EFFICIENT FEATURE SELECTION FOR ONLINE ACTIVITY RECOGNITION ON SMART PHONES

by

Erman Doğan
B.S., Computer Engineering, Boğaziçi University, 2008

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering
Boğaziçi University
2013

ACKNOWLEDGEMENTS

This work is supported partially by the Turkish State Planning Organization (DPT) under the TAM Project, number 2007K120610, and by the Bogazici University Research Fund under the project number 6056.

ABSTRACT

EFFICIENT FEATURE SELECTION FOR ONLINE ACTIVITY RECOGNITION ON SMART PHONES

Activity Recognition (AR) is an active area of research that has direct applications on life quality and health of human beings. Related studies aim to classify different daily activities of people with high accuracy rates using various types of sensors. Becoming an essential part in our daily lives, smartphones are now suitable tools that enable people to make use of AR technologies without being obliged to use or wear some extra device. However, due to power and computational constraints of these devices, it becomes a challenging task to attain accurate results by using power and CPU-intensive classifiers. In this study, we present an efficient selection of features to attain high accuracies in recognizing five daily activities with a lightweight classifier, K Nearest Neighbors (KNN). Since previous studies in this area show that it is possible to obtain high recognition performance with the KNN classification algorithm, we focused on the problem of feature selection to see how far this performance can be enhanced by employing the most appropriate feature sets for the KNN algorithm. We use some well-known features together with some more specific features and in order to keep the system energy-efficient, all features are extracted from the readings of a single accelerometer on a smartphone that is carried in the trousers' pocket with different orientations. In this study, we also evaluated the effect of different window lengths and window functions that are used for segmenting the data prior to feature extraction. The results show that by having an efficient selection of features it is possible to obtain promising accuracy rates with a simple classification algorithm like KNN which facilitates online and real-time activity recognition on smartphones.

ÖZET

AKILLI TELEFONLAR ÜZERİNDE ÇEVRİMİÇİ EYLEM TANIMA İÇİN ETKİN ÖZNİTELİK SEÇİMİ

Eylem Tanıma konusu yaşam kalitesi ve insan sağlığı ile ilgili doğrudan uygulamaları olan ve günümüzde aktif olarak çalışılan bir araştırma alanıdır. Bu konudaki çalışmalar insanların gün boyunca gerçekleştirdikleri eylemleri farklı türlerde algılayıcılar kullanarak yüksek doğruluk oranıyla sınıflandırabilmeyi amaçlamaktadır. Günlük yaşantımızın vazgeçilmez parçalarından biri haline gelen akıllı telefonlar, insanların eylem tanıma teknolojilerinden ilave bir alet kullanma ya da taşıma zorunluluğu olmaksızın faydalanmalarını sağlayabilecek cihazlardır. Fakat bu cihazların güç ve hesaplama açılarından kısıtlı kaynaklara sahip olması, bu cihazlar üzerinde güç-yoğun ve işlemci-yoğun sınıflandırma yöntemleri ile yüksek doğruluk oranları elde etmeyi zorlaştırmaktadır. Bu çalışma hafif bir sınıflandırma algoritması (KNN) ile beş farklı günlük eylemin yüksek doğruluk oranı ile tanınabilmesini sağlayacak bir etkin öznitelik seçimi sunmayı amaçlamaktadır. Bu doğrultuda, bazı özgün öznitelik tipleri ile yaygın olarak kullanılan diğer bazı öznitelik tipleri bir arada kullanılmıştır. Ayrıca, sistemi enerji verimli kılmak adına tüm bu öznitelikler yalnızca pantolon cebinde taşınan bir akıllı telefondaki yerleşik ivmeölçerden alınan verilerden çıkarılmıştır. Bu çalışmada ayrıca öznitelik çıkarımından önce algılayıcıdan edinilen veriyi pencerelemek için kullanılan farklı pencere uzunlukları ve pencere işlevlerinin etkisi de değerlendirilmiştir. Elde edilen sonuçlar gösteriyor ki etkin öznitelik seçimi, KNN gibi gerçek-zamanlı ve çevrimiçi sınıflandırmaya uygun, basit bir sınıflandırma algoritması ile umut veren doğruluk oranlarına ulaşmayı mümkün kılmaktadır.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

A	Sequence of accelerometer magnitude values collected
a_{mag}	Magnitude of the acceleration vector
a_x	Magnitude of acceleration on x-axis
a_y	Magnitude of acceleration on y-axis
a_z	Magnitude of acceleration on z-axis
$\hat{f}$	Normalized value of the feature
F_C	Sequence of FFT coefficients
F_{C_i}	i^{th} FFT coefficient
R	Autocorrelation function
R_i	Value of autocorrelation for a lag of i samples

LIST OF ACRONYMS/ABBREVIATIONS

AR	Activity Recognition
DFT	Discrete Fourier Transform
DHMM	Discrete Hidden Markov Models
DT	Decision Tree
FFT	Fast Fourier Transform
HMM	Hidden Markov Models
KMC	K Means Clustering
KNN	K Nearest Neighbor
NN	Nearest Neighbor
QDA	Quadratic Discriminant Analysis
SBS	Sequential Backward Selection
SFS	Sequential Forward Selection
SVM	Support Vector Machines

1. INTRODUCTION

Human activity recognition using sensing technologies refers to the classification of a person's physical activities like walking, sitting, falling, driving or running, with different sensing modalities, such as motion, location sensors. Activity recognition (AR) solutions have a wide range of application areas including patient monitoring in health care services, child and elderly care and sports or fitness monitoring for computing daily energy expenditure [3–10].

Earlier studies on human activity recognition were commonly making use of cameras for visual analysis of human movements [11]. After the first use of small, wearable accelerometers for monitoring human physical activities in 90's, a wide range of research has been carried out in order to make effective utilization of these small, low-power and cheap devices for activity recognition. Thanks to the improvements in microsensor technology, AR with acceleration sensors has become a popular field of research starting with the studies at MIT [12].

The acceleration sensor (accelerometer) is the most common device used in AR studies, especially in those aiming at physical activities [13, 14]. However, the way researchers make use of this tool and the types of other assistant devices they use do vary. Some of the major characteristics of an AR study are the number of accelerometers used, variation in position and orientation of the device, usage of supporting data like user activity history or field map and usage of supporting devices like GPS receivers. Another important point about an AR study is the type of hardware used: commercial devices [15], custom hardware [16, 17] and mobile phones [18, 19]. Among these, mobile phones obviously differ from others in being adaptable to a person's everyday life without any perceivable change or burden. Today, mobile phones with relatively powerful and rich hardware, including a set of integrated sensors such as accelerometer, microphone, camera, GPS and compass are ordinary parts of our daily lives. Thus, successful application of activity recognition solutions on mobile phones will not cause any extra cost or intervention to a person's daily life. On the other hand, one should consider the typical resource constraints of mobile phones like the limited battery power. Also, the processing power of these devices should be utilized efficiently since they are usually not dedicated to AR applications, but there are other primary tasks they should carry out like phone calls, internet browsing, playing music. The interruption of any of these tasks will disturb the user. So, a candidate AR solution should be efficient enough to keep the user experience level high. As a result, these constraints require the implementation of feature extraction and classification algorithms that have less complexity, when running on the phone. In relation to this, Könönen *et al.* show that even a simple linear classifier can provide good recognition accuracies when a suitable set of features are selected [20].

There are two groups of activity recognition systems that are offline and online systems. In the context of activity recognition, *online* systems commonly refer to those that employ real-time classification that is run mostly on sensory or intermediary devices and simultaneously with sensor data streaming. On the other hand, *offline* systems are able to classify larger datasets on central computing units and they are not practically restricted in terms of time and power. Thus, the most important characteristics of an online AR system is the time and power restrictions for classification and its convenience for interactive, mobile and participatory applications, whereas offline solutions are suitable if the analysis of sensor readings over relatively longer time periods is sufficient for the purposes of an application. A typical application of an offline AR system can be seen in [21] where the purpose of the study is inferencing daily physical activity diaries of people and an example of an online solution is used in [22] for participatory sensing.

On the other hand, training of both online and offline systems is a heavy task. In order to have a system that provides high recognition accuracy besides being independent of subjects, environment, time, etc., a sufficient amount of data should be processed for creating a training model. Overcoming these context dependencies is an issue for online activity recognition systems as much as for offline ones, but it is generally not very efficient to process large datasets for training the system on devices with limited capabilities that are used for online recognition, since the time such heavy tasks take will be too long to be acceptable for real-time processing. Therefore, the computation of an efficient training model that is to be used for online classification should be performed offline [15, 23]. This is also the case for the KNN classification algorithm. Since this algorithm compares each sample with the whole training dataset, a large training dataset will directly cause delays in the classification process. So, a smaller but representative dataset should be formed to be used on the mobile platform for obtaining high accuracy rates in real-time.

The main steps of activity recognition from sensory data include i) sampling of the sensors, i.e. data collection, ii) feature extraction from the collected data, and iii) classification [13]. Feature extraction step includes the generation of abstractions that accurately characterize the sensor data, or in other words, that represent the original data in the best way. The classification phase includes mapping the collected data, i.e. the feature set, to a set of activities. Usually, machine learning algorithms are used in the classification phase [13]. Although the classification step gives the final decision about recognizing an activity, extracting the best features that characterize raw signals is also equally important. Examples of features can be mean, variance in the time domain and spectral energy, spectral entropy in the frequency domain, or wavelet coefficients in the time-frequency domain. There are also examples of heuristic features such as inter-axis correlation, signal vector magnitude and signal magnitude area which are widely used [16]. In most of the activity recognition studies, a fixed set of features are selected for a set of activities, only a few studies investigate the impact of selected features on the performance of recognition [24, 25].

In this study, we develop an online activity recognition software which attains high recognition rates in classifying five daily activities that are walking, running, bicycling, motorized transport and stationary state (sitting/standing/lying) with efficient feature selection techniques. Considering the results obtained by our research group in previous studies in which the performance of several different classifiers are compared for online activity recognition [26] and the high performance of simple classifiers mentioned by Könönen *et al.* [20], we focused on extracting efficient features that would yield high recognition rates with a lightweight, simple classifier like K-Nearest Neighbor (KNN).

In this regard, we evaluated the performance of various time and frequency-domain features. These include some common features used in most of the related works (e.g. variance and FFT coefficients) as well as some more specific features that are extracted from the autocorrelation function of the accelerometer signal. Since wavelet coefficients were shown to perform worse than frequency-based features [25] with accelerometer-based activity recognition and considering the complexity of wavelet calculation on resource-limited smart phones with online recognition, we focused only on the time domain and frequency domain features. Also, we employed feature selection techniques in order to find the most suitable set of features for maximizing the performance of the KNN classifier.

For collecting the activity data, we used only a built-in tri-axis accelerometer on a mobile phone. This contributed to the robustness of the system since accelerometers are hardly affected by environmental conditions unlike other sensors like GPS receivers which work only outdoors and compasses which are sensitive to magnetic fields. Using a single accelerometer also helped us in minimizing the power consumption most of which is usually caused by sensory devices in similar systems. Moreover, we extracted all features from the square sum of three acceleration components on x, y and z-axis (i.e. the magnitude of the acceleration vector) rather than making use of these three components separately. Using only the acceleration magnitude provided us a system that is not affected by the orientation of the phone.

Thus, over other activity recognition solutions on mobile platforms this study has the following advantages:

- Real time classification of activities.
- Flexible orientation of the mobile phone.
- High performance with a lightweight classifier.

