. In 2010, there were only 32 % [46]. It is expected that the number will continue to rise and almost everyone will own a Smartphone with comparable and more sensors in the coming years.

A short application for Android, called SensorTest, shows the current inertial sensor data on the display. A further application, called CombinedWrite, enables the sensor data readout and storage. It delivers three files containing the inertial sensors output, the WiFi data and GPS positions with UTC time stamps. The inertial sensor signals are output with approximately 28 Hz. A screenshot of the two applications is given in Fig. 4.3.

Smartwatch The Omate TrueSmart is a Smartwatch developed by Omate and has been funded by crowd funding via Kickstarter. It is a standalone Smartwatch based on Android and can be used to make calls, navigate and use Android applications independent of the user's Smartphone, although it may also be used as a second screen of the phone. It includes sensors that are comparable to those of the Smartphone Samsung Galaxy Nexus. The coordinate axes are defined as in Fig. 4.4 and the applications described above can be applied in the same way. The output frequency is 100 Hz for the inertial sensors.

The willingness to use a Smartwatch is growing. Two out of five people in Germany would carry a Smartwatch [47]. For 25 percent of German interviewees the collection of sports, health and fitness values is the most important application, followed by the function as a classical clock (20 percent) and the ability to make phone calls (11 percent) [48].



(a) SensorTest (b) CombinedWrite

Figure 4.3: Applications for readout and storage of sensor data on Android.

Figure 4.4: Smartwatch Omate TrueSmart with marked sensor axes.

The three devices are hereinafter referred to as Xsens, Smartphone or Nexus, and Smartwatch or Omate.

4.3 Data Collection

Various experiments were carried out where one subject was wearing one or two of the described devices on a specific location on the body. Most of the measurements were done by the author Karin Wisiol, in the following referred to as Karin, one set of training and testing data was collected by Thomas Moder, hereinafter called Thomas. The subjects performed different activities in a semi-naturalistic way inside the campus of Steyrergasse 30. A second person was manually labelling the activities by recording the start and stop times in UTC. More detailed information on training data collection is given hereafter.

Xsens/Smartphone on the belt Xsens and Smartphone were placed on a fixed location on the belt on the right part of Karin's body. Both devices were attached firmly to ensure a stable position at the body. To facilitate a synchronization of the data, GPS time stamps were recorded additionally. A second Smartphone with internet connection was used to display the actual UTC time for taking handwritten notes of the time epochs when the activity started and ended. The data set was collected within 2 days (18.04.2014 and 21.04.2014) and contains about 1.5 hours of activity data. In addition to the seven contemplated activities, *jumping* and *falling* were registered.

Smartphone in the hand in front of body About 20 minutes of inertial sensor data each were collected by Thomas (15.07.2014) and Karin (18.07.2014) holding the Smartphone in the right hand in front of the body in breast height while they were performing four activities (*standing*, *walking*, *upstairs*, *downstairs*). The aim was to find differences between several persons and to provide an AR model to be used as a support of a 3D indoor positioning algorithm with pedestrian dead reckoning based on Bayes filtering [38].

Smartphone in right hand in front of body and Smartwatch on left wrist An additional data set was recorded by Karin (28.07.2014), where the four just mentioned activities plus *running* were carried out at different speeds - fast and slow, using the Smartphone in the right hand in front of body and the Smartwatch on the left wrist.

Smartwatch on left wrist Wearing the Smartwatch on the left wrist, five activities (*standing*, *walking*, *upstairs*, *downstairs*, *running*) were performed and recorded within 15 minutes on 10.09.2014 by Karin. The arm was swinging in a natural way without

carrying any objects in the hands. It was not considered to register other static activities like *sitting* or *lying* since such activities are very difficult to relate with an arbitrary wrist position.

Smartphone in trouser pocket Around half an hour of data, wearing the Smartphone in the right trouser pocket, was collected by Karin on 07.08.2014. The certainly most common location on the body was chosen to record signals from the activities *standing*, *walking*, *upstairs*, *downstairs*, *running*, *sitting*, and *lying* since this part of the body reflects the motion and position in the best way.

Smartphone at different body locations To examine the effects of different sensor positions on the body, an additional measurement was carried out by Karin (01.08.2014). While *walking*, the Smartphone was located either in the hand in front of the body, in the hand swinging, in the trouser pocket or in the jacket's interior pocket. The idea was to incorporate a classifier, before the actual activity recognition, which determines the sensor position and due to this takes the appropriate classification model.

4.4 Data Preprocessing

To synchronize the collected data from Smartphone and Xsens a common time unit had to be found. Xsens provides time stamps in the format `[year,month,day,decimal seconds of day]` whereas the Smartphone delivers UTC milliseconds. MATLAB's built-in function *datenum* generates the whole and fractional number of days from a fixed, preset date (January 0, 0000) which served as a common format.

The ground-truth which includes time stamps and activity labels was transcribed to Excel in order to import its data automatically with MATLAB. The time stamps were converted to the new time format as well.

The start and stop times were adapted by examining the time series of the norm of the collected acceleration data graphically. Thereby the particular segments of the staircase could be separated to enable an exact differentiation between the favoured activities.

4.5 Feature Computation

The calculation of the norm of the three-dimensional measurements is the most common and meaningful way of feature computation as the sensor orientation in reference to the body and surrounding is unknown.
As walking on stairs (*upstairs* and *downstairs*) and walking on level ground (*walking*) turns out to be difficult to distinguish, the collected accelerometer data were transformed from sensor frame to local level frame (LLF) using a Kalman filter to estimate *roll* and *pitch* for each epoch [38].
A differentiation between static activities like *standing*, *sitting* and *lying* is possible by using features in the body frame (BF). A detailed description on these transformations is given in Section 3.2. At the beginning of each series of measurements, the subject's first activity was *standing* for at least five seconds. The accelerometer measurements in this time period could be used to approximately calculate a mean value for *roll* and *pitch* and take them to transform all measurement data of the test series into the defined BF.

The following components were used as a basis for feature computation (the abbreviations a, ω and m stand for acceleration, angular velocity and magnetic flux density):

- Norm ($\|a\|$, $\|\omega\|$, $\|m\|$),
  where e.g. $\|a\| = \sqrt{a_x^2 + a_y^2 + a_z^2}$,
- Horizontal Component in BF (a_{xy}^B, ω_{xy}^B),
  where $a_{xy} = \sqrt{a_x^2 + a_y^2}$,
- Vertical Component in BF (a_z^B, ω_z^B),
- Horizontal Component in LLF (a_{xy}^{LL}),
- Vertical Component in LLF (a_z^{LL}).

The duration of human activities is relatively long (about seconds or minutes) whereas the sampling rate of the sensor is very high (up to 100 Hz). To obtain sufficient information to describe the performed activity, a single sample on a specific time instant is not suitable. Hence, a time window basis instead of a sample basis is used to recognize activities. In order to receive quantitative measures, statistical methods are used to extract features [28] (e.g., the mean value).

To provide adequate information about the activity, the approximate duration of one step (around 1 second) is used as a guideline for a window length. The samples within each window are used to extract one value for each type of feature. Corresponding to the sampling frequency of each device, different window lengths are used, see Tab. 4.1.

The chosen sizes of the windows allowed fast computation of Fourier coefficients using Fast Fourier Transformation (FFT). In addition, features are computed every quarter second, i.e., at 4 Hz, so their windows overlap each other.

Table 4.1: Different window sizes used in this work.

Device	**Sampling frequency [Hz]**	**Window [samples]**	**Seconds**	**Overlap**
Xsens	100	128	1.28	80.47 %
Smartphone	28	16	0.57	56.25 %
		32	1.14	78.13 %
		64	2.28	89.06 %
		128	4.56	94.53 %
Smartwatch	100	64	0.64	60.94 %
		128	1.28	80.47 %
		256	2.56	90.23 %

Based on the components mentioned above, features were computed in the time and frequency domain: maximum value (MAX), mean (MEAN), standard deviation (STD), root mean square (RMS), interquartile range (IQR), main frequency component (MFC) [18], amplitude of MFC, arc tangent of the ratio of vertical and horizontal component, and the so called Activity Unit (AU) [8]. Hereafter, a more detailed description of the extracted features is given.

4.5.1 Activity Unit

This feature was introduced as a new measurement quantity by some researchers from Fraunhofer IGD who study AR on Smartwatches [8]. They consider the acceleration sensor as the most important for AR. The AU defines the mean acceleration of the sensor in the three-dimensional space per second. It describes the acceleration force and is given in m/s^3. In contrast to the jerk, the AU denotes the difference of the current acceleration to the average acceleration. It is defined as follows:

$$AU = \frac{1}{N} \sum_{n=1}^{N} \sqrt{(x(n) - x_{mean}(n))^2 + (y(n) - y_{mean}(n))^2 + (z(n) - z_{mean}(n))^2} \quad (4.1)$$

The variables $x(n), y(n)$ and $z(n)$ are the sensor values of the n^{th} sample of the window with length N. x_{mean}, y_{mean} and z_{mean} describe the averaged sensor values of each axis

which are determined by a moving average (y_{mean} and z_{mean} are analogue):

$$x_{mean}(n) = x_{mean}(n-1) \cdot a + x(n) \cdot (1-a), \tag{4.2}$$

where $x(n)$ is the current sensor value, a is an absolute average factor ($a = 0.95$) and $x_{mean}(n)$ is the calculated average. Considering the mean acceleration of each axis for the AU value allows one to ignore the constant sensor offset and the constant gravity influence [8].

The computation can be executed in any reference frame, because it is independent from axis orientation. The AU was not only computed with acceleration values but also with angular rates (ω) and magnetic flux density (m) to test for possible benefits.

4.5.2 Maximum, Mean, Standard deviation, root mean square, interquartile range

The next features have also been used by Frank et al. [18]. They are computed on the basis of the above mentioned components, in the following formulas denoted as u, e.g., the horizontal component of the angular rate in the BF. A detailed description is given in the list below:

- MAX
  It is the largest element of the set of all values in the sampling window:

  $$MAX(u) = \max(\{u(1), \ldots, u(N)\}). \tag{4.3}$$

  $u(k)$ is the k^{th} sample within the window with a size of N.

- MEAN
  The averaged value of the elements (u) within the window is defined as

  $$MEAN(u) = \bar{u} = \frac{1}{N} \sum_{k=1}^{N} u(k). \tag{4.4}$$

- STD
  The standard deviation is commonly defined as:

  $$STD(u) = \sigma(u) = \sqrt{\frac{1}{N} \sum_{k=1}^{N} (u(k) - \bar{u})^2} \tag{4.5}$$

- RMS
  The Root Mean Square is given by:

  $$RMS(u) = \sqrt{\frac{1}{N} \sum_{k=1}^{N} u(k)^2} \tag{4.6}$$

- IQR
  The definition of the interquartile range is:

$$IQR(u) = Q_3 - Q_1 \tag{4.7}$$

  with Q_1 as the 25th percentile or first quartile and Q_3 as the 75th percentile or third quartile. The 25th percentile is the value below which 25 percent of the samples in the window can be found.

4.5.3 Main frequency component and its amplitude

The frequency-domain features are obtained by using the Short Time Fourier Transform (STFT). The assumption for this technique is that a non stationary signal can be considered as stationary for short time periods. After the elimination of the mean value of the samples in the window, the short time series is multiplied by a window with the same size as the time series, more precisely a Hann window which is recommended, because it is satisfactory in 95 % of cases and has good frequency resolution and reduced spectral leakage [40]. Zero padding is used to obtain a high sampling rate in the spectrum. Then the FFT algorithm is applied to get the Fourier coefficients $X(k)$:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{\frac{-2\pi i k n}{N}}, 0 \leq k \leq N-1, \tag{4.8}$$

where $x(n)$ are the discrete data points multiplied with the Hann window and N is the window size respectively the time series length. Because of the symmetry of the coefficients only the first half has to be considered. The absolute values are built and scaled receiving the values Y. The amplitude spectrum in dB is calculated via

$$Y \text{ [dB]} = 20 \log_{10}(Y + 10^{-6}), \tag{4.9}$$

where the smallest possible value will be -120 dB. The continuous frequency is generated with the formula

$$f_k = fs\frac{k}{L}, \tag{4.10}$$

where fs is the sampling frequency as listed in Tab. 4.1, k is the index of the Fourier coefficient and L is the total length of the sampled signal (including the zeros from padding). The frequency with the highest amplitude is the MFC. Both values, amplitude and frequency, are used as features.

4.5.4 Arc Tangent of z/xy

In order to distinguish between walking on stairs and walking on level ground, a new idea was produced of building the ratio of the vertical and horizontal component of acceleration in the LLF and calculate a virtual slope angle by applying the arc tangent.

4.5.5 Nomenclature

The just described features were provided with own names to facilitate handling, because they are calculated for different basis components, frames and sensors. The nomenclature is primarily dependent on the sensor, i.e., the name starts with the letter

- a, for acceleration (a stands for accelerometer),
- g, for angular velocity (g stands for gyroscope),
- m, for magnetic flux density (m stands for magnetometer).

Secondary, the name depends on the frame and basis component. For example the mean value of the vertical component of acceleration in the LLF is “a_l9”. The “l” stands for LLF. A complete listing is given in Tab. 4.2.

Table 4.2: Feature nomenclature, dependent on the frame.

Feature	**Norm**	**BF**		**LLF**	
	$\|\| \cdot \|\|$	xy	z	xy	z
AU	_0	-	-	-	-
MAX	_n1	_b1	_b8	_l1	_l8
MEAN	_n2	_b2	_b9	_l2	_l9
STD	_n3	_b3	_b10	_l3	_l10
RMS	_n4	_b4	_b11	_l4	_l11
IQR	_n5	_b5	_b12	_l5	_l12
MFC	_n6	_b6	_b13	_l6	_l13
ampl(MFC)	_n7	_b7	_b14	_l7	_l14
atan(z/xy)	-	-		_l15	

The AU is enqueued in the first column where the norm-features are, although it belongs to an own category independent of the frame and calculated with the components of each axis.

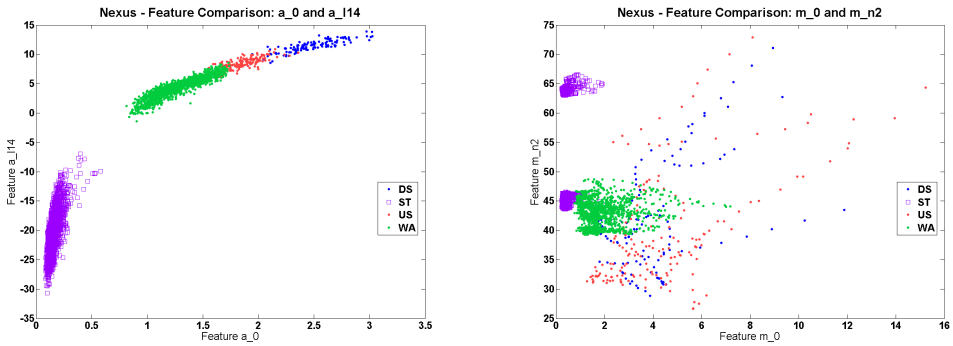
The features are generally numbered consecutively. For the LLF and BF the numbering beginning at the horizontal components continues on the vertical ones. The arc tangent-feature is only calculated in the LLF.

Furthermore, features in the LLF were computed only for accelerometer data and those in the BF only for accelerometer and gyroscope data. Norm-features were calculated for all three measurements. An overview is given in the following Tab. 4.3:

Table 4.3: Feature nomenclature, different sensors. The * stands for all numbers that appear in Tab. 4.2.

	Norm	**BF**	**LLF**
Accelerometer	a_n*	a_b*	a_l*
Gyroscope	g_n*	g_b*	-
Magnetometer	m_n*	-	-

For the investigation of the feature distribution and for comparison, 2D-plots were created in order to identify clouds where the feature values of common activities are concentrated. Two examples are given in Fig. 4.5.



(a) (b)

Figure 4.5: 2D comparison of features - examples, (a) clearly visible clouds of acceleration features, (b) no clearly identifiable clouds of magnetometer features.

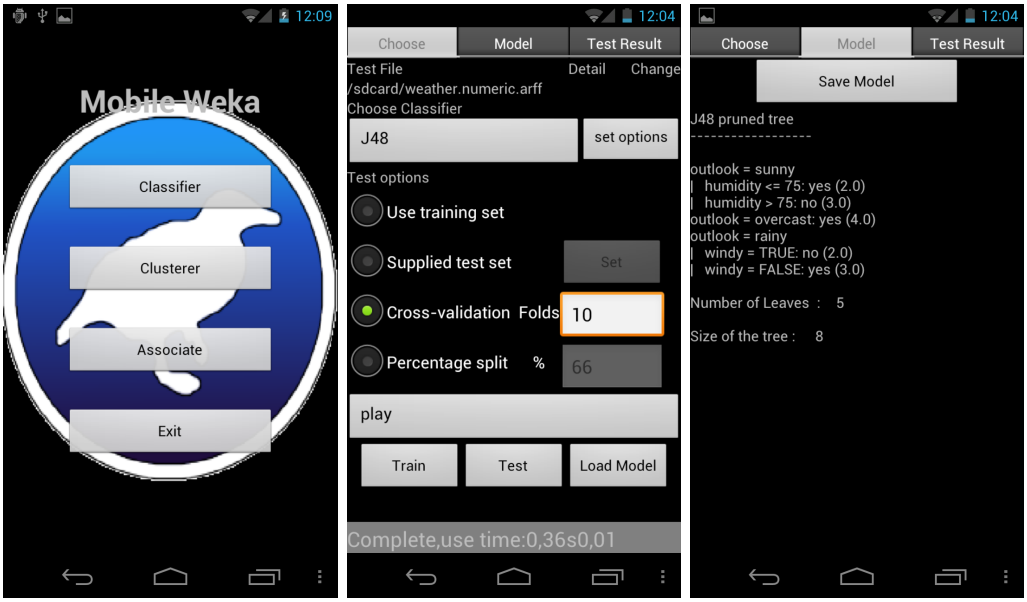
4.6 Feature Selection and Classification Process

The whole classification procedure including the selection of features, building the classifier and evaluating it has been carried out with the use of the widespread machine learning tool WEKA (Waikato Environment for Knowledge Analysis) in the version 3.7.

This software is written in Java and is freely available under the GNU General Public Licence. It is possible to call the WEKA library directly from MATLAB. Thereby, WEKA classes and methods can be used beside MATLAB functions.

Other machine learning tools are for example ELKI, KNIME or RapidMiner, which are also open-source. MATLAB itself includes classification tools as well but less extensive. The C4.5 Decision Tree, for example, is not implemented.