The development of the system involves several steps that are presented in Figure 1.1. In Step 1, we collected the activity data from 10 subjects to be used for analyzing the classifier performance with a full set of features using the leave-one-out cross val-

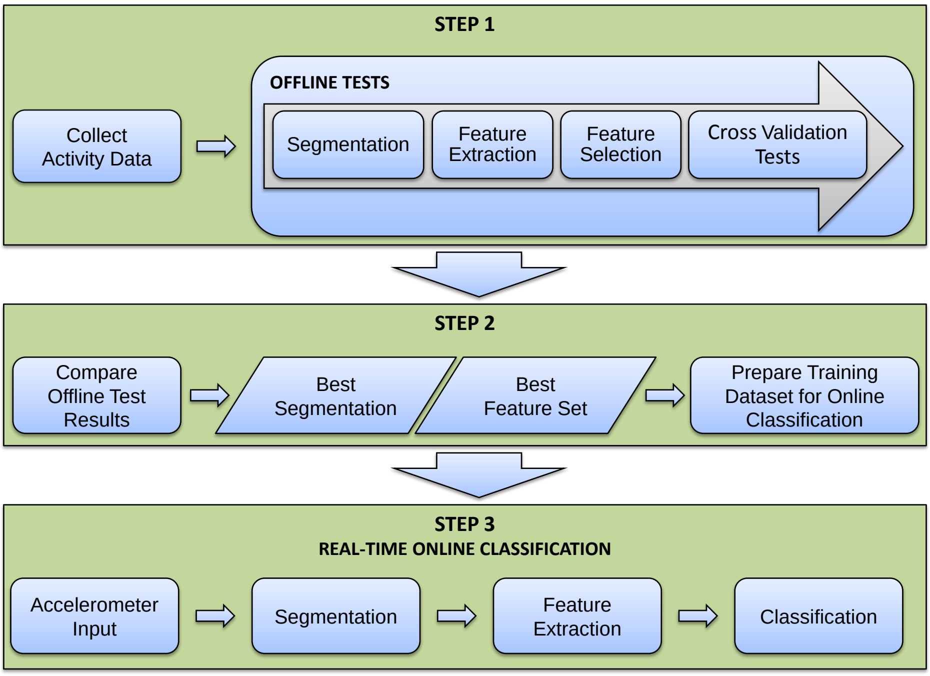


Figure 1.1. Block diagram of the activity recognition study.

idation technique. In the offline analysis, we determined the ideal set of features that gives the best classification performance using feature selection methods. Besides, we decided on the best values for several data pre-processing parameters like window size, window overlap and window function to be applied. In Step 2, we prepared training datasets for the online recognition software that we developed. In the last step, we evaluated the performance of online classification by running the activity recognition software on a mobile phone that is carried by subjects while performing the activities.

The rest of the thesis is organized as follows. In Chapter 2, we provide some background information about concepts, tools and techniques that are used in this study. In Chapter 3, we present related works in the context of feature selection and activity recognition on smart phones. In Chapter 4, the experiment design for evaluating the performance of the proposed online activity classification system is explained. In Chapter 4, we also give information about the main components and functionalities of the applications that we implemented for developing and evaluating the proposed system in both online and offline modes. Chapter 5 includes the evaluation of the results of both offline and online tests. In Section 6, we provide our conclusions and directions for future research.

2. BACKGROUND

In this part, we give some introductory information about the background concepts, tools and techniques which are used throughout this study. Particularly, the details on how accelerometers work, the overall activity recognition procedure and the algorithms that can be employed in different steps of an AR procedure are presented in this chapter.

2.1. Accelerometers

Acceleration of an object is the rate of change in its velocity which can be caused by a constant or a dynamic force. For example, the speed of a falling object increases due to the force of gravity and a stationary ball gathers speed when it is kicked. The electromechanical devices that help us in measuring the acceleration are called *accelerometers*.

Accelerometers are useful tools for obtaining environmental information regarding the changes in speed, orientation and position. For instance, one can detect the slope of a surface since the direction of gravitational force can be extracted from the readings of accelerometer that is placed on that surface or one can carry out vibration tests on an industrial machine. In the context of activity recognition, accelerometers that are embedded in different types of hardware like wearable equipments, mobile phones and custom devices are used to estimate the activity that a subject is performing by interpreting the acceleration of different body parts by attaching these devices to arms, legs, etc., or putting them in a pocket, a bag, etc.

In this study, we use a smartphone (Samsung Galaxy Ace S5830) with a built-in triaxial accelerometer. The device outputs the acceleration values on 3 perpendicular axes with a rate of at most 100Hz. In Figure 2.1, an illustration of the accelerometer's axes in reference to the smartphone is presented.

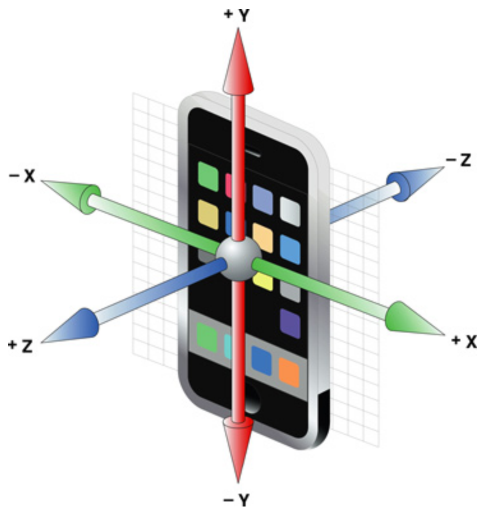


Figure 2.1. Orientation of the accelerometer axes on a mobile phone [1].

2.2. Activity Recognition Process

Different activity recognition solutions that are proposed so far involve several common processing steps. These are preprocessing, segmentation, feature extraction, feature selection and classification [13]. We give brief explanations for each of these steps in the following subsections.

2.2.1. Preprocessing

This step refers to all sorts of processing that remove redundant and useless information from the raw data or transform the data to a more appropriate form for feature extraction. Possible operations that can be done in the preprocessing step are removal of noise, smoothing the signal and converting an unequally-sampled time series to an equally-sampled one. Some of the filters used for removing potential noise, smoothing the signal and removing DC offset in the related works are low pass [16], high pass [27], Laplacian [9] and Gaussian [28] filters.

2.2.2. Segmentation

Segmentation (windowing) is the process of dividing a signal into variable or fixed-sized windows in order to make it easier to extract features from the signal.

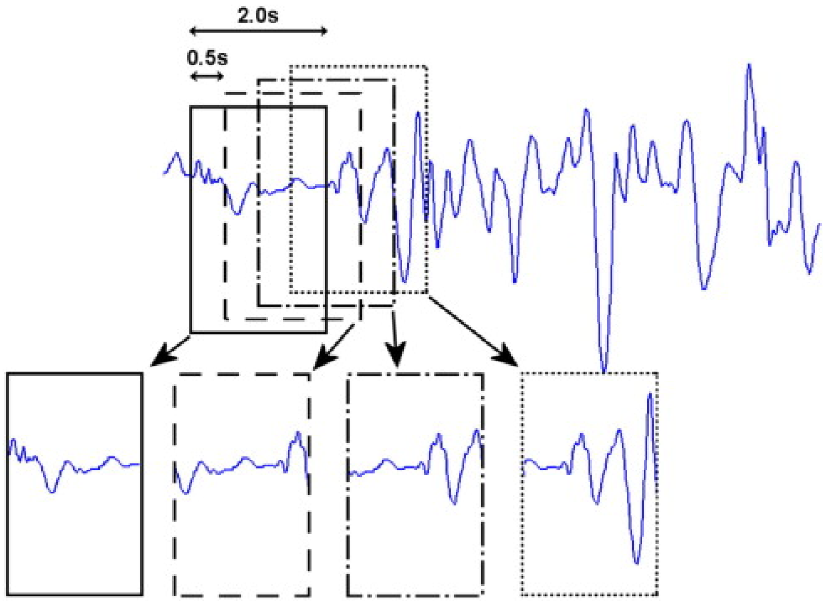
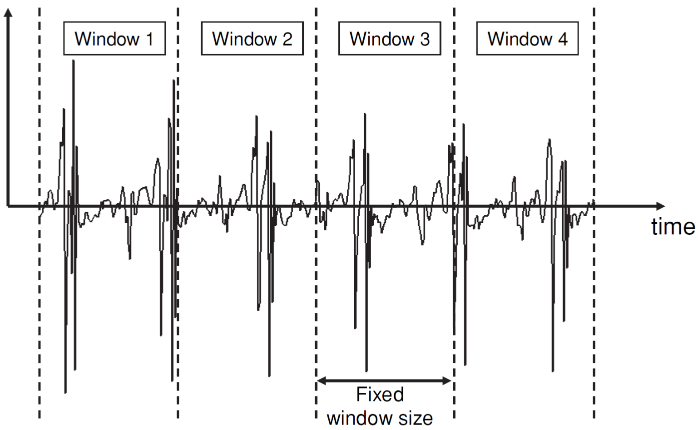


Figure 2.2. Segmentation with sliding window technique.

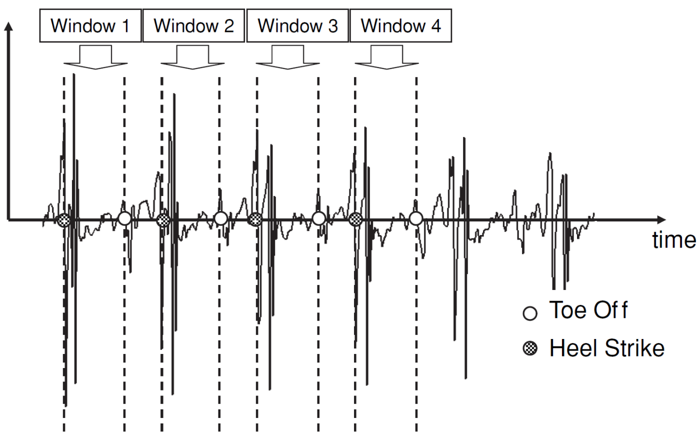
Sliding window algorithms are the most popular ones among segmentation techniques since they are simple and lightweight. Sliding window algorithms split the signal into fixed-length windows by leaving no gaps between consecutive windows [29]. Also, it is a common approach to overlap adjacent windows in these algorithms [25, 30].

In Figure 2.2, an application of the sliding window algorithm is illustrated. In this figure, the signal is split into 2-second long windows with 75% overlap ratio. If we employ this segmentation scenario for real time processing, it will mean that the system should process a data window for every 0.5 second.

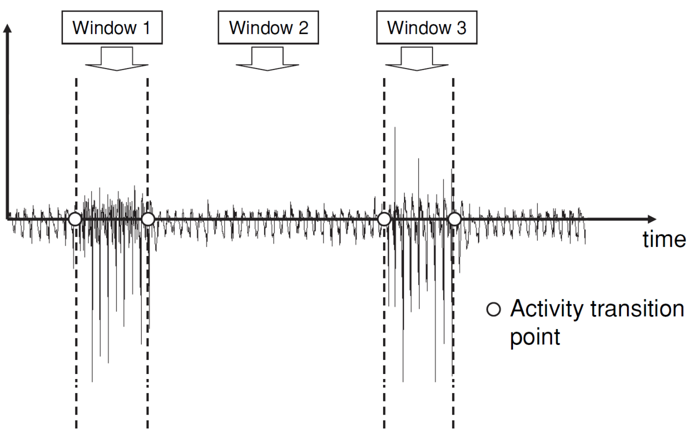
There are also studies that use event-based or activity-based segmentation techniques [31, 32]. As illustrated in Figure 2.3, these techniques decide on how to divide the signal into successive windows by considering the events or the activity transitions detected in the signal. Since these methods require an extra preprocessing phase for detecting the time of event or transition, they are usually not preferred for real-time solutions.



(a)



(b)



(c)

Figure 2.3. Defining (a) sliding windows, (b) event-based windows and (c) activity-based windows along a continuous sensor signal [2].

2.2.3. Feature Extraction

Features can be described as indicators for different characteristics of a data segment. By extracting a set of features from a large data window, we can obtain a simple vector that is much easier to process for a classifier for differentiating similar data segments. Such value sets are called *feature vectors*. Features can be computed by time, frequency and time-frequency analysis.

In this study, we focused on time-domain and frequency-domain features. Time domain involves the analysis of signals with respect to time and a time domain graph plots the changes in the value of a signal over time. On the other hand, frequency domain refers to the analysis of frequency components of a signal. Unlike time domain graphs, the representation of frequency domain shows how much of the signal lies on each frequency band.

The computation of the frequency-domain features requires intensive pre-processing (e.g. windowing, FFT) whereas time-domain features can be computed without such pre-processing tasks but the transmission errors like noise and packet-loss should still be taken into account. The most popular transformation method between time and frequency domains is the Fourier transform which can be used for decomposing a signal into the sum of a number of sine wave components and for recovering the original signal from these components. The computational complexity of the Fast Fourier Transform (FFT) is $n \log(n)$ where n is the size of the sample window which is expected to be a power of two. FFT is the fastest transformation method between time and frequency domains.

2.2.4. Feature Selection

Employing more features does not always yield better classification accuracy. Different features often mask each other's distinctiveness when they are used together by a classifier or the same sort of information may be contained by multiple features which will cause redundancy and increased computational cost. Due to this fact, one should select an ideal subset of features that are extracted from the input signal in order to obtain better recognition performance.

One of the most popular feature selection methods used in the previous works is the sequential forward-backward search [17, 33]. The *Sequential Forward Selection* (SFS) algorithm starts with an empty set of selected features. At each iteration, candidate features are compared according to the increase they provide in the classification accuracy and the best one is added to the selected features. The algorithm stops when no more improvement can be obtained with the remaining features. Unlike SFS, the *Sequential Backward Selection* (SBS) algorithm starts with a full set of selected features and removes a feature at each step by again considering the amount of improvement in the classification performance. The process stops when it is not possible to improve the performance by removing any of the features from the selection.

In this thesis, we decided to use SFS and SBS methods for feature selection, since they are the most widely used methods in the literature besides being very easy to implement. By the way, both of these algorithms apply a greedy optimization approach on an initial feature set. So, the major weakness of these techniques is that they can easily get stuck in local maxima. For avoiding this problem partially, we used these algorithms in two different approaches as described in Section 5.1.

2.2.5. Classification

Classification is the final step of a recognition process. Having a set of categories which is supposed to cover all possible observations, the classification process involves the identification of the category that an observation belongs to. The most widely used classification methods in activity recognition are as follows.

- *K-Nearest Neighbors* (KNN) method simply classifies a sample based on K closest training samples. It is a lightweight algorithm which is easy to implement [20, 23, 25, 26].
- *Decision Tree* (DT) algorithms implement decision models with a tree structure

where leaves represent categories and branches represent decision rules [21,23,29].

- *Support Vector Machines* (SVM) are a group of supervised learning models which can be used for classification [20, 21, 29].
- *Naive Bayes* (NB) is a simple probabilistic classifier. It employs the Bayes' theorem with the assumption that the presence of a feature in a class is not dependent on the presence of another feature [15, 21, 26, 29].
- *Hidden Markov Models* (HMM) are statistical models which assumes that the system is a Markov model with unobserved states [29, 34, 35].

3. RELATED WORK

Activity Recognition using wearable or visual sensors has become a popular field of research in the last decade. Although plenty of research is carried out on this topic since 2000 [13], there are several parameters that make it a difficult task to compare the results of these studies since they all evaluate the performance with different parameters. Some of these parameters are the type of hardware, the number of sensors, the set of activities to be classified and how sensors are placed.

In Table 3.1, characteristic features of related studies are presented. These features include the set of activities to be classified, sensor/hardware types used for data collection, extracted features and the clasification algorithms.

The most common *hardware* types that have been used in AR studies are wearable sensors [25, 36, 37] and mobile phones [38–42]. Multiple wearable sensors can be used in combination for getting the motion data from different parts of the body unlike mobile phones which are typically tested in accordance with their regular usage (e.g. a single mobile phone carried in a pocket). However wearable sensors are usually have limited processing capability and it is not possible to implement real-time classification solutions on them, whereas modern mobile phones can be used as a sensor and a classifier at the same time. Additionally, mobile phones can be integrated seamlessly into practical AR applications since they already provide several other functions that have regular daily usage by people. Other than these two hardware options, some custom sensor boards can be used like in [24].

Table 3.1. Related Work.

Paper	Activities	Sensors	Hardware	Accelerometer Position and Orientation	Features	Classifiers	Recognition Results	Online Classification
[20]	bicycling, soccer, lying, nordic walking, rowing, running, sitting, standing, walking	Multiple Accelerometers	Wearable Sensors (Wrist, Hip)	Fixed Position and Orientation	Mean, max, min, max-min diff., variance, power spectrum entropy, peak frequency, peak power. SFS and SFFS are used for feature selection.	Min. Distance, DT + DHMM, KNN, SVM	80% (SVM), 78% (KNN), 73% (MDC)	No
[25]	level walking, walking upstairs/downstairs, jogging, running, hopping on left/right leg, jumping	Multiple Accelerometers	Wearable Sensors (Waist, Thigh, Ankle)	Fixed Position and Orientation	7 sets of wavelet features, 3 sets of time-domain features, 4 sets of freq-domain features, First 5 FFT Coefficients provided the highest performance	KNN	92% (1 Sensor), 96% (3 Sensors)	No
[37]	resting, typing, gesticulating, walking, running, bicycling	Multiple Accelerometers	Wearable Sensors (Wrist, Hip, Ankle), Mobile Phone	Fixed Position and Orientation	"Motion Intensity" of body parts computed from accelerometers. "Intensity" is proportional to variance.	Neural Network	80% Average Accuracy	No

Continued on the next page

Table 3.1. Related Work (cont.).

Continued from previous page

Paper	Classified Activities	Sensors	Hardware	Accelerometer Position and Orientation	Features	Classifiers	Recognition Results	Online Classification
[29]	stationary, walking, running, bicycling, motorized transport	Accelerometer, GPS	Mobile Phone	Mobile phone is carried in different positions and flexible orientations: arm, waist, chest, hand, pocket, bag	Mean, variance, energy, DFT of total accelerometer magnitude and GPS speed are evaluated using Correlation-Based Feature Selection. Variance, 1-3Hz DFT coeff.s and GPS speed are selected as feature set.	DT, KMC, NB, NN, SVM, DT + DHMM, Continuous HMM,	93% (DT+DHMM), 91% (NN or SVM)	No
[15]	walking, jogging, sitting, bicycling on a stationary bike	Accelerometer, Foot Sensor	Mobile Phone, Nike Sport Kit	Fixed Position, Carried in pocket.	”Foot in Use” Feature, mean, stddev, min, max of accelerometer magnitude, Energy in each band of 10 freq. components of DFT, The value and index of the largest DFT component	NB	97% Average Accuracy	Yes
[21]	sitting, standing, walking, running, driving, bicycling	Accelerometer	Mobile Phone	Fixed Position, Carried in pocket.	Mean, stddev, zcr, entropy, etc. Features are computed from horizontal/vertical components and magnitude of acceleration vector.	DT, NB, KNN, SVM	90% (DT), 89% (KNN), 89% (SVM)	No
[26]	walking, running, standing, sitting	Accelerometer	Mobile Phone	Fixed Position, Carried in pocket.	Average, minimum, maximum, standard deviation	Clustered KNN, NB	92% (KNN)	Yes

Continued on the next page

Table 3.1. Related Work (cont.).

Continued from previous page

Paper	**Classified Activities**	**Sensors**	**Hardware**	**Accelerometer Position and Orientation**	**Features**	**Classifiers**	**Recognition Results**	**Online Classification**
[23]	walking, running, cycling, driving, idling	Accelerometer	Mobile Phone	Fixed Position, Carried in pocket.	Mean, minimum, maximum, standard deviation, different percentiles, square sum of observations above/below certain percentiles	DT, KNN, QDA	95% (DT+QDA), 94% (DT+KNN)	Yes
[38]	sit-to-stand, stand-to-sit	Accelerometer	Mobile Phone	Fixed Position, Carried in pocket.	Vertical acceleration waveform	Custom Method: Cross correlating vertical acceleration waveform with characteristic waveform of sit-to-stand transition.	70% Average Accuracy	Yes
[24]	walking, standing, jogging, skipping, hoppin, riding bus	Accelerometer, Digital Compass, Light Sensor	Custom Sensor Board	Fixed Position and Orientation	Mean, variance, energy, spectral entropy, DFFT coefficients, Correlation of acceleration in all three axes, Variance of digital compass, Variance of light sensor	KMC	N/A	No

One of the most important factors that determines the performance of an activity recognition system is certainly the *classifier* algorithm that is used. Although different classifiers may yield high recognition rates with different sets of features, the successful results attained by the KNN algorithm in several AR studies is remarkable [20, 23, 25, 26]. The results achieved by this lightweight classifier encouraged us to choose the KNN algorithm in this study for evaluating the role of feature selection in activity recognition. In [20], KNN and SVM classifiers attain the two highest recognition accuracies which are 78% and 80% respectively and in [21], KNN and SVM again achieve an equally high classification performance of 89% whereas NB algorithm achieves only 75%. In [23], the KNN classifier and Quadratic Discriminant Analysis (QDA) are used along with a DT algorithm and both attain very high and close accuracy rates that are 94% and 95% respectively. Lastly in [29], KNN and SVM again attain the same accuracy rate (91%) and that is very close to the accuracy attained by a classifier which is a combination of Discrete Hidden Markov Models (DHMM) and DT algorithms (93%).

Another significant factor that characterizes different AR studies is the selection of *sensors* for collecting data. The most widely used sensor for recognizing human physical activities is the accelerometer. However, the number of accelerometers used for activity recognition and how they are placed on the human body vary. There are different setups for the accelerometer usage in different studies like placing one or more wearable accelerometers on different body parts like wrist, ankle, thigh and hip and using the accelerometer embedded in a mobile phone that is carried in pocket. Recent studies usually make use of accelerometers in common, however these sensors are sometimes used along with some supporting sensors like GPS [20, 29], digital compass [24], foot sensor [15] and heart rate sensor [20]. In this thesis, only an accelerometer that is embedded in a mobile phone is used for sensing purposes. The elimination of the energy consumption that may be caused by extra sensors and the high availability of a single accelerometer for smartphone owners are the two important factors behind this decision.

The set of *activities* to be classified is another design parameter which makes it difficult to compare the results of different studies. The set of activities can consist of specific activity transitions like in [38] where sit-to-stand and stand-to-sit transitions are classified or it can contain routine daily activities like walking, sitting and motorized transport. Another significant difference between the previous studies is the evaluation of idle activities like sitting, standing and lying. While some studies take all these activities as one class of activity called *idling* or *stationary* [23, 37], others consider them as separate activities [21, 26]. Also, there are other studies which concentrate on sportive activities like soccer, rowing and jogging [20, 25]. In this study, we focus on typical daily activities of a person which can be classified using the acceleration of a mobile phone that is carried in a pocket. Thus, distinguishing activity pairs like typing-resting [37], sitting-standing [26], soccer-rowing [20] is beyond the scope of this study.

Although time-domain features are used in almost all of these related studies, frequency-domain features are used in a few of them [15, 20, 25, 29]. The most widely used time-domain features are the mean and the variance. Other than these, the minimum, the maximum and different percentile values are used in [15, 20, 23, 26]. On the other hand, all of the frequency-domain features are extracted from the FFT of the accelerometer signals. The raw data from which these features are extracted also vary between different studies. In some of them, features are extracted from the acceleration components on each axis, whereas others use the the magnitude of the acceleration vector as the input for feature extraction. In this thesis we used both time-domain and frequency-domain features. We evaluated the mean and the variance of the time-domain features and FFT coefficients are used as the frequency-domain features. Differently from the previous studies, we also evaluated several features which are extracted from the autocorrelation function of the accelerometer signal. In order to ensure orientation independency of the mobile phone, the magnitude of the acceleration vector is used for computing these features.

As mentioned, dealing with different sets of activities by using different types of sensors makes comparison of these works difficult. However, as a general requirement, we can say that a solution which provides real-time classification on mobile phones, power efficiency and high recognition accuracy can be considered as a successful one.