There are several reasons for the use of WEKA. Besides the above mentioned points WEKA includes a variety of classifiers and the possibility to perform Correlation-based Feature Selection (CFS). Moreover, many recent work exist using this software, e.g. Labrador and Yejas [28] used WEKA in combination with an own designed library MECLA in their application Vigilante. Other researchers have also created a mobile version of WEKA (Mobile Weka) to work on Android platforms [32]. Some example screen-shots are given in Fig. 4.6.



(a) Menu (b) Classification settings (c) Generated model

Figure 4.6: Mobile Weka App on Android.

In the present work, the C4.5 Decision Tree, called *J48* in WEKA, has been chosen as the main classifier. As part of the investigations also other classifiers have been studied and mutually evaluated (see chapt. 5.7). All of them have been used with the default settings. A short list of the classification names and settings can be seen in Tab. 4.4.

Table 4.4: Nomenclature and description of used classifiers.

nomenclature	WEKA classifier	description
C4.5	*J48*	C4.5 Decision Tree learner (implements C4.5 revision 8), as default pruned
NB	*NaiveBayes*	standard probabilistic Naive Bayes (NB) classifier
SVM	*SMO*	sequential minimal optimization algorithm for support vector classification
k-NN	*Ibk*	k-nearest-neighbours (k-NN) classifier

WEKA methods have been used to generate and evaluate classifiers using training sets as well as to predict new instances (samples without labels). An instance is a realization of an object with the features and the class as attributes. In the course of the investigations, several combinations of feature-groups (as stated in Tab. 4.3) were tried out and the CFS of WEKA was applied.
The classifiers were evaluated using 10-fold cross validation, showing the recall values and confusion matrices. For the comparison of different classifiers a 5×2-fold cross validation with a paired t-test has been applied. More information on the evaluation of classifiers is given in Section 3.3.2.

A short overview on the most important WEKA classes which were used in this work are stated in Tab. 4.5.

Table 4.5: Most important WEKA classes and methods used in this work.

class	method	description
weka.classifiers	buildClassifier	generates a classifier
	classifyInstance	classifies the given test instance
weka.core.Instances	randomize	shuffles the instances in the set so that they are ordered randomly
	stratify	select and sort the instances according to its class values for a stratified cross-validation afterwards
	trainCV, testCV	creates the training and test set for one fold of a cross-validation on the dataset
	attribute,	returns an attribute resp.
	classAttribute	the class attribute
weka.classifiers.Evaluation	crossValidateModel	performs a cross validation for a classifier on a set of instances
	evaluateModel	evaluates the classifier on a given set of instances

5 Investigations

This chapter comprises the particular investigations that are made in this work. First, the suitability of low-cost sensors is investigated. Based on this, various window lengths and feature subsets are compared, user-specific and speed-dependent activities are examined carefully. Furthermore, the real-time capability is tested and different classifiers are compared with each other. The latter investigation has the aim to determine the actual sensor location on the body.

The abbreviations for the activities used in this chapter are listed in Section 4.1. In the occurring figures and tables, Smartphone and Smartwatch are referred to as Nexus and Omate.

5.1 Ability of Smartphone for AR

Before using the low-cost sensors, like a Smartphone and Smartwatch, by themselves, AR should be investigated on the basis of a medium-priced IMU, the Xsens, in order to generally determine the suitability.

Xsens and Smartphone have been mounted together on the belt as described in Section 4.3. Fig. 5.1 shows the time series of the acceleration of each axis for the Xsens and the Smartphone (Nexus), given in the SF. It can be seen that a comparison of the two devices in this context is not possible. The sensors have been attached to the body in different orientations and that is why the values for each axis, for example, the activity *sitting* (see Fig. 5.1, black rectangles) do not correlate. Furthermore, a replication of the measurements would not deliver the same results as the orientation would definitely change when the device would be mounted on the belt again.

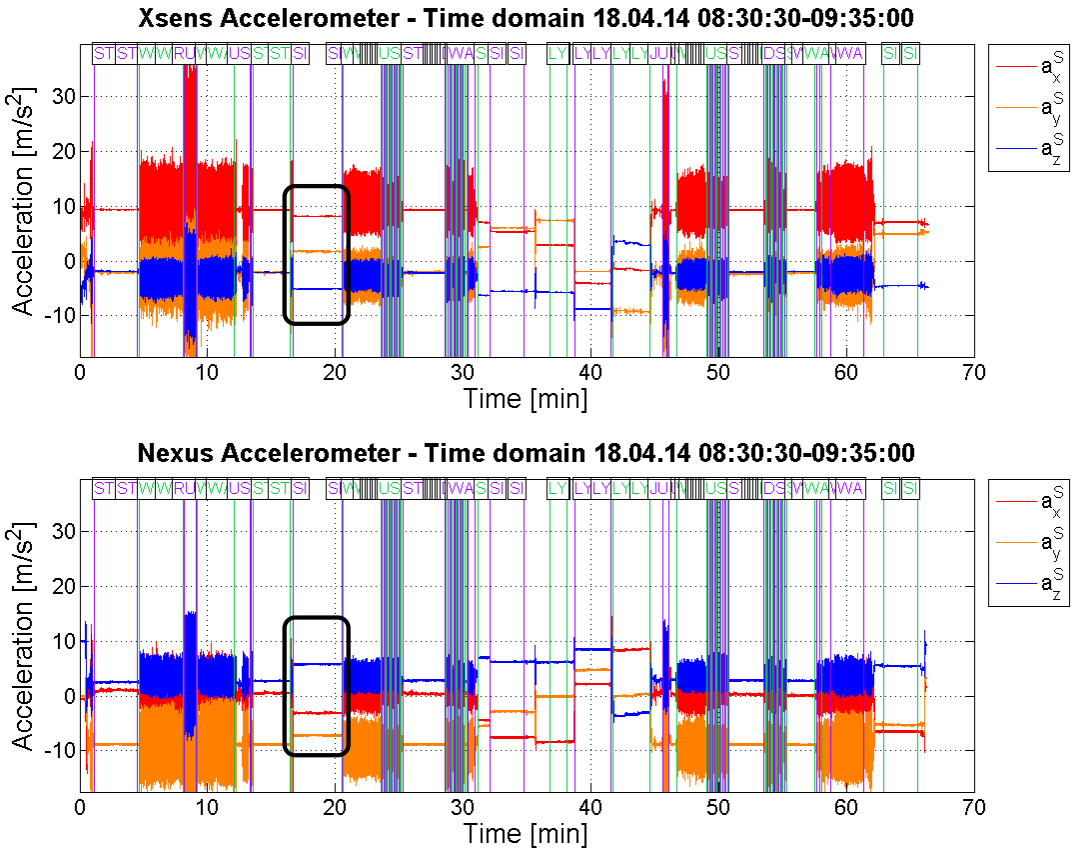


Figure 5.1: Comparison of Xsens and Nexus, acceleration of all three axes in SF.

For this reason, the measurements are transformed into the LLF as described in Chapter 3.2. The horizontal and vertical component (a_{xy}^{LL} and a_{z}^{LL}) are of special interest and can be compared, see Fig. 5.2. It is noticeable that the amplitudes generated by the Xsens appear larger in some cases and differences occur especially in static activities. The reasons can be found in the sensitivity and (in)stability caused by the bias of the Smartphone sensors. The Kalman filter works as best as the signal allows, because the time series of the acceleration norm (Fig. 5.3) shows the same behaviour as the vertical component in the LLF (Fig. 5.2), which definitely includes the largest accelerations, see the cyan rectangles in the figures.

Especially for "low"-motion activities like *standing*, *sitting* and *lying*, where the linear acceleration and angular rate are close to zero, the information depends only on the acceleration due to gravity [15]. If the orientation of the body relative to the LLF is known, the gravity is reflected in particular axes. Therefore the signals are also transformed into the BF.

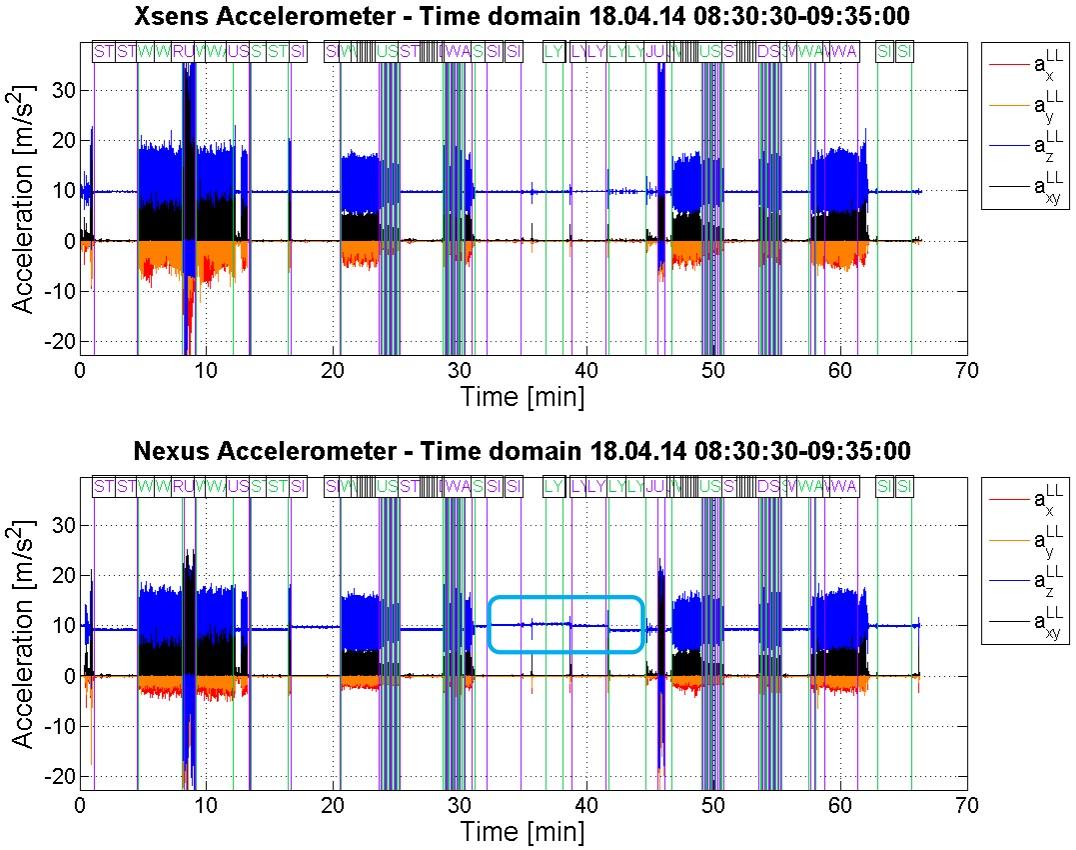


Figure 5.2: Comparison of Xsens and Nexus, acceleration of all three axes in LLF.

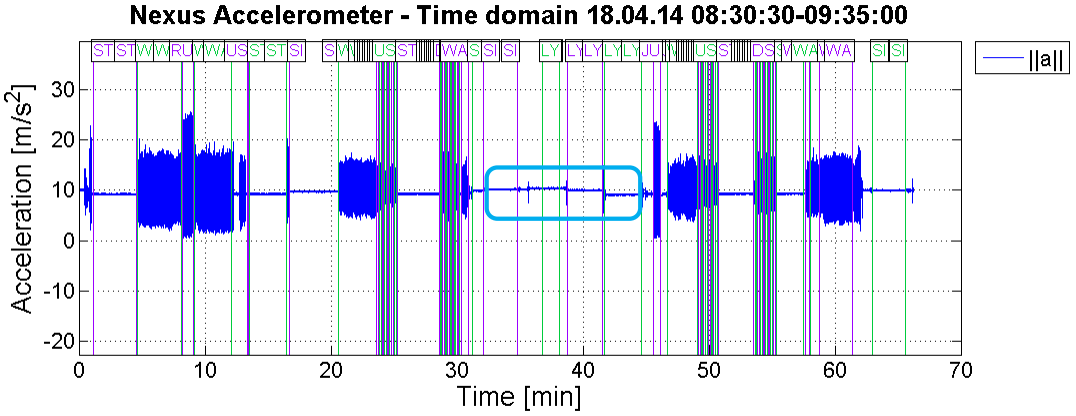


Figure 5.3: Nexus, total acceleration (norm).

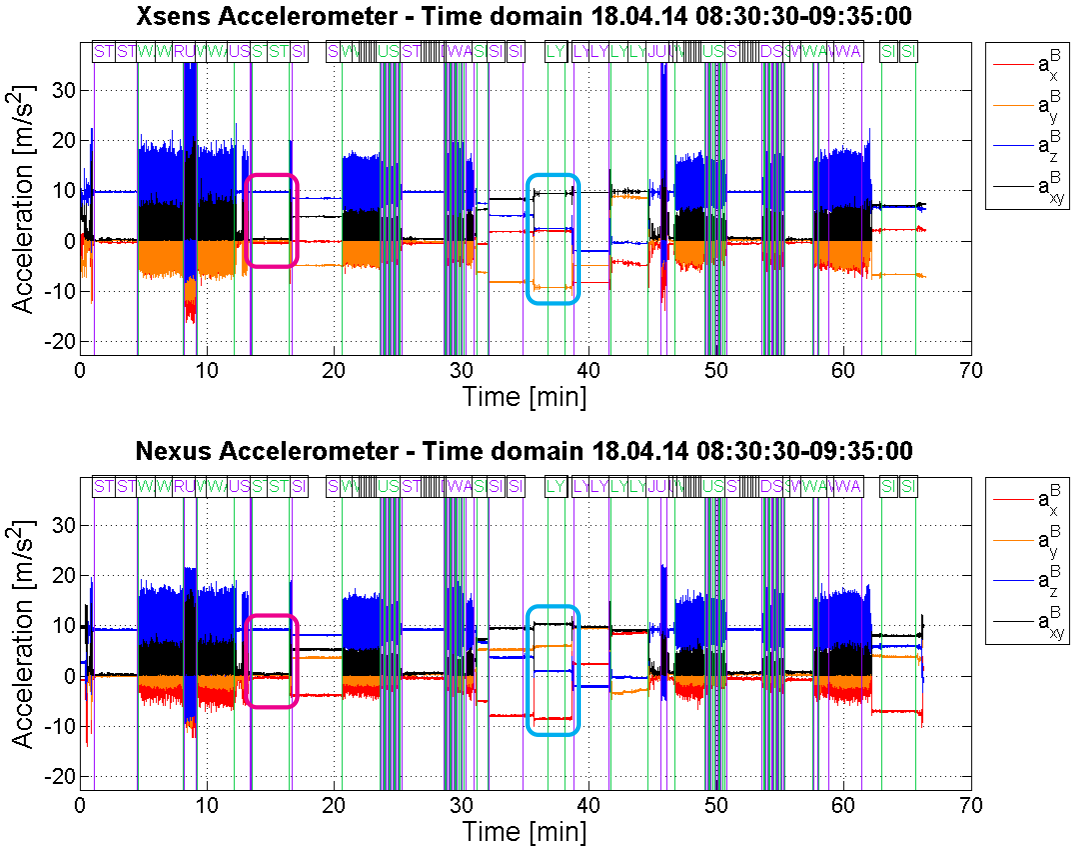


Figure 5.4: Comparison of Xsens and Nexus, acceleration of all three axes in BF.

In Fig. 5.4, where the time series of acceleration in the BF are given, rectangles are drawn, to which special attention is paid. In *standing* position (magenta rectangles) the gravity totally acts on the vertical axis of the BF (a_z^B in blue), in contrast, when the subject is *lying* (cyan rectangles), the gravity is mirrored in the horizontal plane (a_{xy}^B in black). Furthermore, it can be seen that the x and y axes do not match between the devices because of the missing yaw angle needed for the rotation matrix, however, the horizontal component curves show the same run. That is the reason why only vertical and horizontal components are used as a basis for feature computation.

After the calculation of the features with 128 respectively 28 samples per window, their temporal variation can be illustrated. In Fig. 5.5, for example, the time series of the RMS of the acceleration norm (a_n4) is displayed. Although Xsens and Smartphone distinguish recognizable, it can be shown that both devices are able to carry out AR.

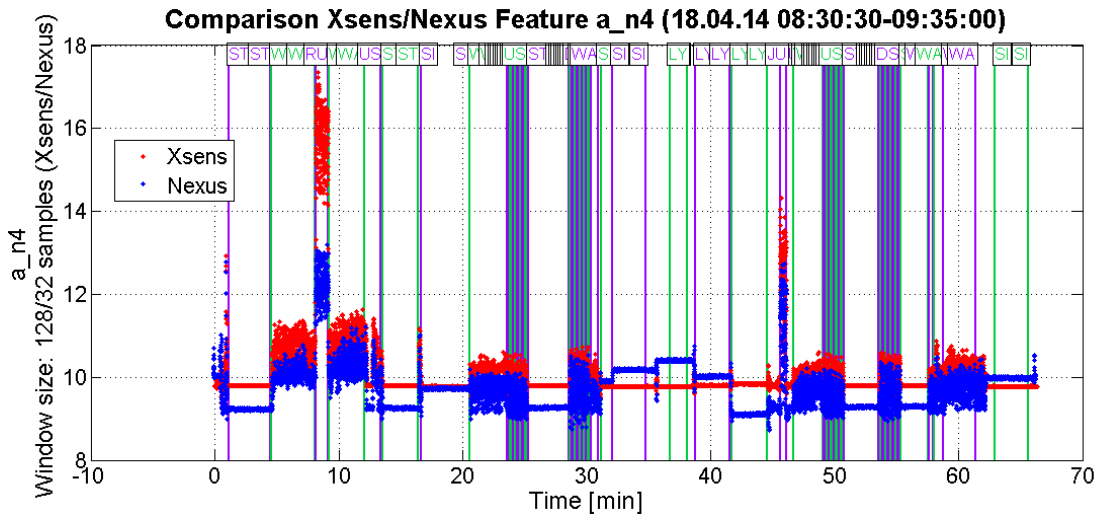


Figure 5.5: Comparison of Xsens and Nexus, time series of the feature a_n4 (RMS of the acceleration norm).

A separate issue is the magnetometer. The measured data can not be compared in the form in which they are presented, see Fig. 5.6, but if a common factor is applied, the curves appear similar to each other, as Fig. 5.7 shows. Apart from the fact that the units do not match, the exemplary feature 2D-plot (Fig. 5.8) shows no separable clouds, which would identify various activities. Additionally, the measured field strength is strongly influenced by electronic devices or other magnetic objects in the vicinity. For these reasons, the magnetometer data is not used in the following for AR.