Among the studies mentioned in this chapter, there are two of them which are very close to this thesis in terms of the activities they classify and the sensors they use. Reddy *et al.* [29] and Siirtola *et al.* [23] both classifies the same five activities that are classified in this thesis which are idling, walking, running, bicycling and motorized transport. In [29], different positions for the mobile phone is studied unlike this thesis where the mobile phone is assumed to be carried in a pocket. However, they make use of an extra GPS receiver along with the accelerometer, thus the need for a clear line of sight with GPS satellites confines the usage of the solution to outdoors. Also in [29], only offline experiments are carried out and in these cross validation tests the system achieves 93% recognition accuracy with DHMM classifier and 91% accuracy with KNN when both of these classifiers are used in combination with a DT algorithm. In [23], only an accelerometer that is embedded in a mobile phone is used, the phone is assumed be placed in a pocket and both offline and online tests are carried out like in this thesis. However, Siirtola *et al.* use only time-domain features like the mean, the minimum, the maximum, the standard deviation and different percentiles. Thus, our study differs from this work in terms of the variety of features used. Besides the extra features evaluated in this thesis, the SFS and the SBS algorithms are applied in two different approaches for an efficient selection of features.

4. ACTIVITY RECOGNITION AND EFFICIENT FEATURE SELECTION

The objective of our study is to evaluate and employ an efficient feature selection that will enable classification of five different daily activities of a person in real-time with high accuracy on a mobile phone. We also aim to provide a solution that will not require any constrained mobile phone usage, so it is assumed that the phone is carried in a pocket in any orientation while the classifier is running. While determining the activities to be investigated in this thesis, we considered how often the activity is performed in daily lives of people and how much it was studied in the literature. By selecting frequently performed activities in daily life, we intended to increase the practical use of the study. Investigating widely studied activities facilitated the comparison of this work with the related ones. Accordingly, we chose to work on the activities of idling (stationary), walking, running, bicycling and motorized transport.

One of the most important restrictions for a mobile platform is the limited power. The effect of this limitation on activity recognition applications becomes critical generally due to the power consumption of sensors and the complexity of the classification algorithm. To overcome this problem, we used only the accelerometer rather than other power-hungry sensors like GPS. Besides being relatively energy efficient, accelerometers can be found in almost all of the smart phones. Additionally, we used a lightweight classifier (KNN) considering the low processing capacity on mobile devices.

Limiting our sensor options and classifier complexity to develop a solution that is suitable for mobile phones, we focused on the selection of efficient features that would help us attaining high recognition rates. Obtaining an appropriate selection of features will help us in maximizing the performance of our lightweight KNN classifier and utilizing the processing power efficiently by decreasing the number of features to be processed by the classifier. In this regard, we evaluated several widely used time and frequency-domain features along with some specific ones that are extracted from the autocorrelation function of the signal.

This chapter presents the details about the steps of our activity recognition process and the implementation of the software on mobile and desktop platforms. Also, information about offline and online tests that are performed to improve and measure the system performance is provided.

4.1. Data Collection

As mentioned, our purpose is to develop a solution that will classify a person's five daily activities (stationary, walking, running, bicycling, motorized transport) in real-time with high accuracy rate on mobile phones. The very first step of our study is the collection of accelerometer data from different subjects while they are performing the given activities.

Since the data collected in this step will shape the direction of the next steps, we worked with different subjects and paid attention to covering different ways of performing the given activities as much as possible. The only restriction in the data collection is that the subjects are supposed to carry the mobile phone that is used for sampling in the front pocket of their trousers and the subjects mostly wore tight trousers in order to sense the motion patterns more clearly.

We collected the activity data from 10 individuals with different ages between 15 and 30. They are asked to perform each of these activities for 2 minutes. This provided us 20 minutes of samples for each of the five activities and that makes a total of 100 minutes long data. Before starting the sampler, a subject provides his name and the type of activity he will perform to the mobile application and places the mobile phone in either left or right pocket of his/her trousers with any orientation. The application always uses the maximum sampling rate of the accelerometer which is 100 Hz for the data collection. The sensor readings are logged as comma separated values to a text file which is given a name that includes the names of the subject and the performed activity as in Figure 4.1.

furkan_walking.csv

```
#TIMESTAMP;ACCEL_X;ACCEL_Y;ACCEL_Z
151061633;2.75812029838562;6.742072105407715;-2.75812029838562
151061643;1.9919757843017578;7.354987621307373;-2.29843354225158 7
151061653;1.225831270217895 5;6.742072105407715;-1.685518026351928 7
151061663;0.6129156351089478;6.742072105407715;0.766144514083862 3
151061673;0.3064578175544739;7.354987621307373;2.451662540435791
151061683;-0.766144514083862 3;7.508216381072998;2.145204782485962
151061693;-0.766144514083862 3;7.048529624938965;1.8387469053268433
151061703;-0.3064578175544739;6.588842868804932;2.29843354225158 7
151061713;1.225831270217895 5;6.435614109039307;3.9839515686035156
151061723;2.29843354225158 7;7.048529624938965;0.6129156351089478
151061733;4.750096321105957;8.12113189697265 6;-8.580819129943848
```

Figure 4.1. Accelerometer sampling log file.

The activities are performed in various possible conditions. For *stationary* activity, samples are collected from subjects while they are sitting, standing or lying. The activities of *walking*, *running* and *bicycling* are performed with variable speeds and on both inclined and flat ground. Three different bicycles are used for bicycling. The subjects tried to continue pedaling as much as possible, but there were still short periods of coasting. Different possibilities are sampled for *motorized transport* as well. Samples are collected while travelling on two different busses and an automobile. All these variations in the performance of activities are supposed to make things more difficult for a classifier and one should consider this fact while evaluating the performance of the system.

The data collected in this phase is used throughout the study. We used it to carry out the offline analysis. In this series of analysis, we investigated the effect of segmentation parameters and several different features. To find out how the system performs for different feature and parameter combinations, we performed leave-one-out cross validation tests on this dataset. We also used this data to form the training datasets which are used for online tests.

4.2. Segmentation

As explained in Section 2.2.2, a continuous signal should be divided into data windows for extracting features. For this purpose, we used fixed-length sliding windows

Table 4.1. Sliding Window Configurations.

Sampling Rate	Window Length	Window Overlap
100 Hz	1350 msec	25%
100 Hz	2700 msec	63%
100 Hz	5200 msec	80%

that overlap with specific ratios. We preferred this technique since it is shown to yield good results [25, 29] although it is computationally efficient and easy to implement.

Some of the features evaluated in this study are computed using FFT. So, in order to better represent the data by avoiding zero padding, we decided to use window lengths that are powers of two. In this regard, we evaluated three different window lengths which are 1350, 2700 and 5200 milliseconds. Since we use a sampling rate of 100 Hz throughout the study, the samples these windows contain counts to numbers that are close to a power of 2 (i.e. 128, 256 and 512). Although the mobile phone we used for carrying out the experiments was stable in sampling with the given rate, we still decided to use a little bit larger windows to eliminate the effect of possible undersampling caused by the operating system or the hardware. So, 1350, 2700 and 5200 msec windows always contain 128, 256 and 512 samples respectively. We applied FFT to the first 128, 256 and 512 samples in those windows. The windows usually contained few more samples, but these extra samples are also used for feature extraction in the next data window since we used overlapping windows as explained below.

In a real-time activity recognition system, the output frequency of the classifier should be consistent with the length of the time an activity transition takes. Since most of the time, it is possible for an individual to change the activity he performs in less than one second, in this study we fixed the output frequency of the classifier to 1 Hz. To attain this classification frequency, we used appropriate window-overlap ratios for different window lengths. As a result, we used three different configurations for the sliding window algorithm as shown in Table 4.1.

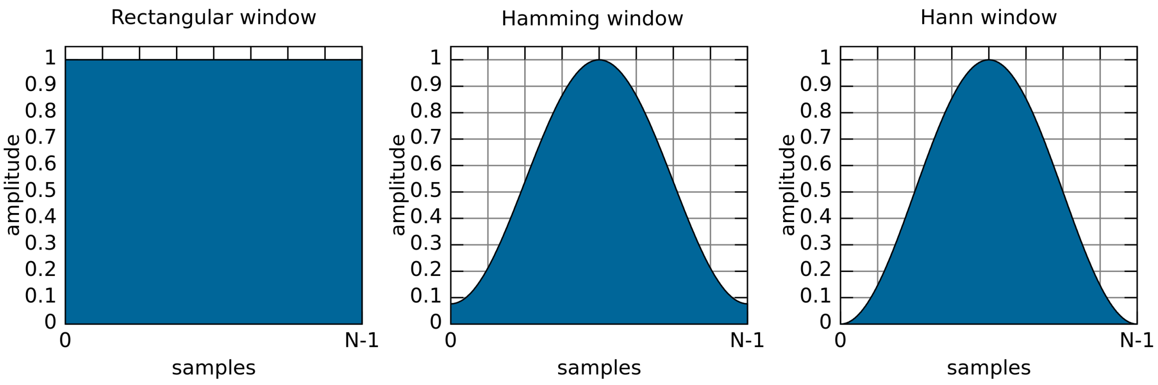


Figure 4.2. Evaluated window functions.

Since we make use of Fast Fourier Transform (FFT) for feature extraction, we also analyzed the effect of window functions on the performance of the system. Since FFT algorithm cannot know how the signal behaves outside the given data segment, it implicitly assumes that the data segment to be processed is repetitive. However, it is very likely that there are discontinuities between the ends of these data segments and this causes the signal energy to spread out over a wide frequency range. Window functions are used to avoid this spectral leakage by eliminatimg discontinuities at the ends of the data segments.

In this study we evaluated the performance of Hann and Hamming windows along with rectangular windows (Figure 4.2). Hann and Hamming are two of the most popular window functions for random signals, because they provide better frequency resolution and leakage protection while offering fair amplitude accuracy.

4.3. Feature Extraction

In order to have a system that is not affected by the position and the orientation of the mobile phone, all features are extracted from the the magnitude of the acceleration vectors as computed in Equation 4.1.

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

Although we may loose detailed information about the accelerometer readings in individual axes by using the magnitude of the vector, this enables the classification process not to be affected by the orientation of the phone. So, we do not have a constraint for the orientation of the phone in the pocket. On the other hand, a successful feature extraction task is highly dependent on the way data is separated into windows (segmentation) and the values of windowing parameters like the window size, the window overlap ratio and the windowing function. In a classification system, the effect of these factors on the overall performance has strong correlation with the characteristics of the input signal and the frequency of outputs to be produced. Inherently, human physical activities have low frequencies. For example, a one-second sampling window for the activity of walking one step per second will contain only one motion pattern and that results in a value of 1 Hz for the frequency of motion. If we use one-second windows for computing features, it is clear that we can hardly extract information about the frequency of motion, since each window will span only one period of the signal. Moreover, applying window functions (i.e. Hann, Hamming, etc.) to such windows will severely mask the signal. In Section 4.2, we provided information about the segmentation method and the windowing parameters evaluated in this study and in Chapter 5, the effect of different windowing parameters on the overall system performance is compared. The results indicate that more than 95% recognition accuracy could be attained by both 2700 and 5200 msec windows, whereas at most 88% accuracy is attained when using 1350 msec windows. Regarding window functions, rectangular windows outperformed Hann and Hamming windows when using 1350 and 2700 msec windows, however Hann windows provided slightly higher performance than rectangular windows when 5200 msec windows are used.

In this study, we extracted a total of 17 distinct features from the accelerometer data (Table 4.2) and compared them in terms of their contribution to the classification performance. These include some common features that are used in most of the activity recognition studies like the mean, the variance, 10 primary FFT coefficients, the value and order of the largest FFT coefficient. Other than these, we extracted some particular features that are extracted from the autocorrelation of the sample windows. Throughout the study, we evaluated the system performance by employing different combinations of these features.

Table 4.2. List of Features.