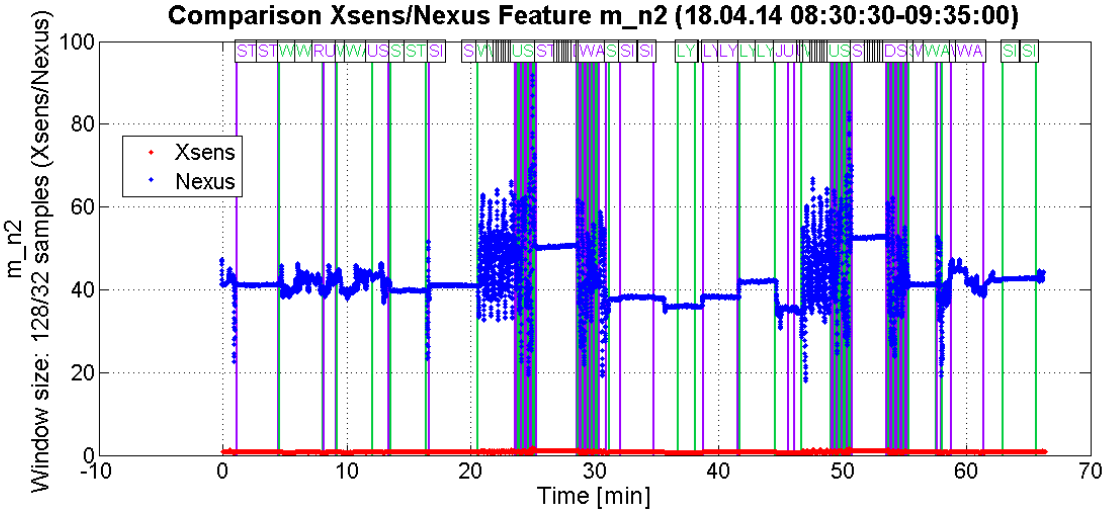


Figure 5.6: Comparison of Xsens and Nexus, time series of the feature m_n2 (Mean of the norm of magnetometer data).

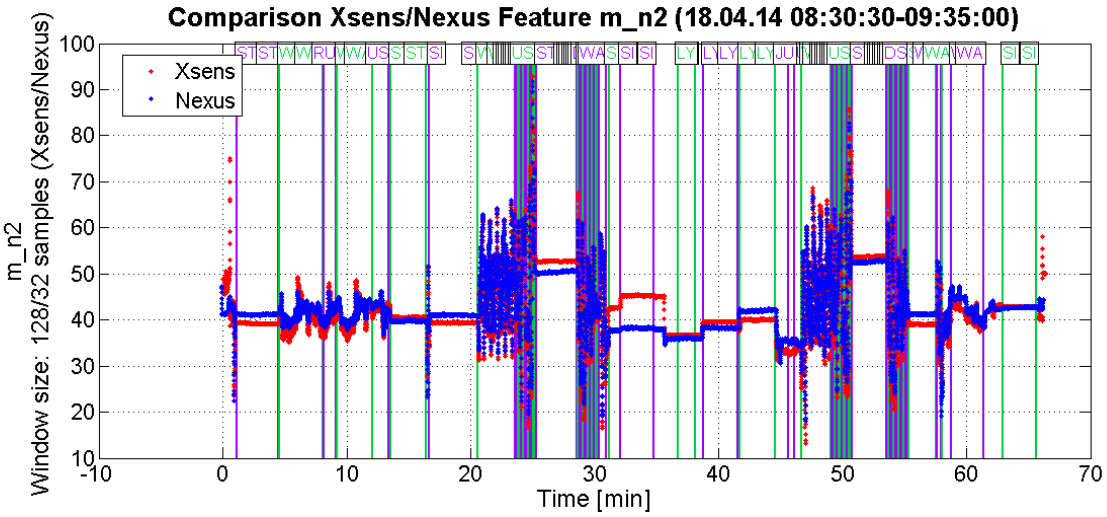
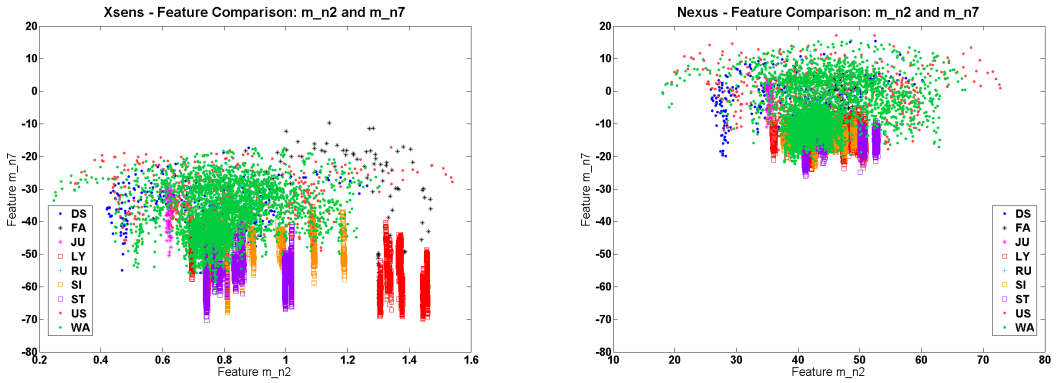


Figure 5.7: Comparison of Xsens and Nexus, time series of the feature m_n2 (Mean of the norm of magnetometer data) - with applied scale factor.



(a) Xsens

(b) Nexus

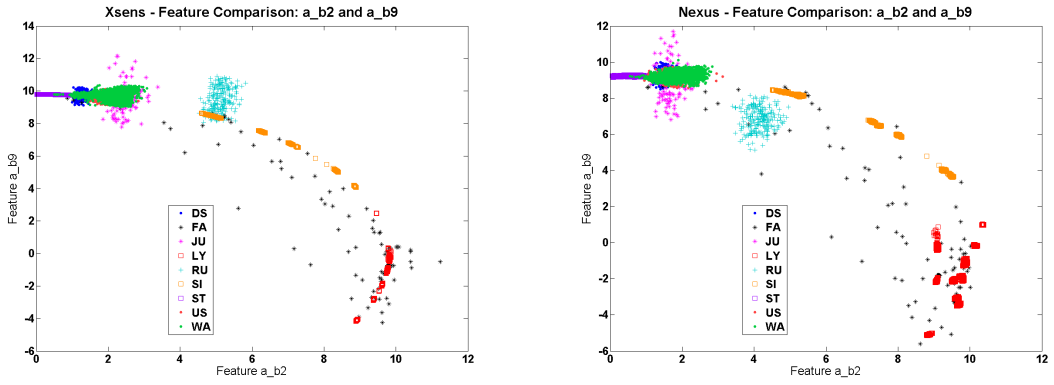
Figure 5.8: Comparison of Xsens and Nexus, 2D-plots of magnetometer-based features. Note the different x-axis scaling!

With the calculated features, a classifier, more particularly a C4.5 Decision Tree, is generated, which is evaluated with a 10-fold cross validation. Different feature groups are used to make any differences identifiable. The recall values for each Xsens and Smartphone (Nexus) are shown in Table 5.1.

Table 5.1: Recall values of 10-fold cross validation of C4.5 Decision Tree classifier, different feature subsets comparing Xsens and Nexus.

Feature subsets	norm+LLF+BF a_n*, g_n*, a_l*, a_b*, g_b*		norm+LLF a_n*, g_n*, a_l*		norm a_n*, g_n*	
Activity \ Device	Xsens	Nexus	Xsens	Nexus	Xsens	Nexus
DS	95.2 %	96.0 %	91.4 %	97.5 %	92.7 %	89.3 %
FA	87.4 %	80.2 %	75.7 %	77.7 %	68.5 %	75.2 %
JU	97.4 %	94.1 %	100.0 %	96.6 %	98.3 %	99.2 %
LY	100.0 %	99.8 %	96.6 %	90.8 %	95.9 %	90.9 %
RU	100.0 %	100.0 %	100.0 %	100.0 %	100.0 %	100.0 %
SI	100.0 %	100.0 %	98.2 %	87.6 %	97.7 %	88.3 %
ST	100.0 %	100.0 %	98.8 %	98.7 %	99.1 %	98.8 %
US	96.7 %	94.5 %	94.5 %	93.0 %	87.5 %	84.0 %
WA	99.5 %	99.1 %	99.1 %	99.1 %	98.2 %	97.8 %
Weighted Avg.	99.6 %	99.4 %	97.8 %	94.2 %	97.2 %	93.7 %

Adding LLF and BF features, the accuracy increases. In all three cases, the overall accuracy of the Smartphone is below the Xsens, but always above 93 %, which is more than satisfactory. The activity *falling* has by far the worst recall values. Furthermore, it is striking that *downstairs* and *upstairs* are more difficult to detect than other activities, both in case of the Smartphone as well as the Xsens. These circumstances may also be observed in the feature 2D-plots in Fig. 5.9.



(a) Xsens (b) Nexus

Figure 5.9: Comparison of Xsens and Nexus, exemplary 2D-plot of BF features.

This study shows that sensor data in LLF and BF contribute significantly to the improvement in accuracy. Because of their impracticality, magnetometer data are not used in the following. The activities *jumping* and *falling* were considered too specific respectively

complex for first experiments with AR within this thesis. Therefore they are no longer dealt with in the next investigations. The conclusion is that the Smartphone Samsung Galaxy Nexus is suitable for AR.

5.2 Variation of the window size

In the second study, the effect of the window length on the recognition accuracy is examined. Both the outcome of the cross validation and the hit rate of predicted data are considered. Three different training datasets are investigated:

- Smartphone in hand in front of the body,
- Smartphone in trouser pocket and
- Smartwatch on the wrist.

In all cases, norm and LLF features in combination with a C4.5 Decision Tree are used.

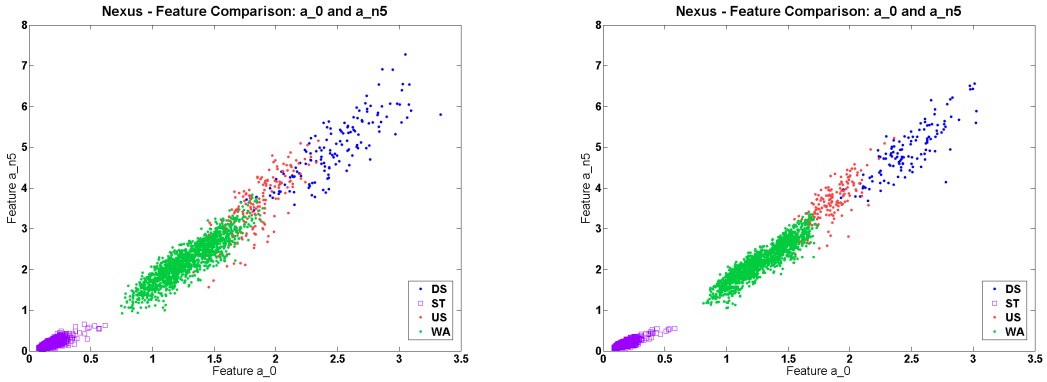
5.2.1 Smartphone in the hand in front of the body

The change in the window size has an effect on the distribution of the features as can be seen in Fig. 5.10. The higher the window size is the closer they are to each other. An exception in this case is the size 128 corresponding to a period of approx. 4.5 seconds, see Fig. 5.10. Looking at the recall results of a 10-fold cross validation (Tab. 5.2), it is quite evident that the accuracy increases with the size of the window and reaches almost 100 % with 64 and 128 samples. This is clear because the windows overlap more with increasing size and the features calculated from these take similar values to each other.

Table 5.2: Recall values of 10-fold cross validation of C4.5 Decision Tree classifier, **Smartphone in the hand in front of body**, different window sizes, norm and LLF features (acceleration and angular rate).

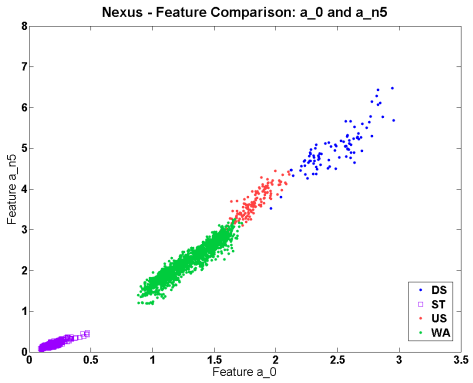
# samples/window	DS	ST	US	WA	Weighted Avg.
16	89.6 %	100.0 %	83.9 %	98.0 %	98.2 %
32	96.7 %	100.0 %	97.1 %	99.7 %	99.7 %
64	100.0 %	100.0 %	100.0 %	99.7 %	99.9 %
128	100.0 %	100.0 %	100.0 %	99.9 %	99.9 %

If the classifiers, which are generated using different window sizes, are applied to test data, the results look different to the cross validation results, see Fig. 5.11.

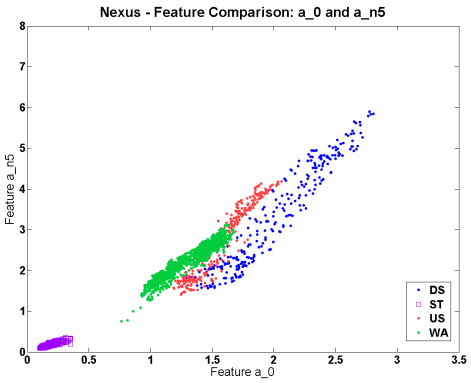


(a) window size=16

(b) window size=32



(c) window size=64

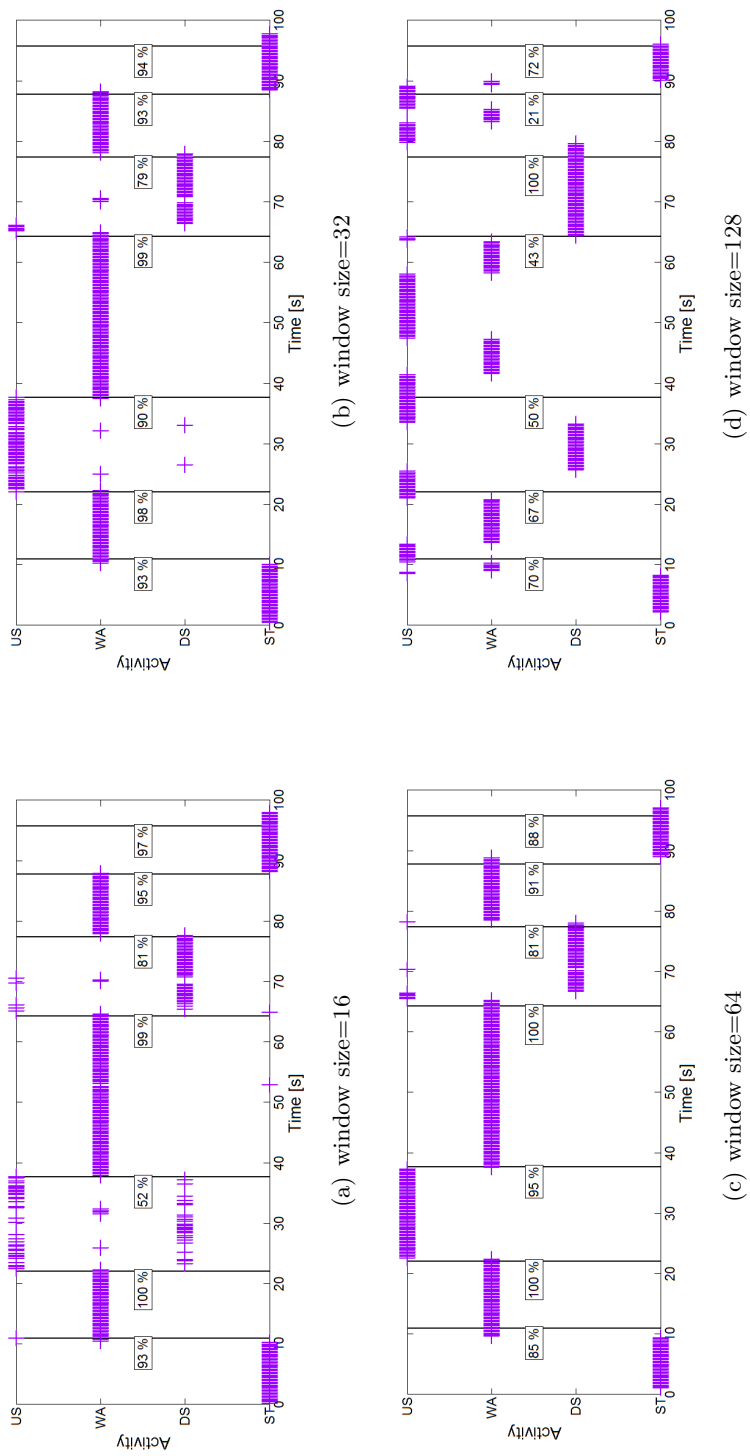


(d) window size=128

Figure 5.10: Comparison of different window sizes, Smartphone in the hand in front of the body, 2D-plot of the features a_0 and a_n5.

Thereby, a sequence of activities was performed, whose transition times were recorded which are marked by black lines.

When using a window size of 128 samples, the activities *walking* and *upstairs* are confused with each other and also *upstairs* with *downstairs*. In various areas, even less than 50 % are predicted correctly. 50 % is the probability, as the chance of playing at dice, which should be achieved at least. In comparison, a very small window size of only 16 samples provides better results. However, the activity *upstairs* is correctly recognized only to the half and the other half as *downstairs*. In the other cases similar good results are obtained. Incidentally, in the test series a staircase was passed, which has a small platform in the middle. This means that for about a second the activity of *walking* was performed which is quite visible in the figures, for example in Fig. 5.11(a) and 5.11(b).



(a) window size=16 (b) window size=32

(c) window size=64 (d) window size=128

Figure 5.11: Prediction of test series **ST-WA-US-WA-DS-WA-ST** collected with **Smartphone in the hand in front of body** using a classification model trained with features which are generated with different window sizes.

5.2.2 Smartphone in trouser pocket

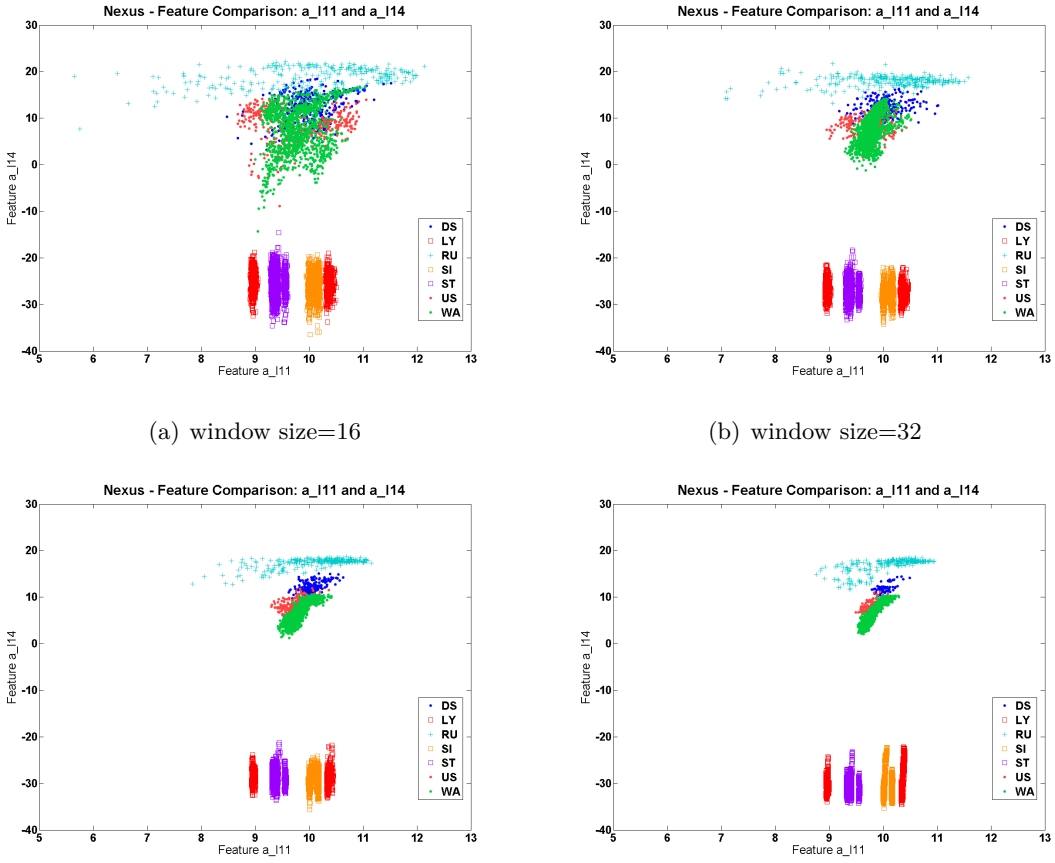
With the Smartphone in the trouser pocket, similar results are observed. The feature clouds are getting more compact and the accuracy observed by cross validation increases with growing window size, see Fig. 5.12 and Tab. 5.3. In this case, also *running* and static activities such as *sitting* and *lying* were recorded. Fig. 5.12 reveals that two point clouds are available for the activity *lying* in the course of the training data recording, because the subject was lying on the back and on the side.

If the DTs which are created on the basis of different window sizes are applied to test data, similar conclusions can be drawn as with the Smartphone in the hand in front of the body. Looking at Fig. 5.13 on page 51, the smallest window size causes inaccuracies in some cases over 50 %. With the highest window size (128 samples) *walking* can practically not be identified and *downstairs* only by about 37 %. Comparing the two middle window sizes, it is noticeable that there are a few inaccuracies at *standing*, whereas *walking* is quite well recognized. In deciding on the best size, the focus is directed to *downstairs* and *upstairs*. If one wants to get at least 50 % correctly predicted data, the decision is made in favour of the window size of 32 samples which corresponds to approximately 1.14 seconds.

A second test series was performed including static activities and the predicted results are displayed in Fig. 5.14 on page 52. The transitions between activities are greyed out and are not taken into account for prediction accuracy, as shown in the figures. Because such transitions are not present in the training data, they are of course difficult to be recognized. It is interesting that they are detected as *upstairs* or *walking*. Apart from that, the static activities can be very well predicted in all cases. In the border areas, there appear inaccuracies that become bigger with the increase of the window size which is logical as more activities are smeared together.

Table 5.3: Recall values of 10-fold cross validation of C4.5 Decision Tree classifier, **Smartphone in trouser pocket**, different window sizes, norm and LLF features (acceleration and angular rate).

# samples/ window	DS	LY	RU	SI	ST	US	WA	Weigh. Avg.
16	85.8 %	100.0 %	100.0 %	100.0 %	100.0 %	83.4 %	94.6 %	97.2 %
32	96.7 %	99.9 %	100.0 %	100.0 %	100.0 %	94.0 %	97.6 %	99.0 %
64	99.1 %	100.0 %	100.0 %	100.0 %	100.0 %	97.8 %	99.6 %	99.8 %
128	98.7 %	100.0 %	100.0 %	100.0 %	100.0 %	100.0 %	99.8 %	99.9 %



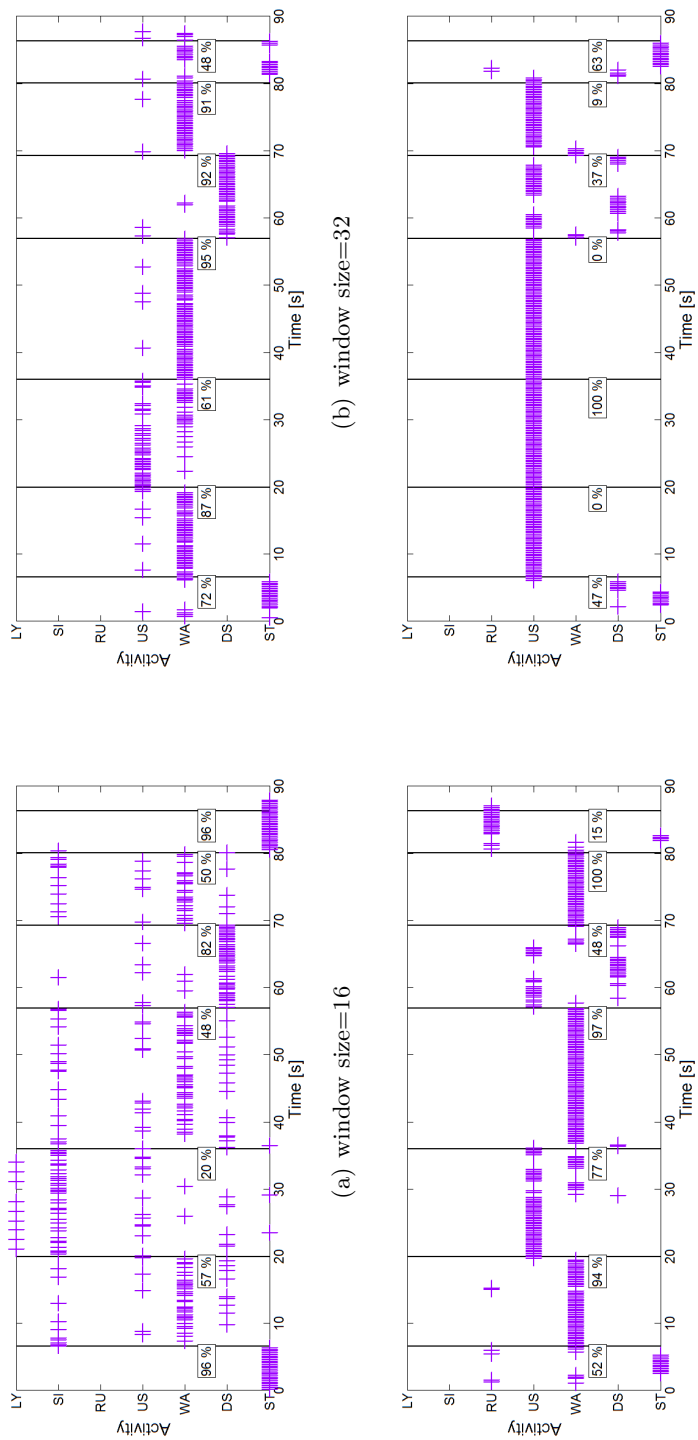
(a) window size=16

(b) window size=32

(c) window size=64

(d) window size=128

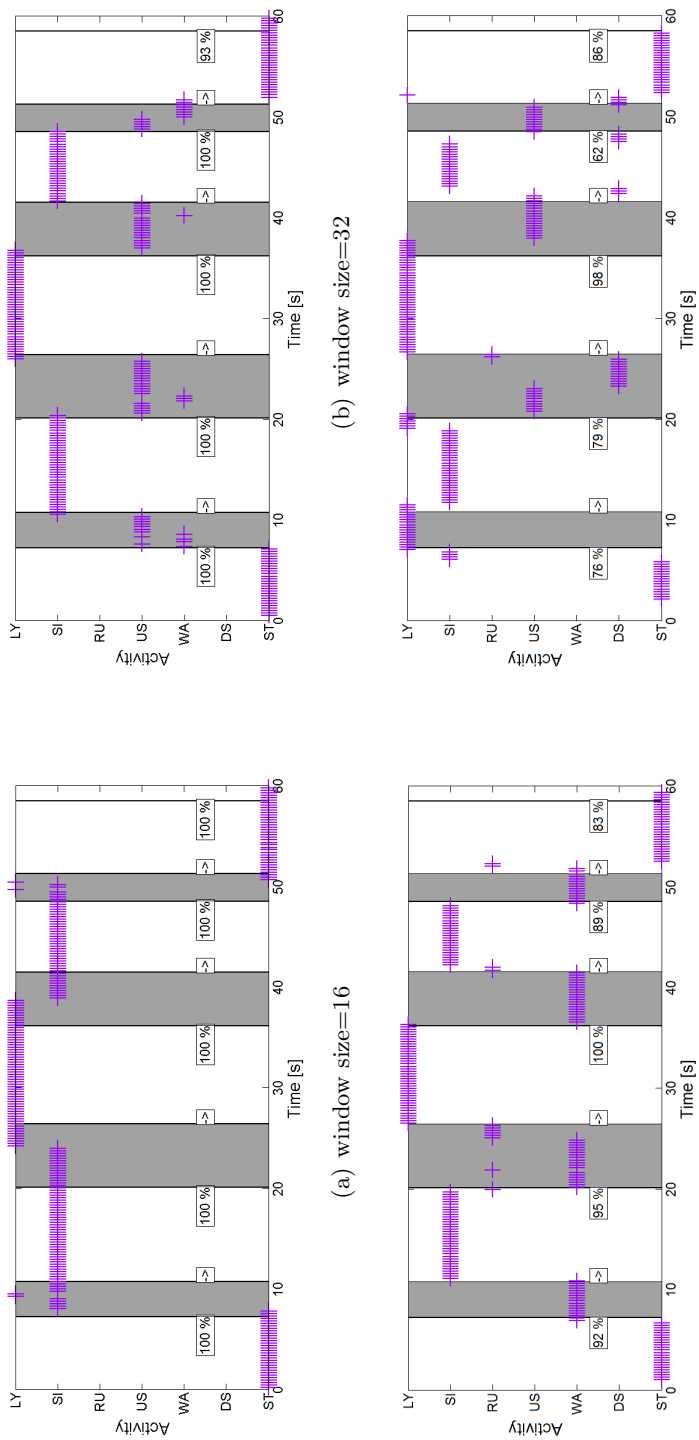
Figure 5.12: Comparison of different window sizes, Smartphone in trouser pocket, 2D-plot of the features a_l11 and a_l14.



(a) window size=16 (b) window size=32

(c) window size=64 (d) window size=128

Figure 5.13: Prediction of test series **ST-WA-US-WA-DS-WA-ST** collected with **Smartphone in trouser pocket** using a classification model trained with features which are generated with different window sizes.

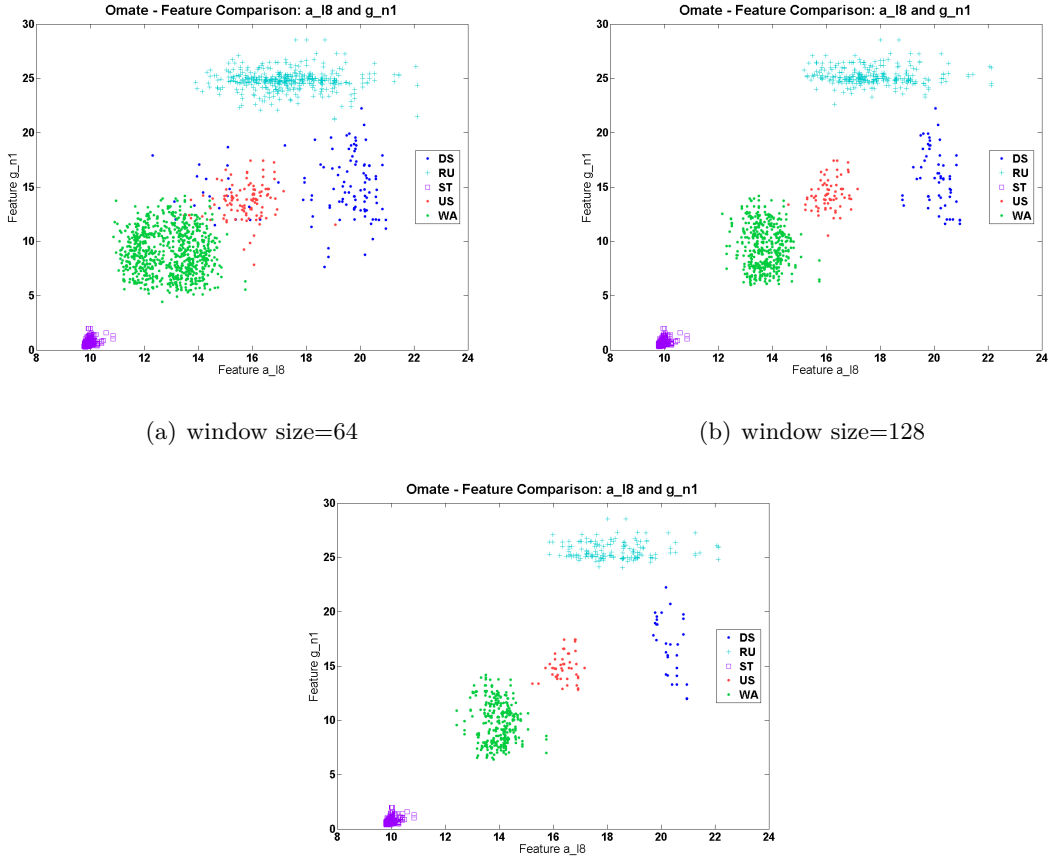


(a) window size=16 (b) window size=32

(c) window size=64 (d) window size=128

Figure 5.14: Prediction of test series **ST-SI-LY-SI-ST** collected with **Smartphone in trouser pocket** using a classification model trained with features which are generated with different window sizes (the grey blocks represent the transitions between consecutive activities).

5.2.3 Smartwatch on left wrist



(a) window size=64

(b) window size=128

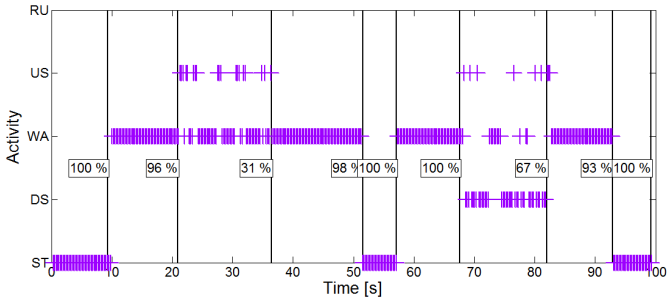
(c) window size=256

Figure 5.15: Comparison of different window sizes, Smartphone in trouser pocket, 2D-plot of the features a_l8 and g_n1.

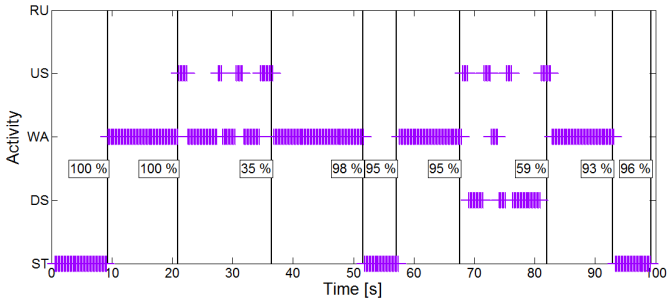
With the Smartwatch on the wrist it does not behave differently than with the Smartphone. Here, however, only three different window sizes are tested. Exemplary feature 2D-plots are shown in Fig. 5.15 and the recall results are stated in Tab. 5.4. The prediction of a test series is shown in Fig. 5.16. There is not much difference between 64 and 128 samples. Using 256 samples results in major confusions. *Downstairs* is almost not detected. Since for the Smartphone 32 samples were selected, i.e., approximately 1.14 seconds, the decision is made for a similar size, namely 1.28 seconds. These are 128 samples per window with a sampling rate of 100 Hz.

Table 5.4: Recall values of 10-fold cross validation and C4.5 Decision Tree, **Smartwatch on wrist**, different window sizes, norm and LLF features (acceleration and angular rate).

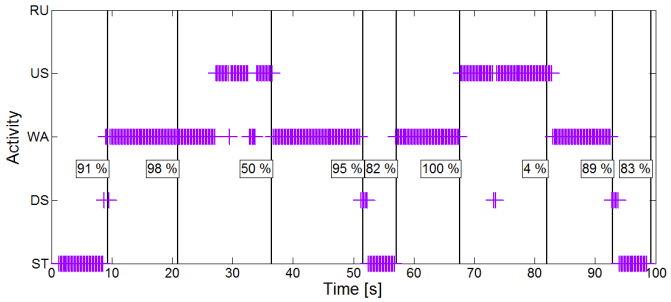
# samples/window	DS	RU	ST	US	WA	Weighted Avg.
64	97.8 %	100.0 %	100.0 %	90.2 %	99.3 %	99.3 %
128	99.1 %	100.0 %	100.0 %	97.5 %	99.7 %	99.8 %
256	100.0 %	100.0 %	100.0 %	100.0 %	99.9 %	100.0 %



(a) window size=64



(b) window size=128

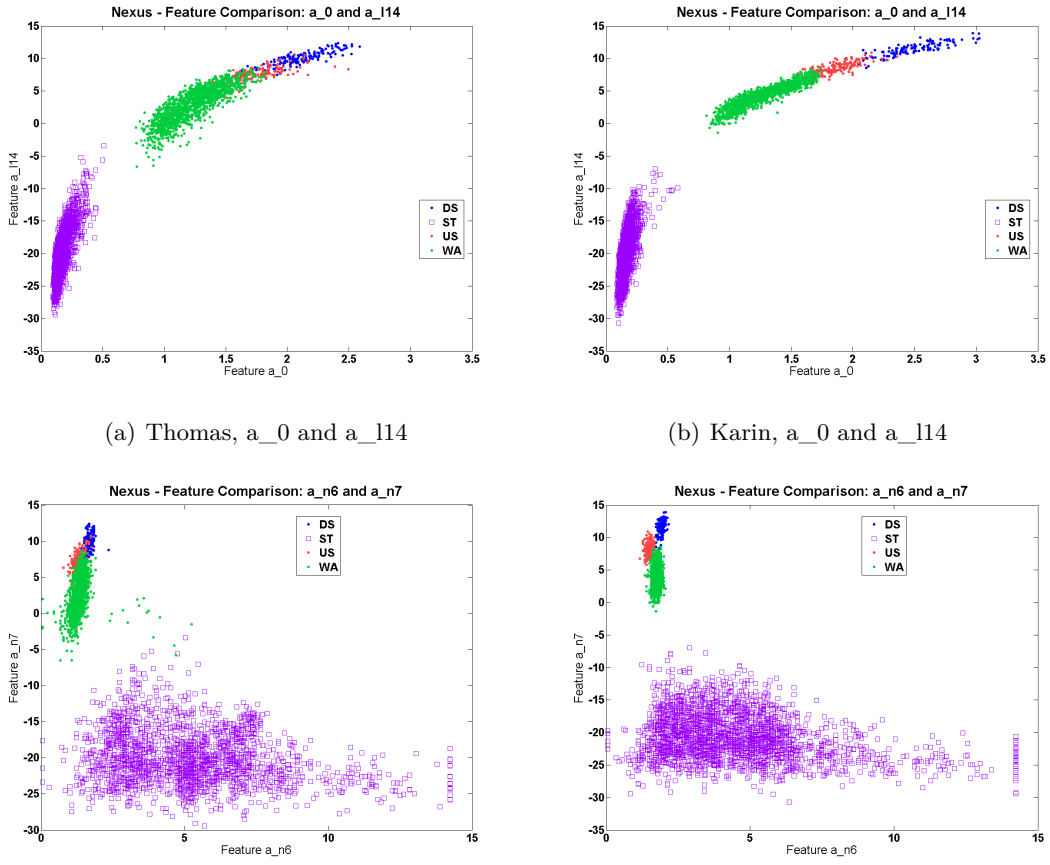


(c) window size=256

Figure 5.16: Prediction of test series **ST-WA-US-WA-ST-WA-DS-WA-ST** collected with **Smartwatch on the wrist** using a classifier trained with features generated with different window sizes.