Domain	**Feature**	**Symbol**	**Extracted From**
Time	Mean	μ_A	Accelerometer Magnitude
Time	Variance	σ_A^2	Accelerometer Magnitude
Frequency	1^{st} FFT Coef	F_{C_1}	FFT of Accelerometer Magnitude
Frequency	2^{nd} FFT Coef	F_{C_2}	FFT of Accelerometer Magnitude
Frequency	3^{rd} FFT Coef	F_{C_3}	FFT of Accelerometer Magnitude
Frequency	4^{th} FFT Coef	F_{C_4}	FFT of Accelerometer Magnitude
Frequency	5^{th} FFT Coef	F_{C_5}	FFT of Accelerometer Magnitude
Frequency	6^{th} FFT Coef	F_{C_6}	FFT of Accelerometer Magnitude
Frequency	7^{th} FFT Coef	F_{C_7}	FFT of Accelerometer Magnitude
Frequency	8^{th} FFT Coef	F_{C_8}	FFT of Accelerometer Magnitude
Frequency	9^{th} FFT Coef	F_{C_9}	FFT of Accelerometer Magnitude
Frequency	10^{th} FFT Coef	$F_{C_{10}}$	FFT of Accelerometer Magnitude
Frequency	Max FFT Coef Value	$\max\{F_{C_i}\}$	FFT of Accelerometer Magnitude
Frequency	Max FFT Coef Index	$I_{\max\{F_{C_i}\}}$	FFT of Accelerometer Magnitude
Frequency	Maximum Value of Autocorrelation	$\max\{R_i\}$	Autocorrelation of Accelerometer Magnitude
Frequency	Zero Crossing Rate of Autocorrelation	$zcr\{R\}$	Autocorrelation of Accelerometer Magnitude
Frequency	Autocorr Peak Index	$I_{\max\{R_i\}}$	Autocorrelation of Accelerometer Magnitude

Before passing the features extracted from the raw data to a classifier as an input, we build a feature vector from the normalized values of these features. Since the KNN classifier in this study uses the Euclidean distance while detecting the nearest samples, the value ranges of different features is expected to effect the weight of their influence on the distance calculation. So, in order to equalize the ranges of the features, we normalized each feature's value by mapping it to a value between 0 and 1. In Equation 4.2, the calculation of the normalized value is shown. Here, f_{max} stands for the maximum value of that feature in the complete dataset and f_{min} stands for the minimum value of it.

$$\hat{f} = \frac{f - f_{min}}{f_{max} - f_{min}} \tag{4.2}$$

In the following sections, we explain how these features behave when samples of different activities are considered. We also give details about the features extracted from the autocorrelation function and describe why they are expected to contribute to the classification performance of our solution.

4.3.1. Common Features

As mentioned above, we can separate the features used in this study into two groups. In the first group, there are some common features that have been used in several previous studies. In this section, we present an analysis of how these common features behave for each of the activities that are to be classified by this work.

Among the features that are studied in this thesis, the mean and the variance are the most widely used features in different domains of signal processing. These features are extracted from each data window that are obtained by segmenting the continuous accelerometer readings. As mentioned above, we only use the the magnitude of the acceleration vector which is the sum of values on all three axes for extracting features, thus the average value and the variance of the accelerometer magnitude is computed for each data window to obtain these two features.

Before including the mean and the variance to our initial feature set, we analyzed the behaviour they exhibit for accelerometer signals of different activities that are targeted in this work. In Figure 4.3, the normalized values of mean and variance are plotted for 100 randomly selected data windows sampled while performing the activities of stationary, motorized transport, bicycling, walking and running. Among the segmentation configurations explained in 4.2, we used 2700 msec 63% overlapping rectangular windows for extracting the values plotted in these graphs. This segmentation configuration is one of the most appropriate ones for obtaining a good separation of

Figure 4.3. Mean and variance values for randomly selected data windows of accelerometer samples for five daily activities.

activities using these features.

The curves that show normalized mean values for activities like walking, running and bicycling appear to be distinct enough so that this feature seems to be a reliable one for differentiating samples from these three activities. However, there are still some intersection zones where the value for these activities overlap significantly. For example, there are data windows of both walking and running activities that reside on the interval [0.5-0.6]. On the other hand, it is not likely that the mean will be useful for classifying the other two: stationary and motorized transport. Also bicycling is partially separable from these two in the graph, thus it is clear that we need some other supporting features to distinguish bicycling from these activities precisely. In Figure 4.4, it can be clearly seen how the mean curves for bicycling, motorized transport and stationary are interlaced.

Looking at the graphs that show the normalized variance values of data windows

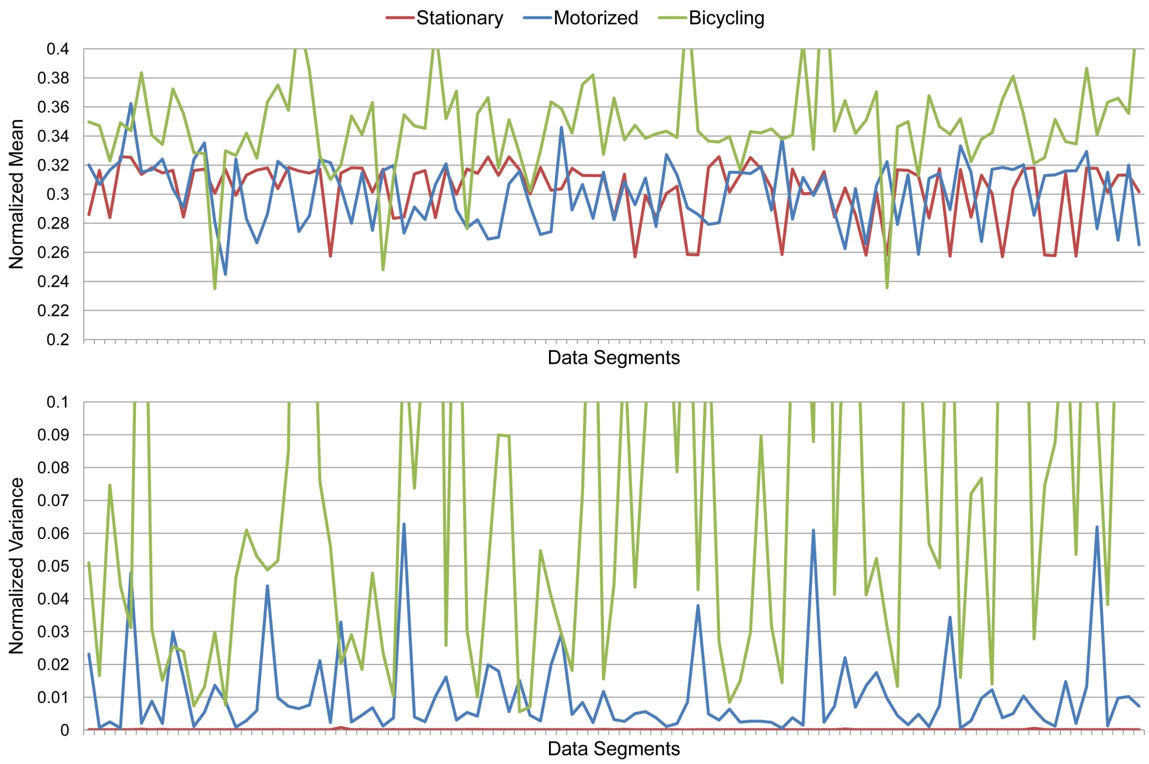


Figure 4.4. Mean and variance values for randomly selected data windows of accelerometer samples for three daily activities.

for different activities, it is possible to say that variance is expected to provide a similar level of differentiation between bicycling, walking and running activities. In the graph, there are still overlapping zones between the curves of activity pairs like running-walking and walking-bicycling. However, the variance curves of bicycling, motorized transport and stationary activities are not as mixed as in the mean curves. As seen in Figure 4.4, although they sit in a relatively narrow interval, variance curves of these activities are separated more clearly than the mean curves.

Other common features that are used in this study are extracted from the Fourier transform of the activity samples. As mentioned in Section 4.2, we used appropriate window sizes to avoid zero padding before applying FFT. Throughout the experiments carried out in this study, we used 128, 256 and 512-point FFT in accordance with the evaluated window size. A total of 12 features are extracted from the result of FFT. These are the index of the greatest FFT coefficient, the value of the greatest FFT coefficient and the value of 10 primary FFT coefficients.

To have an idea about how FFT coefficients behave for the accelerometer samples of the activities that we aim to classify in this study, the graphs in Figure 4.5 can be analyzed. Each of these five graphs shows the results for a different activity. On each graph, the normalized values of 10 primary FFT coefficients are plotted for 20 randomly selected sampling windows separately. So, there are 20 series on each graph which corresponds to those data windows.

As suggested in these figures, there are significant differences between the values of FFT coefficients that are computed from the samples of the five activities. The values for stationary and motorized transport are always very small when compared to other three activities. Also, there is no indication of a dominant FFT component in these values. Instead, the values of coefficients for different data windows exhibit an independent behaviour. On the other hand, the FFT coefficients that are extracted from activities like bicycling, walking and running have much higher maximum values and they follow more regular patterns that are independent of the data windows. This is apparent especially in the graphs of walking and running activities. For example, in the graph of walking activity, 4^{th} FFT coefficient almost always has a significantly higher values than other coefficients regardless of the data window. This is also the case for the 6^{th} FFT coefficient for the activity of running.

Obviously, these differences in the FFT features for the given activities mainly stem from the fact that some of the activities evaluated in this study are composed of periodic motions like taking steps and pedaling and the frequency of these motions differ. So, depending on these results, we can conclude that the use of FFT coefficients are very likely to contribute to the performance of the activity recognition system that we build.

4.3.2. Autocorrelation Features

In this section, we give details about the features that are extracted from the autocorrelation function and describe why they are expected to contribute to the classification performance of our solution.

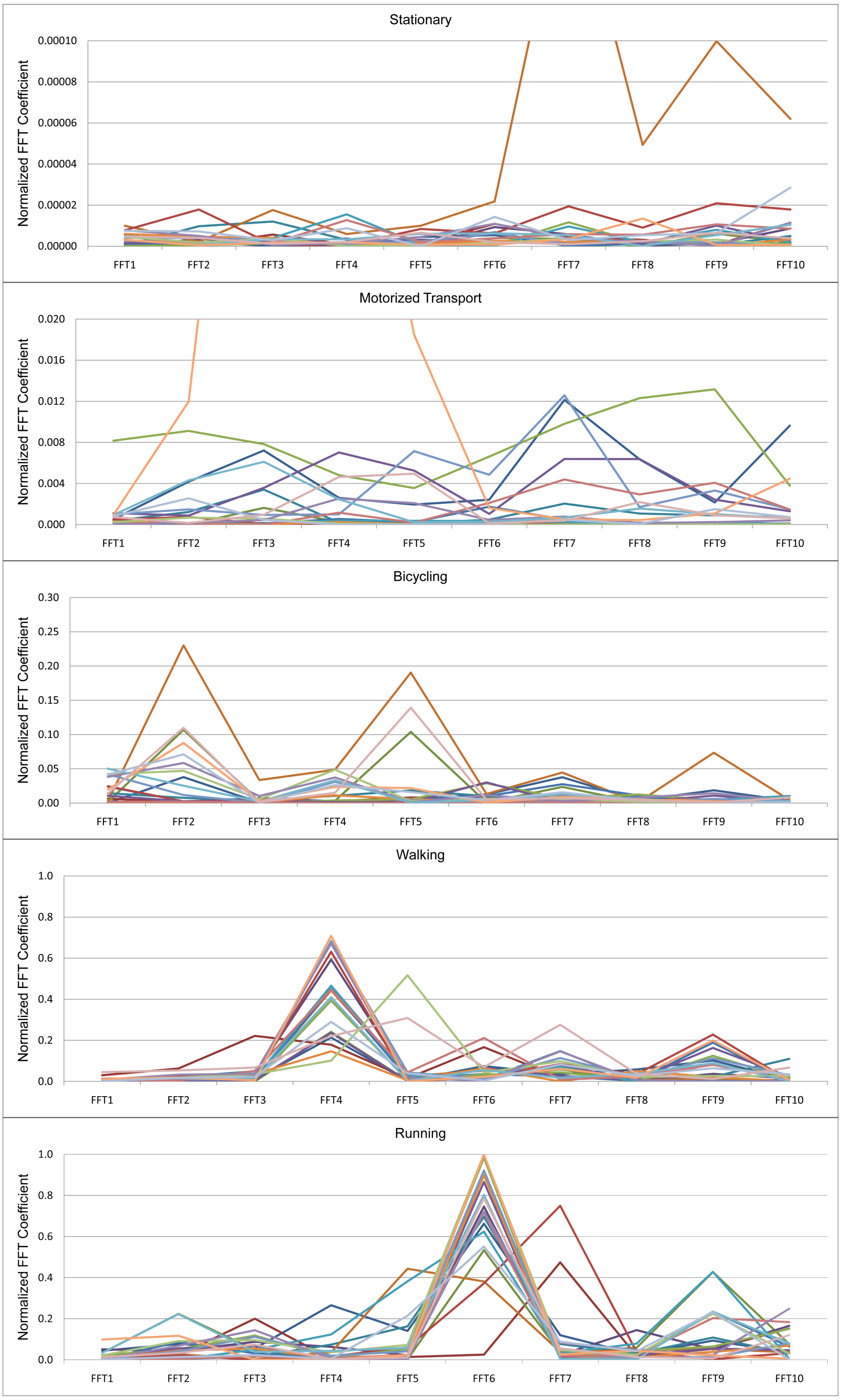


Figure 4.5. FFT Coefficients for randomly selected 20 data windows from each of the five daily activities.