5.3 User-specific activities

This chapter examines, how the data and the classification results change, when different people perform the same activities while holding the Smartphone in the hand in front of the body. Right from the data collection, there was the impression that Thomas moves very smoothly, both on stairs and on level ground. In contrast, Karin moves herself upstairs more impulsively and drops herself downwards. Some feature 2D-plots are given in Fig. 5.17. Visually, the feature-clouds of activities are more clearly separable with Karin's data.



(a) Thomas, a_0 and a_l14

(b) Karin, a_0 and a_l14

(c) Thomas, a_n6 and a_n7

(d) Karin, a_n6 and a_n7

Figure 5.17: Comparison of two different people, 2D-plots of different features.

Tab. 5.5(a) shows the recall values of a 10-fold cross validation using the C4.5 Decision Tree. Norm and LLF features from acceleration and angular rate data and a window size of 32 samples are used. Three classifiers are generated - with the data from Thomas, from Karin and from both of them. The second column in the table contains the number of decision rules, i.e., the number of nodes in the constructed trees. In the case of Thomas,

26 rules are created, where an overall accuracy of 97.7 % can be achieved. That is 2 % less than with Karin, for whom only 10 rules are needed. Combining both data sets yields an overall accuracy located in the middle of both. To reach an accuracy of 98.3 %, much more nodes are created than with the separate data. The recall value for *downstairs* is even smaller than with Thomas' data, namely "only" 90 %.

The corresponding confusion matrices of the cross validation results are shown in Tab. 5.5(b) to 5.5(d). The amount of instances is approximately the same for both subjects. For Thomas, mainly *upstairs* and *walking* are confused with each other. In 10 cases *upstairs* is considered as *downstairs*. This uncertainty is also reflected in the results of the mixed data (Tab. 5.5(d)).

Table 5.5: Evaluation results of 10-fold cross validation of C4.5 Decision Tree classifier, Smartphone in hand in front of body, different users, norm and LLF features (acceleration and angular rate), window size=32.

(a) Recall values

Person	#rules	DS	ST	US	WA	Weighted Avg.
Thomas	26	93.5 %	99.8 %	77.1 %	98.0 %	97.7 %
Karin	10	96.7 %	100.0 %	97.1 %	99.7 %	99.7 %
Thomas & Karin	45	90.0 %	100.0 %	82.9 %	98.7 %	98.3 %

(b) Confusion Matrix, Thomas

a	b	c	d	← classified as
158	0	8	3	a = DS
0	1890	0	3	b = ST
10	0	138	31	c = US
2	0	26	1364	d = WA

(c) Confusion Matrix, Karin

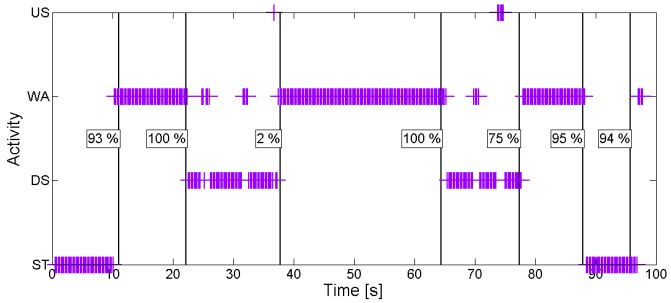
a	b	c	d	← classified as
117	0	2	2	a = DS
0	2331	0	0	b = ST
1	0	166	4	c = US
2	0	2	1416	d = WA

(d) Confusion Matrix, Thomas & Karin

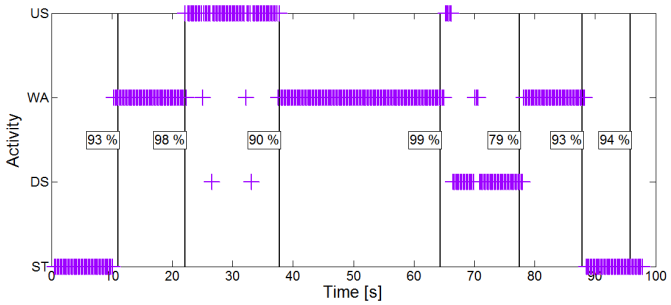
a	b	c	d	← classified as
261	0	27	2	a = DS
0	4222	0	2	b = ST
21	0	290	39	c = US
1	0	36	2775	d = WA

Lockhart and Weiss [33] define three types of models in their work: impersonal, personal and hybrid models. If test data, from which activities need to be predicted, are taken

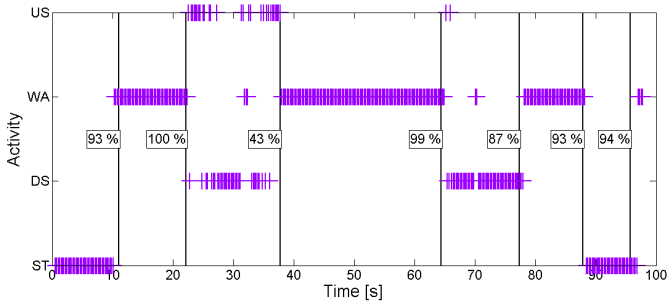
from Karin, the model with her training data is a personal model. If, however, the classifier is taken, which was created with Thomas' data, it is an impersonal model. A DT with mixed data is called hybrid model.



(a) Training data: Thomas - impersonal model



(b) Training data: Karin - personal model



(c) Training data: Thomas & Karin - hybrid model

Figure 5.18: Prediction of test series **ST-WA-US-WA-DS-WA-ST** collected with **Smartphone in the hand in front of the body** from Karin using classification models trained with data from different persons (Thomas; Karin; Thomas & Karin).

In order to assess the impact of such models, a test series was carried out with Karin. The results are depicted in Fig. 5.18. There are activities, such as *standing* and *walking*, where the model makes almost no difference. While walking on stairs, especially up, clear differences can be seen.

As the authors in [33] conclude, a personalized model performs best whereas an impersonal model works much worse, because it cannot effectively distinguish between certain activities which are user-specific. It can be deduced that both *downstairs* and *upstairs* must be user-specific activities.

5.4 Different feature subsets

In the previous studies, norm and LLF features were used as input for the classifier. In this chapter the effects of different combinations of feature-groups, as stated in Tab. 4.3 on page 36, will be studied. The training data to be examined are:

- Smartphone in the hand in front of the body (Karin, Thomas),
- Smartphone in trouser pocket and
- Smartwatch on left wrist.

5.4.1 Smartphone in the hand in front of the body

The training data collected by Thomas respectively Karin were evaluated using eight different combinations of feature-groups, reaching from the acceleration norm features up to all computed features of acceleration and angular rate. If there are more than 30 features, also a Correlation-based Feature Selection (CFS) is applied to the groups in order to reduce the amount of features by not significantly reducing the accuracy. In some cases, the overall accuracy even increases.

In Tab. 5.6, the recall values for different feature-group combinations are given. The amount of features and the number of rules in the DT are stated additionally. The worst result, considering primary the overall accuracy and secondary the amount of features, is marked in red. The best result is coloured in green. With Thomas' data the accuracies range from 97.4 to 98.1 %. The trees have about 25 to 30 nodes. In contrast, the results when using Karin's data differ barely. Accuracies from 99.5 to 99.7 % are reached. The amount of rules varies from 10 to 12.

The confusion matrices of the worst and best results are given in Tab. 5.7.

Table 5.6: Recall values of 10-fold cross validation and C4.5 Decision Tree classifier, Smartphone in the hand in front of the body, window size=32, several users, several selected features (CFS).

location	user	feature subset	#features	#rules	DS	ST	US	WA	Weighted Avg.
hand in front of body	Thomas	a_n*	8	25	91.1 %	99.9 %	74.9 %	97.7 %	97.4 %
		a_n*, g_n*	16	36	91.7 %	99.9 %	76.0 %	98.0 %	97.6 %
		a_n*, g_n*, a_l*	31	26	93.5 %	99.8 %	77.1 %	98.0 %	97.7 %
		CFS(a_n*, g_n*, a_l*)	16	31	95.9 %	99.8 %	79.9 %	98.3 %	98.1 %
		a_n*, g_n*, a_l*, a_b*, g_b*	59	28	95.3 %	99.8 %	82.1 %	97.1 %	97.7 %
		CFS(a_n*, g_n*, a_l*, a_b*, g_b*)	24	29	95.9 %	99.8 %	81.6 %	97.9 %	98.0 %
		a_n*, g_n*, a_b*, g_b*	44	30	94.1 %	99.9 %	82.7 %	97.8 %	98.0 %
		CFS(a_n*, g_n*, a_b*, g_b*)	17	27	95.3 %	99.9 %	81.6 %	97.9 %	98.0 %
hand in front of body	Karin	a_n*	8	11	97.5 %	100.0 %	94.7 %	99.5 %	99.5 %
		a_n*, g_n*	16	11	96.7 %	100.0 %	97.1 %	99.7 %	99.7 %
		a_n*, g_n*, a_l*	31	10	96.7 %	100.0 %	97.1 %	99.7 %	99.7 %
		CFS(a_n*, g_n*, a_l*)	17	11	96.7 %	100.0 %	93.6 %	99.7 %	99.5 %
		a_n*, g_n*, a_l*, a_b*, g_b*	59	10	96.7 %	100.0 %	96.5 %	99.6 %	99.6 %
		CFS(a_n*, g_n*, a_l*, a_b*, g_b*)	26	11	96.7 %	100.0 %	96.5 %	99.7 %	99.7 %
		a_n*, g_n*, a_b*, g_b*	44	10	96.7 %	100.0 %	96.5 %	99.8 %	99.7 %
		CFS(a_n*, g_n*, a_b*, g_b*)	23	12	96.7 %	100.0 %	97.1 %	99.6 %	99.7 %

Table 5.7: Confusion matrix of worst (left, red) and best (right, green) classification evaluation result (see Table 5.6), Smartphone in the hand in front of the body, Thomas and Karin, window size=32.

(a) Thomas: a_n*

a	b	c	d	← classified as
154	0	10	5	a = DS
0	1891	0	2	b = ST
14	0	134	31	c = US
4	0	28	1360	d = WA

(b) Thomas: CFS(a_n*, g_n*, a_l*)

a	b	c	d	← classified as
162	0	4	3	a = DS
0	1890	0	3	b = ST
5	0	143	31	c = US
1	0	23	1368	d = WA

(c) Karin: CFS(a_n*, g_n*, a_l*)

a	b	c	d	← classified as
117	0	2	2	a = DS
0	2331	0	0	b = ST
1	0	160	10	c = US
2	0	2	1416	d = WA

(d) Karin: a_n*, g_n*

a	b	c	d	← classified as
117	0	4	0	a = DS
0	2331	0	0	b = ST
1	0	166	4	c = US
2	0	2	1416	d = WA

It can be seen that the activities *walking*, *upstairs* and *downstairs* are confused at most. The worst result of Karin is better than the best of Thomas. This is an indication that there are persons whose activities can be classified better or worse, all this practically independent of the choice of features.

5.4.2 Smartphone in trouser pocket

Tab. 5.8(a) shows the classification results for data collected with the Smartphone located in the trouser pocket using different feature subsets. In almost all cases the activities *running*, *sitting* and *standing* are reproduced to 100 %. *Lying* with 99.9 to 100 % is just as well. The worst result (red) in matters of the weighted average (last column) is received by using the remaining features after CFS applied to norm features of acceleration and angular rate data. The recall value for the activity *upstairs* is about 68 % only.

Looking at the corresponding confusion matrix (Tab. 5.8(b)), it can be observed that nearly a third of the true *upstairs* instances are classified as *walking*. In contrast, the best result (green, Tab. 5.8(a) and Tab. 5.8(c)) with 96.4 % recall value for *downstairs* and 97.7 % for *upstairs* is achieved by using all kinds of features, including norm, LLF and BF features.

Table 5.8: Recall values and confusion matrices of 10-fold cross validation of C4.5 Decision Tree classifier, Smartphone in trouser pocket, window size=32, several selected features (CFS).

(a) recall values

location	user	feature subset	#features	#rules	DS	LY	RU	SI	ST	US	WA	Weig. Avg.
trouser	Karin	a_n*	8	65	89.6 %	99.9 %	100.0 %	100.0 %	100.0 %	74.9 %	95.7 %	97.3 %
pocket		a_n*, g_n*	16	47	96.1 %	99.9 %	100.0 %	100.0 %	100.0 %	83.5 %	94.6 %	97.9 %
		CFS(a_n*, g_n*)	6	50	94.1 %	99.9 %	100.0 %	100.0 %	100.0 %	67.8 %	95.0 %	97.0 %
		a_n*, g_n*, a_l*	31	30	96.7 %	99.9 %	100.0 %	100.0 %	100.0 %	94.0 %	97.6 %	99.0 %
		CFS(a_n*, g_n*, a_l*)	7	56	96.1 %	99.9 %	99.6 %	100.0 %	100.0 %	80.3 %	96.3 %	98.0 %
		a_n*, g_n*, a_l*, a_b*, g_b*	59	23	96.4 %	100.0 %	100.0 %	100.0 %	100.0 %	97.7 %	99.2 %	99.6 %
		CFS(a_n*, g_n*, a_l*, a_b*, g_b*)	13	30	95.1 %	100.0 %	100.0 %	100.0 %	100.0 %	95.2 %	98.8 %	99.3 %
		a_n*, g_n*, a_b*, g_b*	44	22	95.1 %	100.0 %	100.0 %	100.0 %	100.0 %	98.0 %	99.2 %	99.5 %
		CFS(a_n*, g_n*, a_b*, g_b*)	10	30	95.8 %	100.0 %	100.0 %	100.0 %	100.0 %	95.7 %	98.4 %	99.3 %

(b) confusion matrix: CFS(a_n*, g_n*)

a	b	c	d	e	f	g	← classified as
289	0	1	0	0	7	10	a = DS
0	999	0	1	0	0	0	b = LY
0	0	278	0	0	0	0	c = RU
0	0	0	1041	0	0	0	d = SI
0	0	0	0	2175	0	0	e = ST
6	0	0	0	0	238	107	f = US
15	0	0	0	0	42	1094	g = WA

(c) confusion matrix: a_n*, g_n*, a_l*, a_b*, g_b*

a	b	c	d	e	f	g	← classified as
296	0	0	0	0	2	9	a = DS
0	1000	0	0	0	0	0	b = LY
0	0	278	0	0	0	0	c = RU
0	0	0	1041	0	0	0	d = SI
0	0	0	0	2175	0	0	e = ST
0	0	0	0	0	343	8	f = US
5	0	0	0	0	4	1142	g = WA

The DT for this configuration primary includes BF features, as can be seen in Fig. 5.19. Thus, using the specially defined coordinate system, climbing stairs seems to be under control. This conclusion is, however, immediately contradicted.

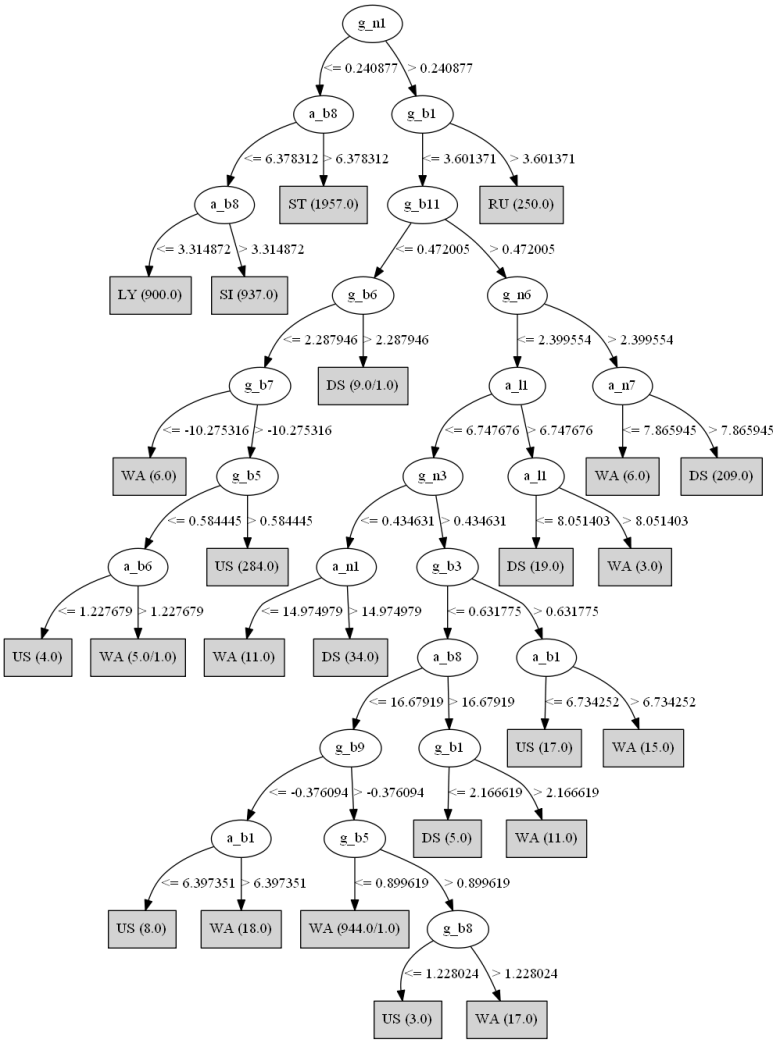


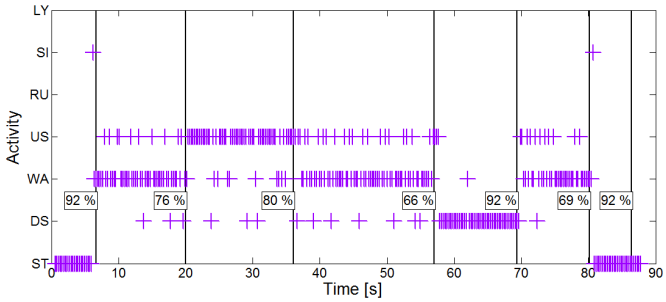
Figure 5.19: C4.5 DT built using all kinds of features (norm, LLF, BF) corresponding to green result in Tab. 5.8, Smartphone in trouser pocket, window size=32.

In Fig. 5.20, the predictions of a test set with the activity-sequence ST-WA-DS-WA-US-WA-ST using the worst and best classifier are shown. Indeed, the activity *upstairs* is reproduced better by the best classifier as well as *walking*, but the activities *downstairs* and *standing* have better recall values using the worst classifier. The DTs do not properly reproduce the activities of the test data. There are many possible reasons for that:

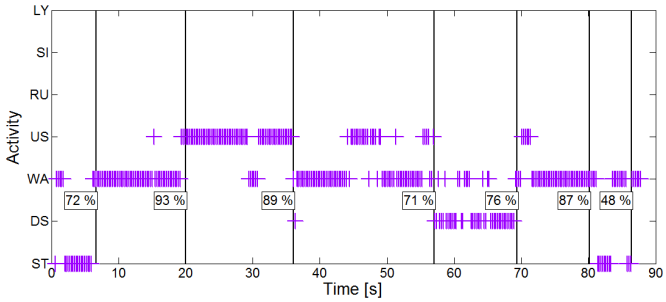
- The number of training data is too small. Variations in the gait depending on mood, physical condition, clothing, or other circumstances are not adequately covered.

- The features are selected inadequately.
- A DT is not suitable for this kind of activities in connection with such features.

The conclusion is that it is necessary to carry out further investigations regarding features and classification trees.



(a) CFS(a__n*, g__n*)



(b) a__n*, g__n*, a__l*, a__b*, g__b*

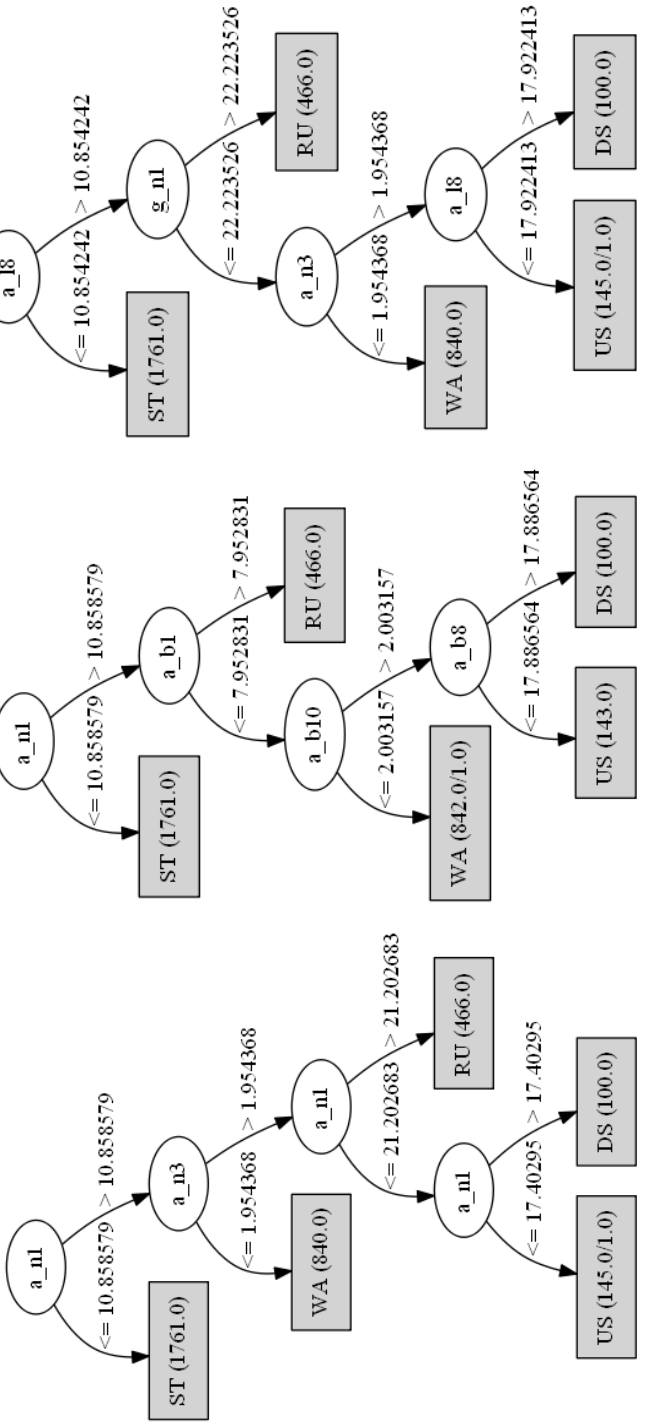
Figure 5.20: Prediction of test series **ST-WA-US-WA-DS-WA-ST** collected with **Smartphone in trouser pocket** using worst and best classifier (Tab. 5.8).

5.4.3 Smartwatch on left wrist

The evaluation results in form of recall values using different groups of features in the case of Smartwatch on the wrist is given in Tab. 5.9.

Table 5.9: Recall values of 10-fold cross validation of C4.5 Decision Tree classifier, Smartwatch on left wrist, window size=128, several selected features (CFS).

location	user	feature subset	#features	#rules	DS	RU	ST	US	WA	Weig. Avg.
wrist	Karin	a_n*	8	4	100.0 %	100.0 %	100.0 %	96.9 %	99.7 %	99.8 %
		a_n*, g_n*	16	4	100.0 %	100.0 %	100.0 %	96.9 %	99.7 %	99.8 %
		CFS(a_n*, g_n*)	9	4	100.0 %	100.0 %	100.0 %	96.9 %	99.7 %	99.8 %
		a_n*, g_n*, a_l*	31	4	99.1 %	100.0 %	100.0 %	97.5 %	99.7 %	99.8 %
		CFS(a_n*, g_n*, a_l*)	13	4	100.0 %	100.0 %	100.0 %	96.9 %	99.7 %	99.8 %
		a_n*, g_n*, a_l*, a_b*, g_b*	59	4	100.0 %	100.0 %	100.0 %	98.1 %	99.8 %	99.9 %
		CFS(a_n*, g_n*, a_l*, a_b*, g_b*)	22	4	100.0 %	100.0 %	100.0 %	97.5 %	99.8 %	99.8 %
		a_n*, g_n*, a_b*, g_b*	44	4	100.0 %	100.0 %	100.0 %	98.1 %	99.8 %	99.9 %
		CFS(a_n*, g_n*, a_b*, g_b*)	18	4	100.0 %	100.0 %	100.0 %	97.5 %	99.8 %	99.8 %



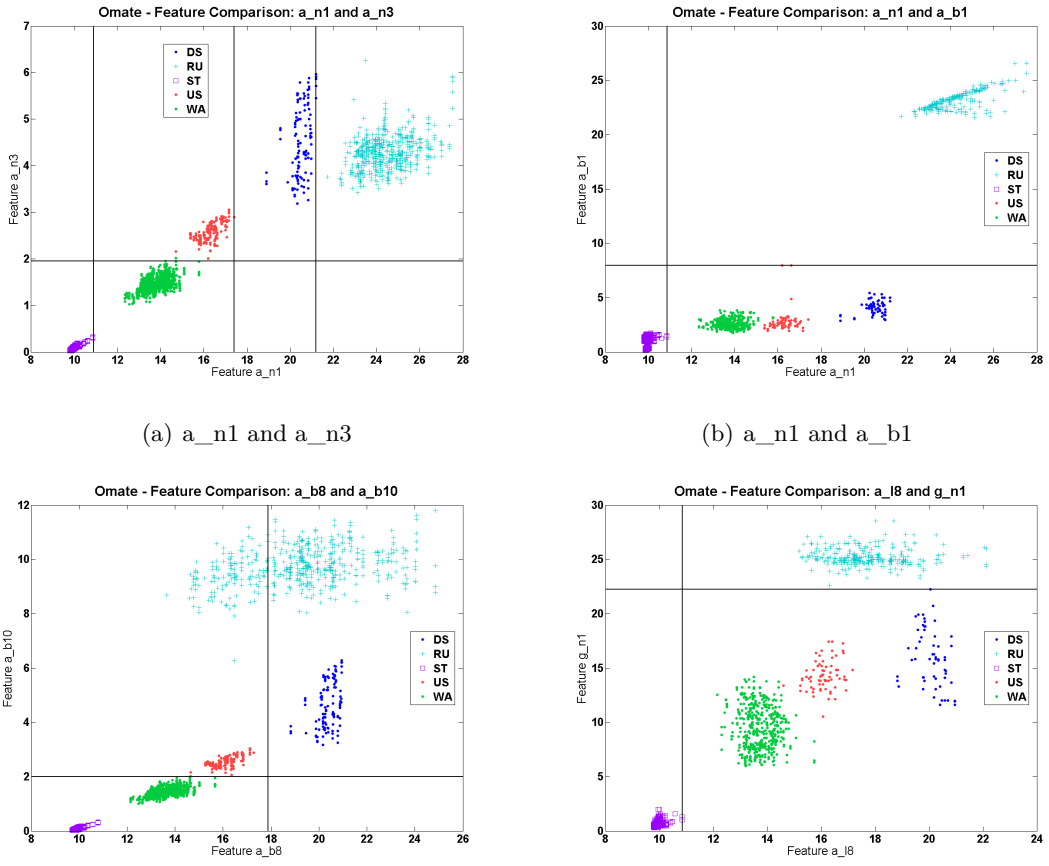
(a) a_n* / a_n*, g_n*

(b) a_n*, g_n*, a_l*, a_b*, g_b* / a_n*, g_n*, a_b*,g_b*

(c) a_n*, g_n*, a_l*

Figure 5.21: C4.5 Decision Trees built using different feature subsets, Smartwatch on left wrist, window size=128.

In contrast to the previous studies, neither a worst nor a best result is emerging in this investigation. They all perform equally well. It is noticeable, that in all cases a tree with only four rules is generated. Overall, in the eight cases only three different trees occur, which are shown in Fig. 5.21 on the previous page.



(a) a_n1 and a_n3

(b) a_n1 and a_b1

(c) a_b8 and a_b10

(d) a_l8 and g_n1

Figure 5.22: Exemplary feature 2D-plots showing the dividing rules used in the C4.5 Decision Trees in Fig. 5.21. Features from Smartwatch training set, window size=128.

The numerical values, that appear as rules in the trees to separate the individual activities, are shown as black lines in the feature 2D-plots of Fig. 5.22. The feature-clouds of the individual activities are well separable. The remaining features for classification are limited to a_n1, a_n3, g_n1, a_l8, a_b1, a_b8 and a_b10. Their denomination is declared in Tab. 4.3 and 4.2 in Chapter 4.5.5.

If these DTs are used for prediction of test data, the results hardly differ. The predicted data using norm and LLF features for the tree is shown in Fig. 5.16(b) on page 54. Tree (a) and (c) of Fig. 5.21 deliver exactly the same result, while in (b) 50 % instead of 59 %

of *downstairs* are recognized correctly.

The conclusion is, that no clear statement can be made concerning the selection of feature subsets. For certain sensor positions different features are better or worse. Overall, all combinations provide good results. Solely, the activities *downstairs* and *upstairs* are difficult to detect, as shown by test data.

5.5 Activities performed with different speed

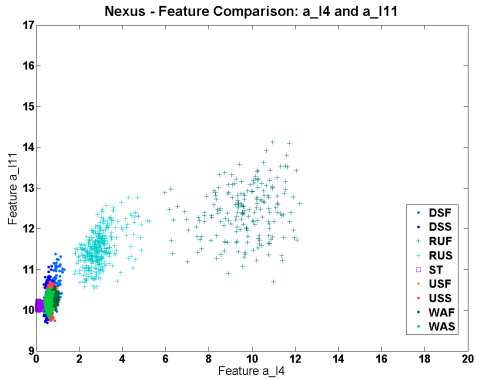
This study shows, on the one hand, that activities at different speeds yield different results, and on the other hand, the Smartphone in the hand in front of the body is compared to the Smartwatch on the wrist.

Again, norm and LLF features are taken as input for the DT classifiers. There will be chosen those window sizes that were selected in the previous studies.

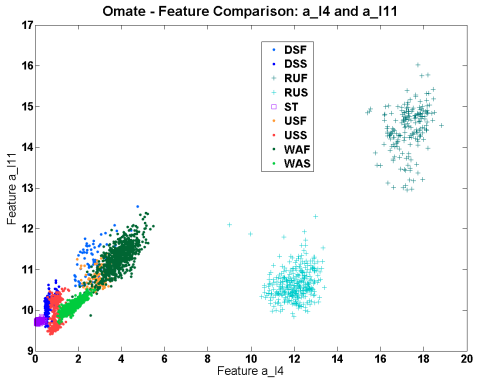
Fig. 5.23 shows four exemplary feature 2D-plots, where features of activities performed in slow and fast manner are represented (the abbreviations are extended by a letter "F" or "S", e.g., walking fast is shortened by WAF). The data were recorded with Smartphone and Smartwatch each and are compared with each other in the figure. On the left side the data of the Smartphone are shown, on the right side the results of the Smartwatch. It is noticeable that both datasets differ significantly from one another.

The Smartphone was held in the right hand in front of the body. During *running* the arm is moved back and forth in a natural way, resulting in the Smartphone constantly changing its location in relation to the body. Accelerations occur in the horizontal plane of the LLF, so the RMS of the acceleration (a_l4) for *running* reaches larger values, see Fig. 5.23(a). The linear acceleration is therefore greater than the angular variation. The Activity Units of acceleration and angular rate are given in Fig. 5.23(c). Visually, slow and fast performed *running* can be separated very well. The feature-clouds of the other activities overlap partially.

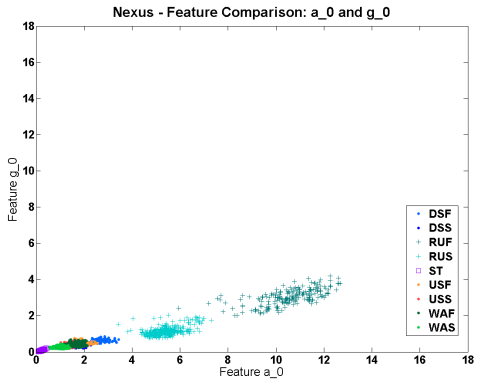
The Smartwatch was worn on the left wrist, the arm naturally swinging when walking, either on stairs or on level ground. While *running*, the arm is bent and moved back and forth as with the Smartphone. Therefore, there occur larger differences in the feature plots, see Fig. 5.23(b) and 5.23(d). Unlike the Smartphone, the clouds of the slow and fast performed activities may be separated visually in Fig. 5.23(d). Only *walking fast* (WAF) and *upstairs fast* (USF) overlap each other for the most part.



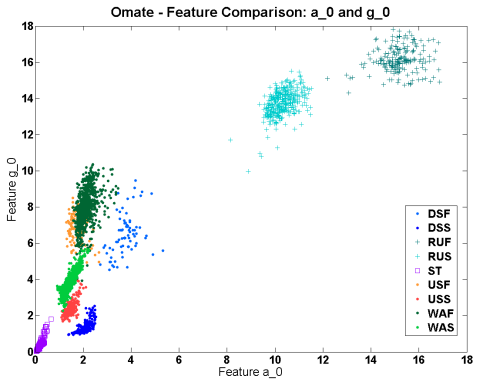
(a) Smartphone: a_l4 and a_l11



(b) Smartwatch: a_l4 and a_l11



(c) Smartphone: a_0 and g_0



(d) Smartwatch: a_0 and g_0

Figure 5.23: Comparison of Smartphone in hand in front of body and Smartwatch on wrist, speed dependent activities, exemplary feature 2D-plots.

Looking at the 10-fold cross validation result of the DT on the basis of the Smartphone (Nexus) data in Tab. 5.10, it can be seen that the speed-dependent activities can be recognized to at least 73.9 %, in most cases over 90 %. Walking fast (WAF) and upstairs fast (USF) are confused most often in the classification process. The same applies to the Smartwatch (Omate) wherein comparatively less instances are mapped incorrectly. The Smartwatch receives better overall accuracy as the Smartphone, namely 99 %, for almost all activities the recall value is higher than with the Smartphone, as can be extracted from 5.10(b).

Table 5.10: Evaluation results 10-fold cross validation of C4.5 Decision Tree classifier, dataset with slow and fast performed activities, Smartphone (hand in front of the body, Nexus) and Smartwatch (Omate), norm+LLF features (acceleration and angular rate), window size=32.