In general, physical activities have two important properties which we can use for classifying them more efficiently. The first one is the periodicity of an activity. This is a feature that differentiates activities that consists of repeating motion patterns (e.g. walking) from those that are not actually a repetition of a specific motion (e.g. sitting). Furthermore, different types of periodic activities can be distinguished using a second property which is the period of motion. However, the period of an activity signal is not sufficient in classifying different periodic activities, since two different activities may have the same periods for the repeating patterns that constitute them like in fast walking and slow running. Nevertheless, this feature is still expected to contribute to the classification performance when typical speeds of activities are considered.

One can consider using FFT coefficients for estimating such frequency components of an activity signal. However, since FFT components have limited resolution in low frequencies, they will not help us in finding the period of the signal precisely. For instance, we can consider the application of a 256-point FFT to a window of 256 samples which are collected with 100 Hz. First five frequency components produced by this transform will be 0.39 Hz (100/256), 0.78 Hz (200/256), 1.17 Hz (300/256), 1.56 Hz (400/256) and 1.95 Hz (500/256). Our observations show that distinguishing walking and running activities will require more resolution for the period of signal, because we can have 1.4 Hz motion frequency in a walking signal while having 1.6 Hz in running when they are performed with specific speeds. Moreover, FFT is not suitable for estimating periodicity since the true fundamental frequency might not be the one with the largest amplitude.

In order to precisely estimate the frequency and the periodicity of a signal, we decided to use the autocorrelation function which can be computed efficiently from raw data $X(t)$ with two Fast Fourier Transforms as in Equation 4.3. The autocorrelation of a signal describes the correlation between signal values within different time frames, as a function of the time difference between these frames. In order to detect the existence of repeating patterns, one should look at the autocorrelation values which approach to 1 when the correlation between two time frames increases. The maximum value and the zero crossing rate of the autocorrelation function are the two features that we use

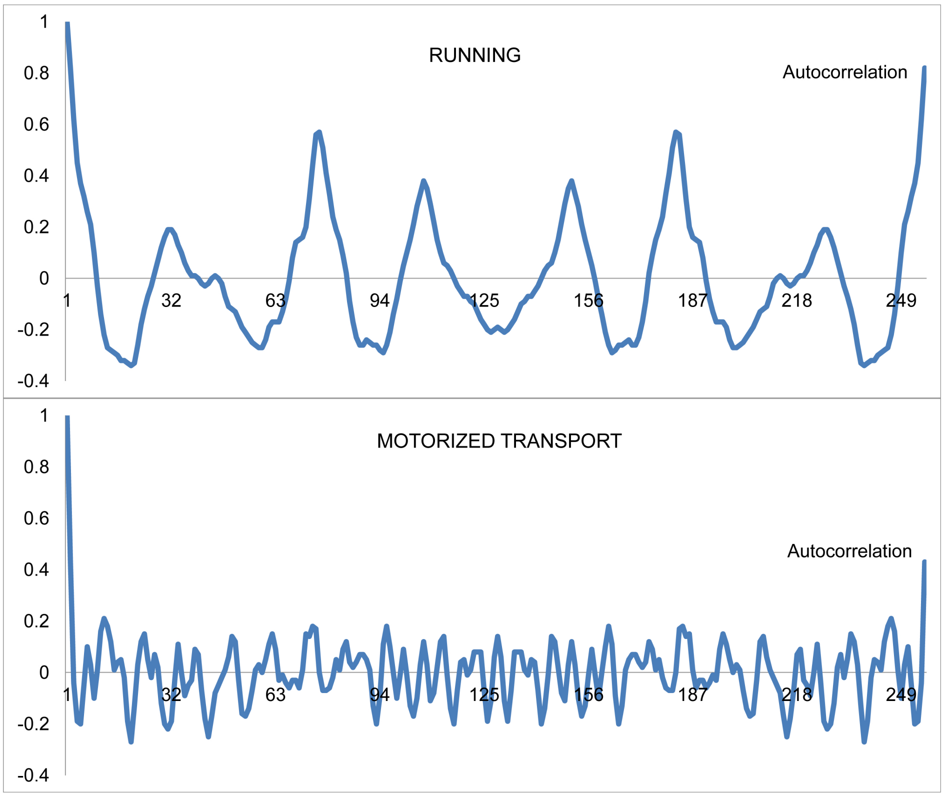


Figure 4.6. The zero crossing rate and the maximum values of autocorrelation function.

for estimating the periodicity of activities. Besides these, we use the index of peaks in the autocorrelation function as an indicator of the period of motion for periodic activities.

$$\begin{aligned} F_R(f) &= FFT[X(t)] \\ S(f) &= F_R(f){F_R}^*(f) \\ R(\tau) &= IFFT[S(f)] \end{aligned} \tag{4.3}$$

In Figure 4.6, it is seen that the maximum autocorrelation value is higher for samples collected while walking than those collected during motorized transport, since walking consists of periodic motion patterns. The two activities can also be separated by looking at the zero crossing rate of the autocorrelation function. Due to the randomness and the lack of patterns in the samples for motorized transport, the autocorrelation values exhibit an irregular course that frequently alternates between negative and positive values. Thus, the zero crossing rate has higher values for aperiodic signals.

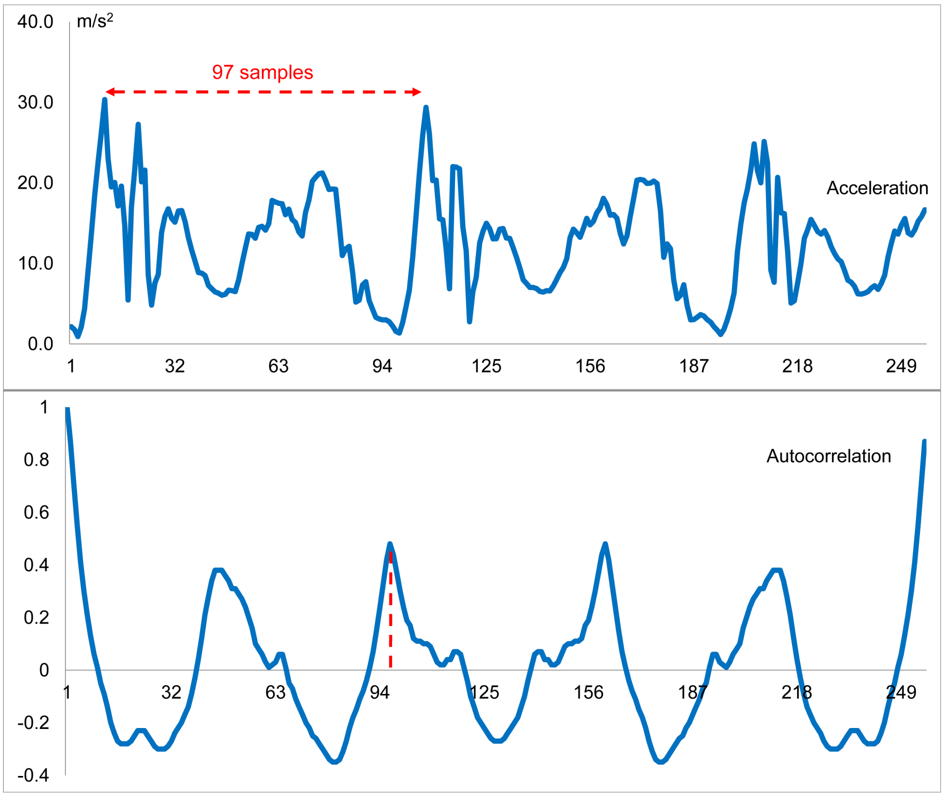


Figure 4.7. The peak of autocorrelation function and the period of motion.

Figure 4.7 illustrates the reflection of the period of a motion on the autocorrelation function. In these two diagrams, the magnitude of acceleration and its autocorrelation is plotted for a window of 256 samples which are recorded while the subject is walking. From the acceleration signal, it is clearly seen that a complete movement cycle (a left and a right step) is completed in 97 samples which correspond to 970 msec since the sampling rate is 100 Hz. When we look into the peak values in the autocorrelation diagram, we will see that the function reaches its highest value at around 97th point which point to the period of motion seen in the sample window. Note that we should ignore the higher values in the beginning of the autocorrelation function since it actually shows the autocorrelation of the sample window with itself and also the second half of it can be disregarded since the autocorrelation function is symmetric.

4.4. Feature Selection

Finding an efficient set of features that will provide high classification accuracies with a lightweight classifier on smartphones is one of the most important objectives of this study. To attain this goal, we employed two well-known feature selection algorithms, after the extraction of 17 candidate features that are mentioned above. These two methods are Sequential Forward Selection (SFS) and Sequential Backwards Selection (SBS). The reason for choosing these two algorithms for feature selection is that they are the most widely used methods in the literature besides being very easy to implement.

The SFS algorithm starts with an empty set of selected features. At each iteration, candidate features are compared according to the increase they provide in the classification accuracy and the best one is added to the selected features. The algorithm stops when no more improvement can be obtained with the remaining features. Unlike SFS, the SBS algorithm starts with a full set of selected features and removes a feature at each step by again considering the amount of improvement in the classification performance. The process stops when it is not possible to improve the performance by removing any of the features from the selection.

Since the sequential approaches like SFS and SBS do not reevaluate the usefulness of features that were selected previously, they converge to local minima. For avoiding this problem partially, we used these algorithms in two different approaches. The details about these approaches are given in Section 5.1.

All feature selection procedures are employed in the offline cross validation tests. The best feature set found during these tests is used in the online activity classification. In Chapter 5, we present the best feature sets and the recognition performance attained by each of them.

4.5. Classification

Since the main focus of this study is to evaluate the contribution of several different features to classifying different daily activities, we decided to use a single, lightweight classifier for analyzing the efficiency of the features more clearly. In this regard, we used the KNN classifier which is shown to yield high classification accuracies in several related studies [23, 25]. In the previous studies of our research group, the performances of several different classifiers are also compared for online activity recognition [26].

4.6. Software Implementation

We implemented two applications for running both offline and online tests on the performance of our activity recognition solution with different feature sets and system parameters. One of these applications is built for offline analyses that are to be run on a desktop computer and the second one is a mobile application that can be used for testing the system in real-time. Both applications are written in Java and the mobile software is implemented for Android OS. In the following subsections, we present the details about the components and the functionalities of these software.

4.6.1. Mobile Application

We implemented our activity recognition software on a Samsung Galaxy Ace S5830 smartphone with Android OS version 2.2.1. The phone is equipped with a 800 MHz CPU and a tri-axial accelerometer. The accelerometer has a measurement range of ±2g and a maximum sampling rate of 100 Hz.

The application that we developed for the Android OS contains three main *services* that implements the core activity recognition procedure. These are the *Sampler*, the *Feature Extractor* and the *Activity Recognizer*. Besides, we implemented a desktop application for carrying out offline tests. We used these three services also in the desktop application which feeds the *Sampler* service with the offline data that is to be tested unlike the mobile application in which *Sampler* registers as a sensor-event listener with the help of Android API and listens for the real-time sensor input. In the following sections, the details about these components and the user interfaces for making use of them are presented.

Table 4.3. Android sampling rate options on Samsung Galaxy Ace S5830.