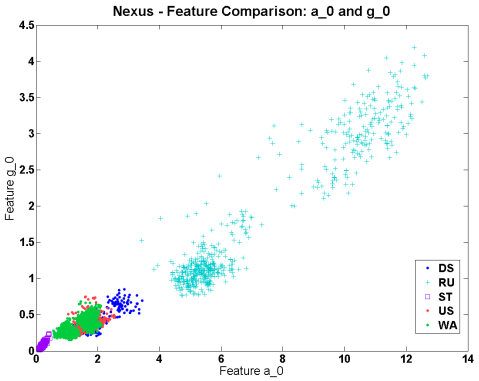
(a) Confusion matrix, speed dependent activities, Nexus

a	b	c	d	e	f	g	h	i	← classified as
82	0	0	0	0	2	0	0	0	a = DSF
1	178	0	0	1	2	6	8	3	b = DSS
0	0	196	0	0	0	0	0	0	c = RUF
2	0	0	357	0	0	0	0	0	d = RUS
0	0	0	0	3144	0	0	0	0	e = ST
0	3	0	0	0	105	0	34	0	f = USF
0	8	0	0	0	0	233	2	5	g = USS
0	3	0	0	0	37	3	629	6	h = WAF
0	2	0	0	0	2	4	4	874	i = WAS

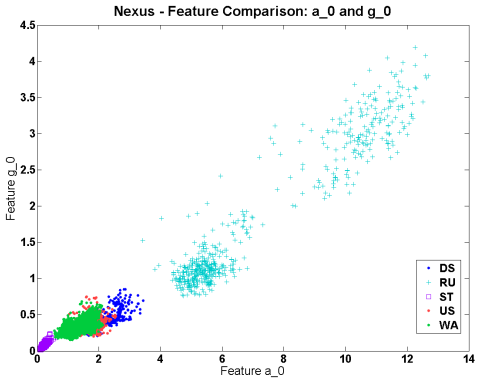
(b) Recall values, speed dependent activities

	Nexus	Omate
# rules	56	30
DSF	97.6 %	97.4 %
DSS	89.4 %	99.5 %
RUF	100.0 %	100.0 %
RUS	99.4 %	99.7 %
ST	100.0 %	100.0 %
USF	73.9 %	87.9 %
USS	94.0 %	94.0 %
WAF	92.8 %	97.9 %
WAS	98.6 %	99.0 %
Weighted Avg.	97.7 %	99.0 %

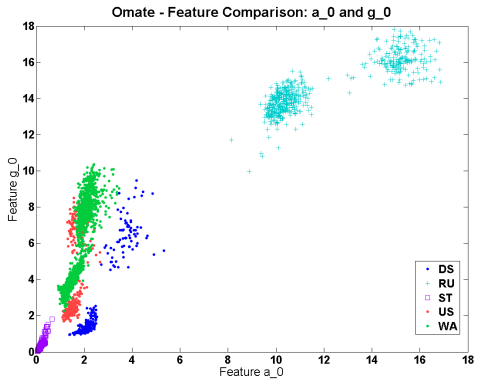
In a further step, the speed dependent activities are merged together, so there remain only speed independent classes, for example, *walking* (WAS + WAF = WA). The feature 2D-plots are displayed in Fig. 5.24(a) and 5.24(c). If the dataset is further extended with other training data, which were recorded with independence of speed, the feature-clouds melt slightly together which can be seen in Fig. 5.24(b) and 5.24(d).



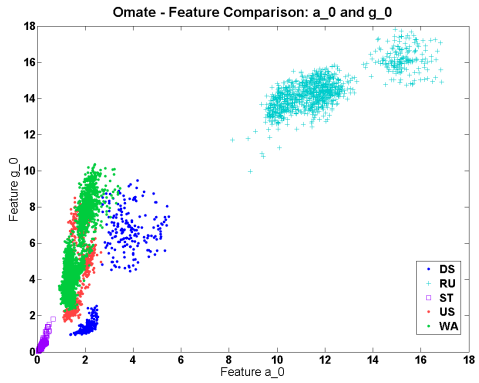
(a) Smartphone, combined data



(b) Smartphone, combined & training data (18.07.)



(c) Smartwatch, combined data



(d) Smartwatch, combined & training data (10.09.)

Figure 5.24: Comparison of Smartphone in the hand in front of the body and Smartwatch on the wrist, exemplary feature 2D-plots, (a) and (c): combined data (speed independent activities, e.g. WAS+WAF=WA), (b) and (d): combined data extended by other training data.

These four datasets are now evaluated using 10-fold cross validation after building a DT classifier each. The recall values are illustrated in Tab. 5.11. It is noticeable that for Omate less rules in the tree are required, while even higher accuracies are achieved. In the extended data set, there are generally more rules produced in the tree, the overall accuracy remains approximately the same. The individual recall values change in both directions, better and worse, mainly with *walking*, *upstairs* and *downstairs*.

Table 5.11: Recall values of 10-fold cross validation of C4.5 Decision Tree classifier, dataset with speed independent activities, Smartphone (hand in front of body, Nexus) and Smartwatch (Omate), norm+LLF features (acceleration and angular rate), window size=32.

(a) combined activites (speed independent)

	Nexus	Omate
# rules	52	25
DS	89.8 %	97.7 %
RU	99.8 %	100.0 %
ST	100.0 %	100.0 %
US	83.6 %	94.0 %
WA	96.1 %	99.0 %
Weighted Avg.	97.4 %	99.2 %

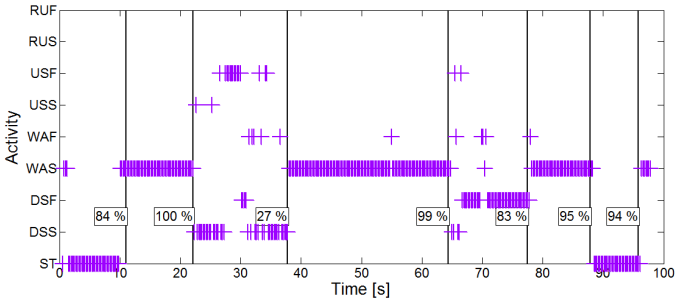
(b) combined activites + other training data

	Nexus	Omate
added data	18.07.14	10.09.14
# rules	79	50
DS	83.7 %	99.2 %
RU	99.6 %	100.0 %
ST	99.9 %	100.0 %
US	79.1 %	91.4 %
WA	98.0 %	98.7 %
Weighted Avg.	97.5 %	99.1 %

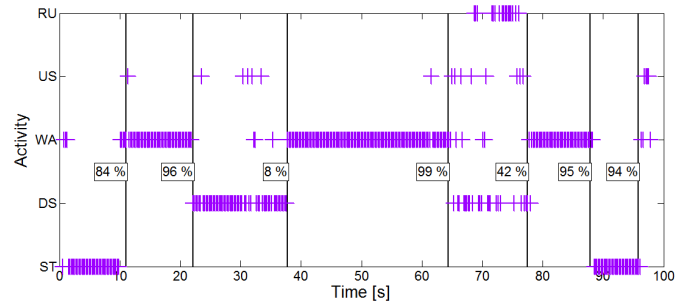
The three models, built from speed dependent, combined and extended data, for Smartphone and Smartwatch each, are applied to a test series. The time series of the predicted activities are illustrated in Fig. 5.25 respectively in Fig. 5.26.

The analysis of the predicted Smartphone data, given in Fig. 5.25, yields that the activity *standing* is correctly classified to 84 to 94 % and *walking* to 93 to 100 %. *Upstairs* reaches only 27 % at speed independent data, in the combined data only 8 %, but in the extended data 43 %. *Downstairs* deteriorates by a half with the combination of slow and fast performed activities into independent activities to 42 %. With the addition of other training data it gets better again, to 89 %.

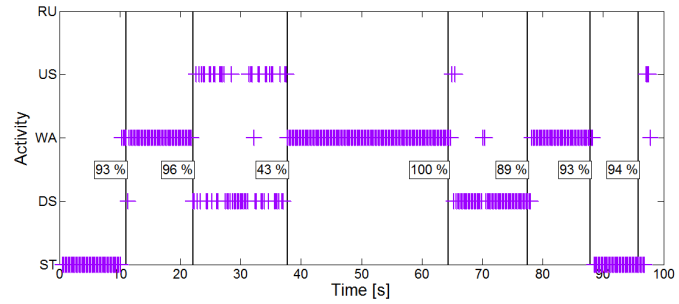
The predicted Smartwatch data in Fig. 5.26 show no differences for the activity *standing*, which is on average correctly recognized to 98 %. *Walking* is best reproduced with the combined data. The best result for *upstairs*, although only 26 %, is achieved with the extended data. 76 % of *downstairs* instances are correctly classified with the separated data according to speed. The combined data cause only 33 % correctly classified instances of *downstairs*, while 57 % accuracy can be achieved with the extended data. For each activity another model fits best.



(a) Nexus, speed dependent

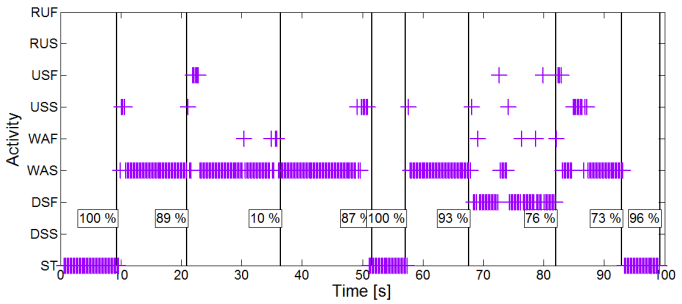


(b) Nexus, combined data

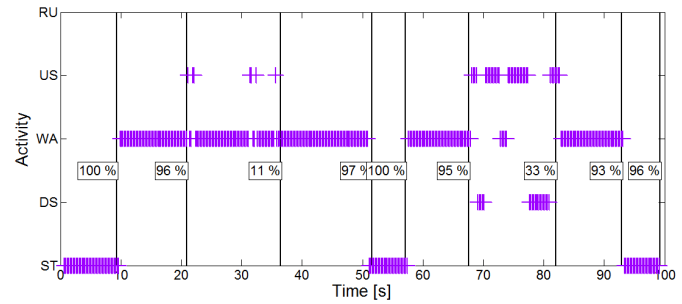


(c) Nexus, extended data

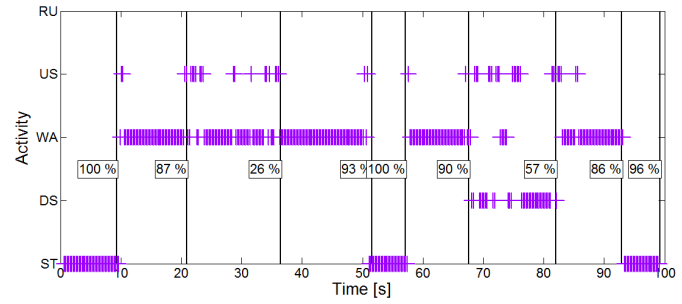
Figure 5.25: Prediction of test series **ST-WA-US-WA-DS-WA-ST** collected with **Smartphone in the right hand in front of the body** using classification models containing speed dependent and independent activities.



(a) Omate, speed dependent



(b) Omate, combined data



(c) Omate, extended data

Figure 5.26: Prediction of test series **ST-WA-US-WA-ST-WA-DS-WA-ST** collected with **Smartwatch on the left wrist** using different classification models containing speed dependent and independent activities.

In this study, no clear pattern for a best model emerges, there is no clear improvement or deterioration in the evaluation results and the predicted data.

The conclusion is that, under cross validation results, all classifiers provide overall accuracies better than 97 %, but with test data emerge different results. They are poor and non-satisfying in case of *downstairs* and *upstairs*. Possible reasons for that have already been given in Chapter 5.4.2. Standing and walking reach a sufficiently high accuracy.

5.6 Real-time capability

This investigation according real-time capability is carried out to show that AR performed on a Smartphone may be possible in real-time.

A test data set was selected and the computation time of the prediction of its instances, together with preprocessing and feature computation, was determined. The AR algorithm was repeated ten times in order to get statistical values. The notebook used for the calculation is a Lenovo Thinkpad Edge E530 with Intel Core i3-2350M CPU at 2.3 GHz, 4 GB RAM, and 64-bit operating system Windows 7. The used software is MATLAB, in the version R2012b.

Table 5.12: Real-time measurement, computation time, averaged values from 10 repetitions.

	min	max	mean	median
prearrangement & preprocessing	694 ms	1087 ms	824 ms	736 ms
feature computation	7 ms	41 ms	7 ms	7 ms
feature computation & prediction	20 ms	65 ms	21 ms	21 ms

As can be taken from Tab. 5.12, the prearrangement and preprocessing step requires 824 ms on average and is executed only once at the beginning of each entire AR process. The time for feature computation is approximately 7 ms. The maximum value of 41 ms is reached at the very first step, where the Kalman filter, which is used to estimate roll and pitch angles for the transformation into the LLF, has to go through 32 iterations for the Smartphone or 128 for the Smartwatch. From the second instance, whose activity is to be predicted, there are only 7 respectively 25 iterations to be made, because the time window is shifted by a quarter second and overlaps on the most samples. The total time for feature computation and prediction covers 20 to 65 ms. Since there are made predictions only every quarter second, i.e. every 250 ms, the averaged 21 ms lie within this time window and amount to only 8.4 % of it.

The Smartphone used in this work includes a 1.2 GHz dual-core ARM Cortex-A9 CPU, 1 GB RAM, and operating system Android 4.3.

Based on these data, it can be assumed that AR can be carried out in real-time, even on the Smartphone. If this is not the case, it would be possible to outsource the classification process to a server and transmit the signal data and predicted activities via an interface.

5.7 Different classifiers

The C4.5 Decision Tree classifier, used in this work, should also be compared with other classifiers. Based on the training data set of Smartphone in the trouser pocket, classifiers C4.5 Decision Tree, Naive Bayes (NB), Support Vector Machine (SVM) and k-Nearest Neighbour with 10 neighbours (10-NN) are considered.

For comparison of different classifiers a 5×2-fold cross validation is applied. Five different seeds are used for the random number generator, which is needed to randomize the data, more details in Chapter 3.3.2. The overall accuracies and their averaged value are determined and the classifier with the highest mean accuracy is compared to the others by applying a paired t-test with significance level $\alpha = 0.05$. The null hypothesis H_0 states that the chosen classifier has the same accuracy as the other one compared to it. If H_0 is rejected, their accuracy is different.

These investigations were carried out with three different feature subsets. The obtained accuracy and hypothesis test results are shown in Tab. 5.13. Using only norm features or norm and LLF features, the C4.5 Decision Tree classifier achieves the highest overall accuracy. In the first case, it is more accurate than NB and SVM, but comparable to 10-NN. With the LLF features in addition, C4.5 Decision Tree outperforms all other classifiers. If all 59 features are used including norm, LLF and BF features, the SVM performs best and is significantly better than the other classifiers.

Kotsiantis et al. [25] give a good overview of the characteristics, advantages and disadvantages of the individual classifiers. An important characteristic of the C4.5 Decision Tree is comprehensibility. The algorithm is easy to understand and has a good combination of computational speed and error rate. The performance degrades with problems that require diagonal partitioning of the instances. The feature space is always divided orthogonal to the axis of one feature and parallel to the others, whereby hyper-rectangles appear as resulting regions.
In contrast, SVMs can handle multi-dimensions and continuous features very well. A large sample size improves the performance, but the interpretability is poor.
Bayesian Networks as the NB are not adapted for data containing huge amount of features, although they provide short computation time and little storage.
Large computational time and storage are required by k-NNs which are sensitive to irrelevant features. Furthermore, the accuracy depends on the number of instances in the training dataset, the probability for each feature, and the number of considered neighbours k. Therefore, the feature selection process is essential for it.
It is important to know, that no learning algorithm is generally better in performance

Table 5.13: Overall accuracies of 5×2-fold cross validation of various classifiers using different seeds s_i, paired t-test with significance level $\alpha = 0.05$, null hypothesis H_0: accuracy of best classifier (bold marked) and other classifiers is the same. Smartphone in trouser pocket, different feature subsets.

(a) norm features (#=16)

feature subset	classifier	s_1	s_2	s_3	s_4	s_5	Avg.	$-2.57 \leq \tilde{t} \leq 2.57$	test decision
a_n*, g_n*	**C4.5**	97.4%	97.6%	97.6%	97.6%	97.4%	**97.5%**	-	-
	NB	88.6%	88.6%	88.8%	88.8%	88.6%	88.7%	-11.23	H_0 rejected
	SVM	80.1%	79.9%	80.1%	80.2%	80.5%	80.2%	-26.65	H_0 rejected
	10-NN	97.3%	97.6%	97.3%	97.6%	97.6%	97.5%	-0.49	H_0 accepted

(b) norm+LLF features (#=31)

feature subset	classifier	s_1	s_2	s_3	s_4	s_5	Avg.	$-2.57 \leq \tilde{t} \leq 2.57$	test decision
a_n*, g_n*,	**C4.5**	98.5%	98.7%	98.6%	98.3%	98.8%	**98.6%**	-	-
a_l*	NB	93.0%	93.4%	93.1%	93.3%	93.1%	93.2%	-7.51	H_0 rejected
	SVM	84.2%	84.2%	84.4%	84.3%	84.5%	84.3%	-33.34	H_0 rejected
	10-NN	95.8%	95.7%	95.9%	95.9%	96.1%	95.9%	-9.95	H_0 rejected

(c) norm+BF+LLF features (#=59)

feature subset	classifier	s_1	s_2	s_3	s_4	s_5	Avg.	$-2.57 \leq \tilde{t} \leq 2.57$	test decision
a_n*, g_n*,	C4.5	99.0%	99.0%	99.1%	99.1%	99.1%	99.1%	-6.01	H_0 rejected
a_b*, g_b*,	NB	99.3%	99.2%	99.3%	99.2%	99.2%	99.3%	-2.87	H_0 rejected
a_l*	**SVM**	99.8%	99.8%	99.8%	99.8%	99.8%	**99.8%**	-	-
	10-NN	99.4%	99.4%	99.5%	99.3%	99.4%	99.4%	-16.27	H_0 rejected

than other algorithms. The algorithm, that appears to be the most accurate for the particular classification problem, must be selected.
So, overall it is a good decision to take the C4.5 Decision Tree as the classifier in this work.

5.8 Detection of the sensor's current body position

A final study of this thesis is the design of a classifier which can be used to determine the sensor position on the body. The aim is to select the appropriate classifier for AR depending on the determined sensor position. The actual location of the Smartphone on the body is determined during the activity *walking*, because the movement of the body affects certain body parts differently. The analysed positions are: *hand in front of the body* (HFB), *hand swinging* (HSW), *trouser pocket* (TP) and *jacket pocket* (JP). The activity *standing* (ST) is added additionally, on the one hand as a control to determine

whether the person is actually *walking* or not, and on the other hand, because the programmed algorithm requires at least 5 seconds of this activity to estimate a first orientation in the processing of training data. But this information will not be used, because the sensor position was constantly changing during the recording of the training data and thus can not be assumed to be stable and fixed. Moreover, the determination should be possible without knowledge of the orientation of the device. Therefore, only norm features are used as input for the classifier.

The evaluation results of the 10-fold cross validation of the C4.5 Decision Tree are displayed in Tab. 5.14. The recall values of the sensor positions are in the range of 97.8 to 99.8 %. *Standing* is detected to 100 %. From the confusion matrix one can see that sometimes *hand swinging* is confused with *trouser pocket*. This makes sense, since the movement has a direct effect on these areas.

Table 5.14: Evaluation results of 10-fold cross validation of C4.5 Decision Tree classifier, Smartphone at different body locations, norm features (acceleration and angular rate), window size=32.

(a) Recall values

# rules	15
HFB	99.8 %
HSW	98.8 %
JP	99.4 %
ST	100.0 %
TP	97.8 %
Weighted Avg.	99.1 %

(b) Confusion matrix

a	b	c	d	e	← classified as
640	1	0	0	0	a = HFB
0	984	0	0	12	b = HSW
0	2	665	0	2	c = JP
0	0	0	486	0	d = ST
0	16	0	0	698	e = TP

The built classification tree is shown in Fig. 5.27. It contains 15 nodes with rules dividing the feature spaces into areas including only instances of the same class. The first four dividing rules are illustrated as black lines in the feature 2D-plots in Fig. 5.28. g_0 and a_n1 separate *standing* and *hand in front of body* from the other classes. *Jacket pocket* can be restricted with g_0 and a_n2. The separation of the two remaining classes is more complex and uses norm features from acceleration as well from angular rate.

Finally, it can be stated that it would be possible to perform sensor position recognition before the actual AR.

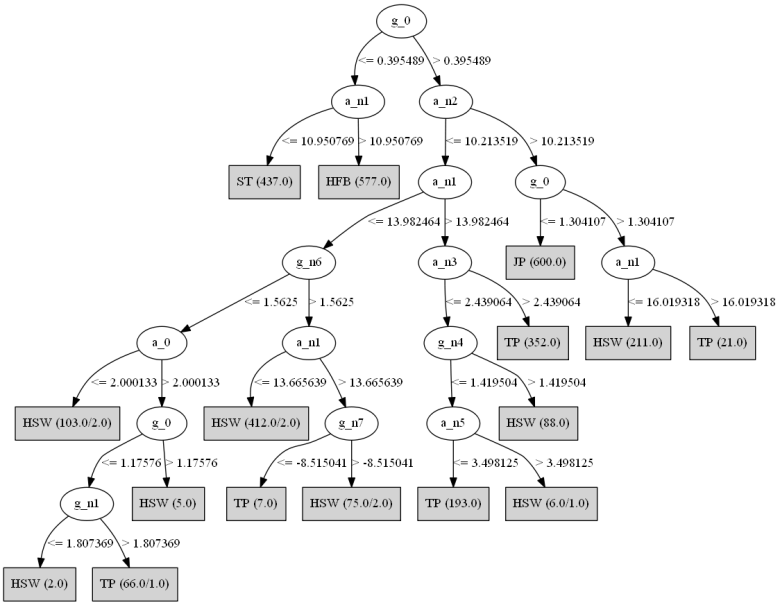
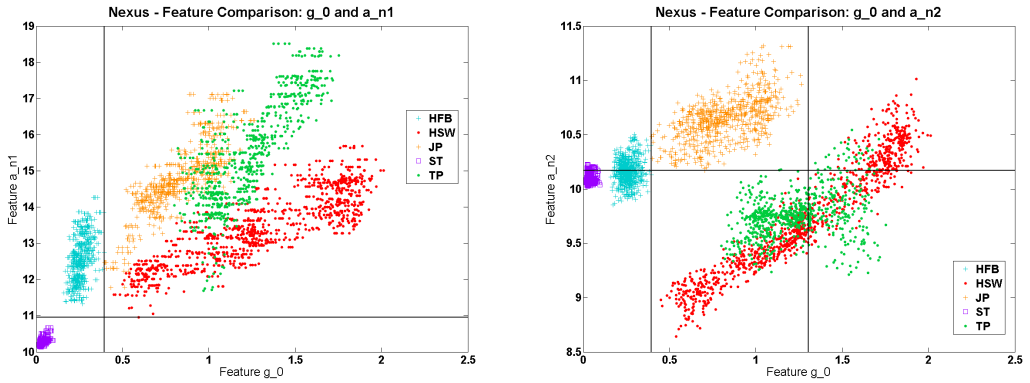


Figure 5.27: C4.5 Decision Tree built using norm features, corresponding to result in Tab. 5.14, for distinguishing between different body location of the Smartphone.



(a) g_0 and a_n1

(b) g_0 and a_n2

Figure 5.28: Exemplary feature 2D-plots showing the dividing rules used in the C4.5 Decision Trees in Fig. 5.27. Features from Smartphone at different body locations training set, window size=32.

6 Conclusions and Outlook

In this thesis, the C4.5 Decision Tree was used as a classifier in almost all investigations. The evaluation of the results was based on a 10-fold cross validation and the prediction of test data. The various conclusions that were made in this thesis are described in the following.

It could be demonstrated that the Smartphone, as a representative low-cost sensor, is suitable for Activity Recognition, because it can be compared with the capabilities of the Xsens, although of course leads to greater errors. In addition, it should be noted that magnetometer data should better not be used as feature input, since they strongly depend on surrounding electronic devices and magnetic metals. Features on the basis of local-level frame and body frame coordinates contribute to recognition accuracy, but need to be treated with caution because one can only rely on them if reliable data of the activity *standing* are available in the beginning of the recording and the device stays in place on the body. The conditions are described in more detail in Chapter 3.2.

Further investigations have shown that a window size covering a duration of approximately one second was the best choice for the Activity Recognition system, as it was designed in this work. When comparing two persons it was demonstrated that personal models perform best in the presence of user-specific activities, like walking *downstairs* or *upstairs*. If not, impersonal or hybrid models are sufficient. Accuracies of 90 % and 79 % for *upstairs* and *downstairs* with the personal model applied to test data were considered satisfactory. Moreover, it depends on the user whether and how well the user-specific activity is detected.

The comparative analysis of different feature subsets provided an overall accuracy of 99.7 % for the training data set with the Smartphone in the hand in front of the body, 99.6 % with the Smartphone in the trouser pocket and 99.9 % with the Smartwatch on the left wrist, each with a different combination of features. But there arose contradictions between the cross validation results of training data and the predicted activities of test data. The prediction results were generally acceptable, but the hardest to detect were the activities *downstairs* and *upstairs*.

The investigation of additional training data, which cover two extreme contrasts of speeds, namely fast and slow, showed that an insufficient amount of training data has been recorded. The joined data are more representative. They reached lower accuracies with cross validation and even with the predicted data a deterioration was visible, namely at *downstairs* and *upstairs*. *Walking* and *standing* were equally well recognized in all cases. Possible reasons for these contradictory and poor results may be:

- the training data did not adequately cover variations in the gait depending on mood, physical condition, clothing, or other circumstances,
- the features may have been selected inadequately,
- a C4.5 Decision Tree was perhaps not suitable for this special kind of activities in connection with such features.

In future studies it is definitely necessary to carry out further investigations, regarding features and classification methods, and to collect training data of activities that were performed with all possible speeds, and from different days, so that there are covered as many physical variations as possible.

The prediction of one single instance needed only 20 ms for the calculation with a notebook. Since the features for classification were calculated each quarter second, this is more than sufficient in order to realize a real-time application on a Smartphone.

The comparison of several classifiers on the basis of the data set with the Smartphone in the trouser pocket showed that the C4.5 Decision Tree generally outperforms other classifiers, taking into account the selected activities and features.

Potential applications for Activity Recognition (AR) in the field of navigation could be, for example, the support of a Pedestrian Dead Reckoning (PDR) system, where the activity information, more precisely whether walking on level ground or on stairs, could be useful in the reduction of errors within step length estimation. Furthermore, it has been shown to be useful in 3D indoor positioning with PDR based on Bayes (particle) filtering, see [38]. Activity information contributes to the estimation of the height and to the determination of the 2D position by constraining the possible particle space.

Conversely, it would be possible to incorporate the current location of the person as an information in the AR. In the kitchen, the activity of *lying* would be unlikely whereas it would be normal in the bedroom. On the one hand, this could increase the accuracy of AR algorithm, on the other hand, it would be an indicator of the distress of a person which could trigger a call for help.

List of Figures

List of Tables

Bibliography

[1] EvAAl competition. URL `http://evaal.aaloa.org/`. Last visited 27.10.2014.

[2] AAL Forum. AAL Joint Programme, 2014. URL `http://www.aalforum.eu/joint-progamme`. Last visited 27.10.2014.

[3] M. F. A. bin Abdullah, A. F. P. Negara, M. S. Sayeed, D.-J. Choi, and K. S. Muthu. Classification algorithms in human activity recognition using smartphones. *World Academy of Science, Engineering and Technology*, 68, 2012.

[4] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz. *Human activity recognition on smartphones using a multiclass hardware-friendly support vector machine*. Ambient assisted living and home care. Springer, 2012.

[5] J. Bancroft, D. Garrett, and G. Lachapelle. Activity and environment classification using foot mounted navigation sensors. In *Proceedings of International Conference on Indoor Positioning and Indoor Navigation*, 2012.

[6] L. Bao and S. Intille. *Activity Recognition from User-Annotated Acceleration Data*, volume 3001. Springer Berlin Heidelberg, 2004.

[7] A. Bayat, M. Pomplun, and D. A. Tran. A Study on Human Activity Recognition Using Accelerometer Data from Smartphones. *Procedia Computer Science*, 34, 2014.

[8] G. Bieber, M. Haescher, and M. Vahl. Sensor requirements for activity recognition on smart watches. In *Proceedings of the 6th International Conference on PErvasive Technologies Related to Assistive Environments*. ACM, 2013.

[9] T. Brezmes, J.-L. Gorricho, and J. Cotrina. *Activity Recognition from Accelerometer Data on a Mobile Phone*, volume 5518. Springer Berlin Heidelberg, 2009.

[10] S. Chernbumroong, A. S. Atkins, and H. Yu. Activity classification using a single wrist-worn accelerometer. In *Software, Knowledge Information, Industrial Management and Applications (SKIMA), 2011 5th International Conference on*. IEEE, 2011.

[11] Y. Cho, Y. Nam, Y.-J. Choi, and W.-D. Cho. SmartBuckle: human activity recognition using a 3-axis accelerometer and a wearable camera. In *Proceedings of the*

2nd International Workshop on Systems and Networking Support for Health Care and Assisted Living Environments. ACM, 2008.

[12] T. Choudhury, S. Consolvo, B. Harrison, J. Hightower, A. LaMarca, L. LeGrand, A. Rahimi, A. Rea, G. Bordello, and B. Hemingway. The mobile sensing platform: An embedded activity recognition system. *Pervasive Computing, IEEE*, 7(2), 2008.

[13] S. Dernbach, B. Das, N. C. Krishnan, B. L. Thomas, and D. J. Cook. Simple and complex activity recognition through smart phones. In *Intelligent Environments (IE), 2012 8th International Conference on*. IEEE, 2012.

[14] T. G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, 10(7), 1998.

[15] B. Florentino-Liano, N. O'Mahony, and A. Artes-Rodriguez. Human activity recognition using inertial sensors with invariance to sensor orientation. In *Cognitive Information Processing (CIP), 2012 3rd International Workshop on*. IEEE, 2012.

[16] F. Foerster and J. Fahrenberg. Motion pattern and posture: correctly assessed by calibrated accelerometers. *Behavior research methods, instruments, & computers*, 32(3), 2000.

[17] F. Foerster, M. Smeja, and J. Fahrenberg. Detection of posture and motion by accelerometry: a validation study in ambulatory monitoring. *Computers in Human Behavior*, 15(5), 1999.

[18] K. Frank, M. J. V. Nadales, P. Robertson, and M. Angermann. Reliable real-time recognition of motion related human activities using MEMS inertial sensors. In *Proceedings of the 23rd International Technical Meeting of the Satellite Division of the Institute of Navigation*, 2010.

[19] Google. Google Scholar, 2011. URL `http://scholar.google.com/`. Last visited 06.10.2014.

[20] N. Gyorbiro, A. Fabian, and G. Homanyi. An Activity Recognition System For Mobile Phones. *Mobile Networks and Applications*, 14(1), 2009.

[21] B. Hofmann-Wellenhof, K. Legat, and M. Wieser. *Navigation*. Springer, 2003.

[22] D. T. G. Huynh. Human Activity Recognition with Wearable Sensors, 2008. Ph.D. dissertation, Technische Universität Darmstadt.

[23] A. M. Khan, Y.-K. Lee, S. Lee, and T.-S. Kim. Human activity recognition via an accelerometer-enabled-smartphone using kernel discriminant analysis. In *Future*

Information Technology (FutureTech), 2010 5th International Conference on. IEEE, 2010.

[24] A. M. Khan, A. Tufail, A. M. Khattak, and T. H. Laine. Activity Recognition on Smartphones via Sensor-Fusion and KDA-Based SVMs. *International Journal of Distributed Sensor Networks*, 2014.

[25] S. B. Kotsiantis, I. Zaharakis, and P. Pintelas. Supervised machine learning: A review of classification techniques, 2007.

[26] N. C. Krishnan and S. Panchanathan. Analysis of low resolution accelerometer data for continuous human activity recognition. In *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*, 2008.

[27] J. R. Kwapisz, G. M. Weiss, and S. A. Moore. Activity Recognition Using Cell Phone Accelerometers. *SIGKDD Explor.Newsl.*, 12(2), 2011.

[28] M. A. Labrador and O. D. L. Yejas. *Human Activity Recognition: Using Wearable Sensors and Smartphones*. CRC Press, 2013.

[29] O. D. Lara and M. A. Labrador. A mobile platform for real-time human activity recognition. In *Consumer Communications and Networking Conference (CCNC)*. IEEE, 2012.

[30] M. Lee, J. Kim, K. Kim, I. Lee, S. H. Jee, and S. K. Yoo. Physical activity recognition using a single tri-axis accelerometer. In *Proceedings of the world congress on engineering and computer science*, volume 1, 2009.

[31] J. Lester, T. Choudhury, and G. Borriello. *A Practical Approach to Recognizing Physical Activities*, volume 3968. Springer Berlin Heidelberg, 2006.

[32] P. Liu, Y. Chen, W. Tang, and Q. Yue. Mobile WEKA as Data Mining Tool on Android. In *Advances in Electrical Engineering and Automation*. Springer Berlin Heidelberg, 2012.

[33] J. W. Lockhart and G. M. Weiss. The Benefits of Personalized Smartphone-Based Activity Recognition Models. *In Proc. SIAM International Conference on Data Mining*, 2014.

[34] X. Long, B. Yin, and R. M. Aarts. Single-accelerometer-based daily physical activity classification. In *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, 2009.

[35] A. Mannini and A. M. Sabatini. Machine learning methods for classifying human physical activity from on-body accelerometers. *Sensors*, 10(2), 2010.

[36] U. Maurer, A. Smailagic, D. P. Siewiorek, and M. Deisher. Activity recognition and monitoring using multiple sensors on different body positions. In *Wearable and Implantable Body Sensor Networks, 2006. International Workshop on*, 2006.

[37] E. Miluzzo, N. D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S. B. Eisenman, X. Zheng, and A. T. Campbell. Sensing meets mobile social networks: the design, implementation and evaluation of the CenceMe application, 2008.

[38] T. Moder, P. Hafner, K. Wisiol, and M. Wieser. 3D Indoor Positioning with Pedestrian Dead Reckoning and Activity Recognition Based on Bayes Filtering. In *Indoor positioning and indoor navigation (IPIN), 2014 international conference on*, 2014.

[39] M. J. V. Nadales. Recognition of Human Motion Related Activities from Sensors, 2010. Master thesis, University of Malaga.

[40] National Instruments. The Fundamentals of FFT-Based Signal Analysis and Measurement in LabVIEW and LabWindows/CVI, 2009. URL `http://www.ni.com/white-paper/4278/en/#toc2`. Last visited 07.10.2014.

[41] J. Parkka, M. Ermes, P. Korpipaa, J. Mantyjarvi, J. Peltola, and I. Korhonen. Activity classification using realistic data from wearable sensors. *Information Technology in Biomedicine, IEEE Transactions on*, 10(1), 2006.

[42] S. Pirttikangas, K. Fujinami, and T. Nakajima. *Feature selection and activity recognition from wearable sensors.* Ubiquitous Computing Systems. Springer, 2006.

[43] J. R. Quinlan. *C4.5: programs for machine learning.* Morgan Kaufmann, 1993.

[44] A. Rasekh, C.-A. Chen, and Y. Lu. Human activity recognition using smartphone. *arXiv preprint arXiv:1401.8212*, 2014.

[45] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman. Activity recognition from accelerometer data. In *AAAI*, volume 5, 2005.

[46] Statista GmbH. Ist Ihr Mobiltelefon ein Smartphone? Umfrage zum Besitz von Smartphones in Österreich bis 2014, 2014. URL `http://de.statista.com/statistik/daten/studie/322885/umfrage/umfrage-zum-besitz-von-smartphones-in-oesterreich/`. Last visited 14.10.2014.

[47] Statista GmbH. Etwa zwei von fünf Deutschen würden eine Smartwatch tragen, 2014. URL `http://de.statista.com/infografik/2690/bekanntheit-und-nutzungsbereitschaft-von-smart-glasses-und-smartwatches-in-deutschland/`. Last visited 14.10.2014.

[48] Statista GmbH. Aktivitätenkontrolle wichtigstes Einsatzgebiet von Smart-

watches, 2014. URL `http://de.statista.com/infografik/2681/wichtigste-funktionen-von-smartwatches/`. Last visited 14.10.2014.

[49] M. Susi, V. Renaudin, and G. Lachapelle. Detection of quasi-static instants from handheld MEMS devices. In *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on*. IEEE, 2011.

[50] E. M. Tapia, S. S. Intille, W. Haskell, K. Larson, J. Wright, A. King, and R. Friedman. Real-time recognition of physical activities and their intensities using wireless accelerometers and a heart rate monitor. In *Wearable Computers, 2007 11th IEEE International Symposium on*. IEEE, 2007.

[51] D. Titterton and J. L. Weston. *Strapdown inertial navigation technology*, volume 17. IET, 2004.

[52] G. M. Weiss, J. W. Lockhart, T. T. Pulickal, P. T. McHugh, I. H. Ronan, and J. L. Timko. Actitracker: A Smartphone-based Activity Recognition System for Improving Health and Well-Being, 2014.

[53] I. H. Witten, E. Frank, and M. A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques (Third Edition)*. Morgan Kaufmann, Boston, 2011.

[54] W. Wu, S. Dasgupta, E. E. Ramirez, C. Peterson, and G. J. Norman. Classification accuracies of physical activities using smartphone motion sensors. *Journal of medical Internet research*, 14(5), 2012.

[55] Xsens Technologies B.V. MTi-G User Manual and Technical Documentation. URL `http://www.xsens.com/products/mti-g/`. Last visited 14.10.2014.

[56] J. Yang. Toward physical activity diary: motion recognition using simple acceleration features with mobile phones, 2009.

[57] J.-Y. Yang, Y.-P. Chen, G.-Y. Lee, S.-N. Liou, and J.-S. Wang. *Activity recognition using one triaxial accelerometer: A neuro-fuzzy classifier with feature reduction*. Entertainment Computing-ICEC 2007. Springer, 2007.

[58] J.-Y. Yang, J.-S. Wang, and Y.-P. Chen. Using acceleration measurements for activity recognition: An effective learning algorithm for constructing neural classifiers. *Pattern Recognition Letters*, 29(16), 2008.

[59] B. Yuan, J. Herbert, and Y. Emamian. Smartphone-based Activity Recognition Using Hybrid Classifier. In *Proceedings of the 4th International Conference on Pervasive and Embedded Computing and Communication Systems (PECCS2014)*, 2014.