Sensor Delay Parameter	**Sampling Rate**
SENSOR_DELAY_FASTEST	100 Hz
SENSOR_DELAY_GAME	16 Hz
SENSOR_DELAY_UI	8 Hz
SENSOR_DELAY_NORMAL	4 Hz

4.6.1.1. Sampler. *Sampler* service is responsible for reading accelerometer values and logging them into text files on demand. For fetching the list of sensors on a device and registering as a listener for the events of a specific sensor, Android API offers an interface called *SensorManager* [43]. Using this interface an application is able to retrieve the samples from an accelerometer sensor with four possible sampling rates which are specified by the predefined values for the delays between sensor readings. In the API these values are referred as *SENSOR_DELAY_FASTEST*, *SENSOR_DELAY_GAME*, *SENSOR_DELAY_UI* and *SENSOR_DELAY_NORMAL*. Although these values are constant, they may correspond to different sampling rates on different hardware since the sampling rate is effected also by hardware and operating system constraints.

Registering an application for accelerometer readings with these *SENSOR_DELAY* parameters on the smartphone that we used in this study provides the constant sampling rates given in Table 4.3. Among these values we used the fastest one (i.e. 100 Hz) for both offline and online tests, since a sampling rate above 30 Hz is recommended for ascertaining human movements [44].

In Figure 4.8, a user interface for the *Sampler* service is given. Via this interface the user can provide parameters like the user name (for naming the log file), the sampling rate and the data label (activity name) before starting the sampler. This interface is mainly used for collecting the activity dataset that was used in offline and online tests. On the other hand, *Sampler* is actually implemented as a standalone service to provide accelerometer readings to *Feature Extractor* and *Activity Recognizer*.

Figure 4.8. User interfaces for sampling and feature extraction.

<u>4.6.1.2. Feature Extractor.</u> This component is responsible for retrieving sensor data from the *Sampler*, segmenting this data into windows, extracting features from data windows and writing feature vectors to a log file. The detailed information about segmentation and feature extraction approaches of this study is already presented in Sections 4.2 and 4.3.

The *Feature Extractor* service takes the sensor data as the input and gives a feature vector per second as the output which is consumed by the *Activity Recognizer*. However, it is also possible to obtain Feature Extractor's output as a text file after running it with specific parameters via the user interface given in Figure 4.8.

<u>4.6.1.3. Activity Recognizer.</u> This service implements a KNN classifier which is trained with a dataset that is read from a text file containing labeled feature vectors. After the short training phase *Activity Recognizer* starts to fetch real-time feature vectors from the *Feature Extractor* service and outputs the classification result for each of them.

Activity Recognizer is designed to be trained only once, so that no additional input is needed from the user. The training data is read from a predefined file location.

Figure 4.9. User interface for the online activity recognition tests.

Figure 4.9 shows the main user interface that makes use of the *Activity Recognizer* service. This interface is used for performing online tests where the user provides which activity he will perform and the duration of the test. When the user starts the test, all three services described above are activated in sequence and by comparing the outputs of *Activity Recognizer* with the activity type entered by the user, the recognition accuracy is calculated and it is displayed to the user.

4.6.2. Desktop Application

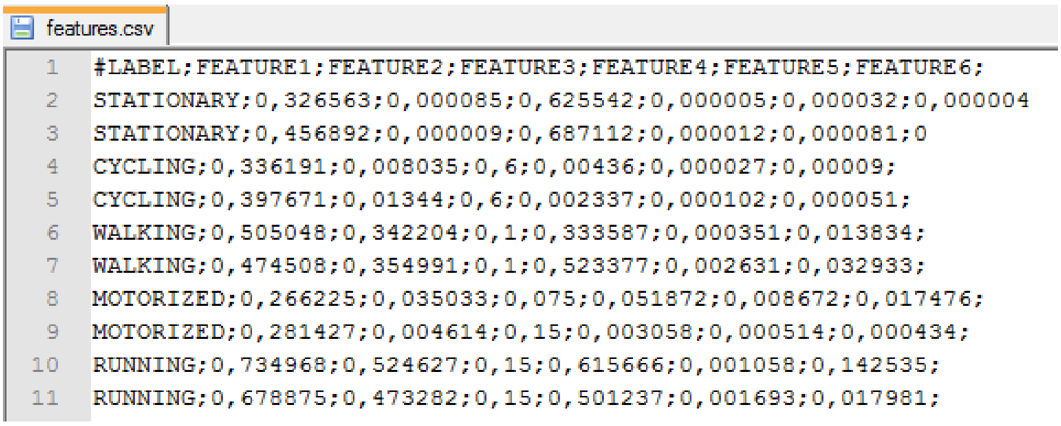
As mentioned above, we also embedded the activity recognition services that are described above into a desktop application which is written in Java. In this application, we implemented leave-one-out cross validation tests, feature selection algorithms and training dataset generators for the online classifier. The offline tests and analyses that we made with the help of this desktop application facilitated the evaluation and comparison of different features and system parameters. In the following sections, we refer to the tests that are performed with the desktop application as offline tests and to those that are performed in real-time on mobile phone as online tests.

We did not create a graphical user interface for the desktop application since it was implemented for analysis of system parameters rather than a standalone end-user application. So, we ran the application from the command line or the IDE that we used for Java development (i.e. Eclipse).

The input and output of the application are always handled using text files. The main input files are the ones that contain activity samples that are collected in the first phase of this study. As mentioned in Figure 4.1, each of these files contain data for a specific activity-subject pair and they are named accordingly. Each line contains an accelerometer reading that includes the timestamp and the acceleration values on all three axes. Since the timestamps of the accelerometer readings are available in the file, it is possible to feed an offline application by passing each reading at the corresponding time just like they are being read from a sensor.

By reading the accelerometer samples from these input files, the application can extract labeled feature vectors for each subject. Before this operation, we can set values for segmentation parameters and select the set of features that we want in the output file. This feature extraction function outputs new data files that contain the resulting feature vectors and the activity labels. The format of this output file is given in Figure 4.10. After generating these labeled feature vectors, the application is able to use them for performing cross validation tests. For example, we can write the features generated from the data collected from subject X and Y to a file and in another file we can have the features extracted from the samples collected from subject Z. Then we can employ a cross validation test by using the first file for training and the second one for testing. Since we can choose any values for windowing parameters before extracting features and we can select any set of subjects and features for both training and testing datasets, we can run cross validation tests with all possible configurations to analyze the effect of parameters and features.

The next important functionality of our desktop application is the feature selection. It employs SFS and SBS algorithms according to the performance measurements done by cross validation. For determining the best set of features for different segmentation parameters, the software implements two approaches that makes use of these


features.csv

```
#LABEL;FEATURE1;FEATURE2;FEATURE3;FEATURE4;FEATURE5;FEATURE6;
STATIONARY;0,326563;0,000085;0,625542;0,000005;0,000032;0,000004
STATIONARY;0,456892;0,000009;0,687112;0,000012;0,000081;0
CYCLING;0,336191;0,008035;0,6;0,00436;0,000027;0,00009;
CYCLING;0,397671;0,01344;0,6;0,002337;0,000102;0,000051;
WALKING;0,505048;0,342204;0,1;0,333587;0,000351;0,013834;
WALKING;0,474508;0,354991;0,1;0,523377;0,002631;0,032933;
MOTORIZED;0,266225;0,035033;0,075;0,051872;0,008672;0,017476;
MOTORIZED;0,281427;0,004614;0,15;0,003058;0,000514;0,000434;
RUNNING;0,734968;0,524627;0,15;0,615666;0,001058;0,142535;
RUNNING;0,678875;0,473282;0,15;0,501237;0,001693;0,017981;
```

Figure 4.10. Labeled feature vectors extracted for offline analysis.

algorithms. These approaches will be described in Section 5.1.

Finally, we can form online training datasets by making use of this offline application. Since it is not possible to train the classifier with a large dataset on a mobile phone in a reasonable time, we implemented an algorithm to select a representative subset of the complete activity dataset. At each turn, the algorithm selects an equal number of feature vectors for each activity from the dataset. Then, it performs a classification test where the classifier is trained with this selection and the remainder of the dataset is used for testing. In this test, a high classification accuracy indicates that the selected subset is a good representation of the complete dataset. After repeating this procedure for a sufficient number of times, the algorithm returns the best subset which can be used for training the online classifier.

5. PERFORMANCE EVALUATION

As mentioned above, this study offers an efficient selection of features that yields high recognition rates with a simple classifier like KNN. Meanwhile, we evaluate the parameters used for the segmentation task, since these strongly effect the influence of features on the recognition performance. So, while testing and evaluating the system performance, we focused on two main factors that effect the performance of the activity recognition system. These are the parameters used for dividing the input data into windows and the feature set used for classifying the activities.

For evaluating the performance of the proposed system, we performed a series of tests which can be grouped into two stages. The first stage is the longer one that involves the offline tests. In this stage, we worked on the sampling data collected from 10 different participants as mentioned in Section 4.1. Using this dataset, we applied cross validation tests for finding the best configuration for data segmentation and the best feature set to be used in classifying activities. In the second stage, we carried out real-time online tests with our final activity recognition system that is configured with the most successful setup discovered as a result of the offline tests.

In the following sections, the details of offline and online tests are presented along with the performance results obtained during these tests.

5.1. Offline Tests and Results

For measuring the performance of our system in the offline analysis, we used leave-one-out cross validation on our sampling data which is collected from 10 individuals as mentioned above. At each iteration, the system is trained with the samples of 9 subjects and tested with the samples of the remaining subject. This is repeated until samples collected from each subject is once used as test data. At the end of 10 iterations, each subject's data is once used for testing and we take the average of recognition rates obtained in these iterations as the main indicator of the system performance.

Figure 5.1. Snapshots from experiments.

As described in Chapter 4, before measuring the performance of the system at each test, we are able to tune parameters for the sliding window segmentation and select any set of features to be used in the classification. In the first part of the offline tests, we analyzed the performance of the system when we do not apply any feature selection procedure and use all 17 features at the same time. Next, we utilized SFS and SBS algorithms for each fold of the cross validation tests and aggregated the results to find out which features are selected most. Then, we tested the system with these mostly selected features. Lastly, we ran the SFS algorithm over all 10 folds of the cross validation at the same time.

We applied all these test steps with different sliding window configurations separately. As mentioned in Section 4.2, we evaluated three possible [window size, window overlap] pairs that satisfy 1 Hz output rate for the classifier (Table 4.1). We also compared the performance for three different window functions that are *Rectangular*, *Hann* and *Hamming*. So, we had nine different parameter combinations for comparison as shown in Table 5.1. In Section 4.2, information about the reasoning behind the selection of given window sizes and window overlap ratios was explained.

Table 5.1. Evaluated Windowing Configurations.

Window Function	**Window Length**	**Window Overlap**
Rectangular	1350 msec	25%
Rectangular	2700 msec	63%
Rectangular	5200 msec	80%
Hann	1350 msec	25%
Hann	2700 msec	63%
Hann	5200 msec	80%
Hamming	1350 msec	25%
Hamming	2700 msec	63%
Hamming	5200 msec	80%

5.1.1. Performance Before Employing Feature Selection

In the first part of the offline tests, we measured the system performance using all 17 candidate features. So, the feature set is fixed in this test and we repeated the test for different segmentation options listed above.

Figure 5.2 shows the recognition accuracy of the system when all features are used at the same time. As the results suggest, using rectangular windows for feature extraction yields significantly better accuracy rate than the rate achieved by Hann and Hamming windows. On the other hand, the results show that the best window length is 2700 msec which provides slightly higher accuracy than 5200 msec. As mentioned in Section 4.3, the relatively low accuracy rates for Hann and Hamming windows may possibly be due to the low frequency of activity signals which causes the signal to be distorted when multiplied with a window function when short window lengths are used. Also the complete collection of features that we used in this test may be more suitable for Rectangular windows. So, feature selection tests may shed a light on this performance difference in the following sections.

Table 5.2 shows the confusion matrix of the classification done by 2700 msec

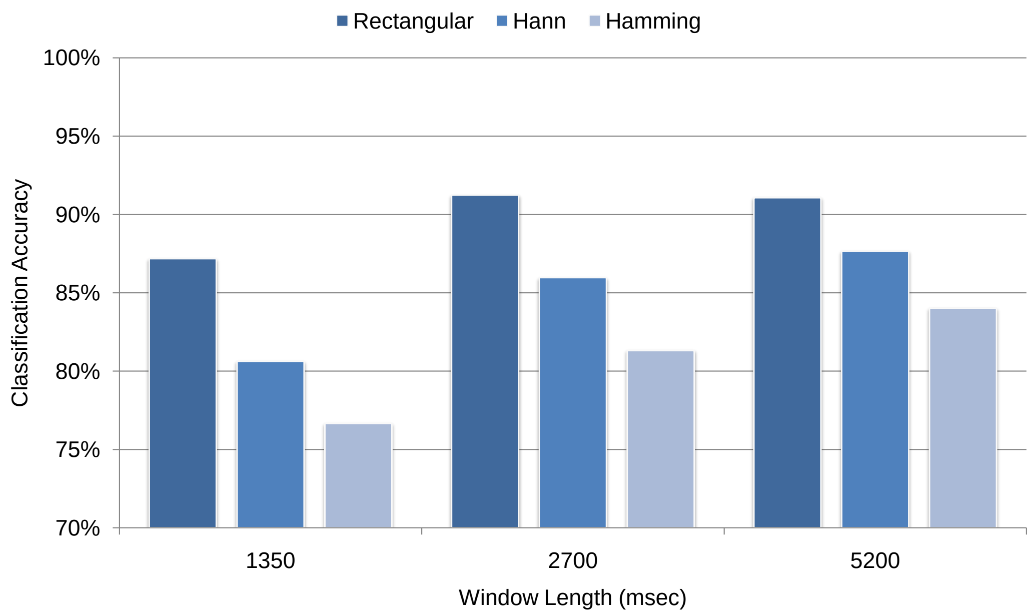


Figure 5.2. Performance of different segmentation parameters with all features.

Rectangular windows which achieved the best results in this test. As the results show, the average recognition accuracy is 91.22% which means that we already have a good starting point and that will surely make it more difficult to further improve the system performance. However, we still managed to improve the recognition accuracy significantly as we present in the following sections. Besides improving the system performance, feature selection algorithms speed up the application execution by decreasing the number of features to be computed.

With a first look into the differentiation of activities in the given confusion matrix, we can clearly see that the system mostly fails in classifying *bicycling* and *motorized transport* activities. Although, these two activities produce signals with close variances, the periodic signal of bicycling activity should help in differentiating them. Also, the bicycling samples that are collected while the subject is coasting rather than pedaling may cause a confusion with the samples collected during motorized transport. The other significant confusion is between *stationary* samples and *motorized transport* samples. The signals for these activities are sometimes very similar especially when the motor vehicle is stuck in a traffic jam. Since feature selection algorithms are not applied in these tests, it is possible that some features suppress the influence of others that can be more helpful when used together with a better subset of features. This

Table 5.2. Confusion matrix for leave-one-out cross validation test with complete set of features that are extracted from 2700 msec rectangular windows.

		CLASSIFICATION					ACCURACY	AVERAGE
		Stationary	Motorized	Bicycling	Walking	Running		
GROUND TRUTH	Stationary	942	53	3	0	0	94.39%	91.22%
	Motorized	67	846	89	0	0	84.43%	
	Bicycling	1	163	812	20	0	81.53%	
	Walking	0	1	24	966	2	97.28%	
	Running	0	0	1	14	979	98.49%	

will be investigated in Section 5.1.2 and 5.1.3.

Before going into feature selection analysis, we also tested the performance of the time-domain features that are used in this study. Since there are related studies that attain good recognition rates by using only time-domain features [21,23,26], evaluating the performance of these features will enable us to see how frequency-domain features contributes to overall system performance. Although we used only two time-domain features (mean and variance) they provided pretty good results in cross validation tests. We attained an average recognition rate of 78% when both variance and mean are used, 67% when only mean is used. The maximum performance is achieved when we used only variance and that provided 88% average accuracy.

The confusion matrix of the tests that are performed by using only variance is given in Table 5.3. The results are consistent with the variance graphs that are presented in Figure 4.3 and 4.4. As can be seen in these graphs, there are intersecting parts mostly between the activities of *motorized transport* and *bicycling*. This fact is reflected to the confusion matrix in which a relatively higher confusion is seen between these two activities. The second mostly confused activities are *bicycling* and *walking*. In the mentioned graphs, there are obvious intersection zones between these two activities as well. On the other hand, the activities like *running* and *stationary* are separated more successfully in both the confusion matrix and the graphs.

Table 5.3. Confusion matrix for leave-one-out cross validation test with only variance feature that is extracted from 2700 msec rectangular windows.

		CLASSIFICATION					ACCURACY	AVERAGE
		Stationary	**Motorized**	**Bicycling**	**Walking**	**Running**		
GROUND TRUTH	**Stationary**	972	26	0	0	0	97.39%	88.32%
	Motorized	42	827	133	0	0	82.53%	
	Bicycling	0	99	817	80	0	82.03%	
	Walking	0	0	81	847	65	85.30%	
	Running	0	0	0	56	938	94.37%	

As a result of the two tests presented in this section, we can say that the time-domain features achieve pretty good performance (88%) in classifying the given activities. However, the frequency-domain features that are evaluated in this study are very likely to contribute much to this performance since a higher recognition accuracy (91%) is obtained even without employing any feature selection algorithms.

5.1.2. Personalized Feature Selection Approach

In this study, we evaluated two different methods of applying feature selection algorithms for determining the best set of features for our purposes: personalized and generalized approach. In this section, we evaluated the performance of the system with our first approach. In this approach, we applied SFS and SBS methods for each fold of the leave-one-out cross validation tests. Since we test the system with a different subject's samples at each fold, we get different feature selections for recognizing the activity data of each subject. At the end of this test, we evaluated the performance of the system by using first N mostly-selected features for all N between 1 and 17 (total number of features) and determined the best one among these features.

The feature selection algorithms are applied when testing a different subject's data at each iteration of these tests. This is why we named this feature selection approach as *Personalized Feature Selection*. Since we aggregate the results at the end

Table 5.4. Selection rate of features with 5200 msec Rectangular windows and the highest recognition accuracies with mostly-selected features.

Feature	**Selection Rate**			**Highest Recognition Accuracies**
	SFS	**SBS**	**Overall**	
$\boldsymbol{\sigma_A^2}$	10/10	10/10	20/20	
$\boldsymbol{F_{C_2}}$	4/10	10/10	14/20	91.47%
$\boldsymbol{F_{C_3}}$	4/10	10/10	14/20	92.95%
$\boldsymbol{F_{C_8}}$	3/10	10/10	13/20	94.10%
$\boldsymbol{F_{C_6}}$	4/10	7/10	11/20	95.67%
$\boldsymbol{F_{C_9}}$	3/10	7/10	10/20	**95.76%**
μ_A	2/10	8/10	10/10	85.07%
F_{C_4}	1/10	9/10	10/20	
F_{C_5}	2/10	7/10	9/20	
$\max\{R_i\}$	1/10	8/10	9/20	
F_{C_7}	0/10	9/10	9/20	
$\max\{F_{C_i}\}$	1/10	7/10	8/20	
$F_{C_{10}}$	1/10	7/10	8/20	
F_{C_1}	0/10	8/10	8/20	
$I_{\max\{F_{C_i}\}}$	3/10	4/10	7/20	
$zcr\{R\}$	0/10	7/10	7/20	
$I_{\max\{R_i\}}$	1/10	4/10	5/20	

of the tests and reach to an efficient collection of mostly-selected features, this technique actually outputs results that are shown to yield high recognition rates for all subjects.

The results for 5200 msec rectangular windows are presented in Table 5.4. This segmentation setup provided one of the highest recognition accuracies (95.76%) using only six features. The table shows the number of times each feature is selected by feature selection algorithms and the highest accuracy rates attained by the first N mostly-selected features where N is between 1 and 17. The detailed results of the tests for each segmentation configuration (Table 5.1) are presented in Appendix A.

Table 5.5. The best feature sets composed of mostly-selected features and the recognition accuracies they yield.

1350 msec			**2700 msec**			**5200 msec**		
Rectangular	**Hann**	**Hamming**	**Rectangular**	**Hann**	**Hamming**	**Rectangular**	**Hann**	**Hamming**
σ_A^2	F_{C_1}	μ_A	σ_A^2	F_{C_1}	$\max\{F_{C_i}\}$	σ_A^2	$\max\{F_{C_i}\}$	$\max\{F_{C_i}\}$
$F_{C_{10}}$	$\max\{F_{C_i}\}$	$\max\{F_{C_i}\}$	F_{C_1}	F_{C_4}	μ_A	F_{C_2}	F_{C_5}	F_{C_3}
F_{C_9}	μ_A	F_{C_1}	F_{C_4}	F_{C_3}	F_{C_4}	F_{C_3}	F_{C_3}	F_{C_2}
$zcr\{R\}$	$F_{C_{10}}$	$F_{C_{10}}$	F_{C_9}	F_{C_6}	F_{C_1}	F_{C_8}	F_{C_8}	$zcr\{R\}$
F_{C_3}	F_{C_7}	F_{C_5}	F_{C_5}	$\max\{F_{C_i}\}$	F_{C_6}	F_{C_6}	F_{C_2}	F_{C_5}
	$I_{\max\{F_{C_i}\}}$	F_{C_9}	F_{C_6}	μ_A	F_{C_3}	F_{C_9}	F_{C_6}	F_{C_9}
	F_{C_3}	F_{C_2}	F_{C_7}	F_{C_2}	$\max\{R_i\}$		F_{C_4}	F_{C_6}
	F_{C_8}	F_{C_4}	$F_{C_{10}}$	F_{C_7}	F_{C_2}		F_{C_9}	$F_{C_{10}}$
	F_{C_9}	F_{C_6}	F_{C_2}	F_{C_8}	F_{C_7}		μ_A	$I_{\max\{R_i\}}$
	F_{C_2}	F_{C_8}	F_{C_3}	F_{C_5}	F_{C_5}			μ_A
	F_{C_6}	F_{C_7}						
	F_{C_4}							
88.06%	**86.67%**	**84.14%**	**95.26%**	**94.71%**	**91.97%**	**95.76%**	**95.76%**	**93.66%**

In order to interpret the results of these tests, we can take a look at Table 5.5 which shows the feature sets that yield the highest recognition performance for each segmentation scenario.

The results clearly show that segmenting the data into 1350 msec windows always yields relatively lower recognition performances regardless of the windowing function applied to the data segments. Using the mostly-selected features, we could attain at most 88% classification accuracy when 1350 msec windows are used. On the other hand we obtained more than 95% accuracy when we used specific subsets of mostly-selected features for 2700 and 5200 msec windows. This clearly indicates that a window length of 1350 msec is not sufficient for classifying the five daily activities that are evaluated in this study. This result is expected since we are classifying activities like walking which can be performed so slowly that 1.35 second interval may span less than two steps and this prevents us from detecting the frequency of motion in such a short window.

When the results for 2700 and 5200 msec windows are compared, it is seen that the maximum classification accuracies that could be attained with rectangular windows are very close and above 95% for both window sizes. So, the maximum recognition accuracy obtained in the tests does not show any significant increase while switching from 2700 to 5200 msec windows. This indicates that both of these window lengths are long enough for classifying the activities when rectangular windows are used. So, it is clear that the most preferable window length is 2700 msec since it requires lower computational complexity than 5200 msec while providing equally high recognition accuracies. The lower computational requirement of this window length becomes especially important when we deploy the solution to the target mobile platform.

Regarding window functions, it is seen that the use of *Rectangular* windows always yield the maximum accuracy when compared to Hann and Hamming windows which could give competitive performance only for 5200 msec windows. The main reason behind these result seems to be the low frequency of accelerometer signals for physical activities. For example, a 1350 msec window for the activity of walking one step per second will contain less than two motion patterns. Thus, if we use this window length for computing features, it will not be possible to extract a reliable information about the frequency of motion. Moreover, applying window functions (i.e. Hann, Hamming) to such short windows will further mask the signal. This is why the difference between the performance attained with rectangular windows and the one attained with Hann or Hamming windows decreases when the window size is increased.

Since Hann and Hamming windows could not yield any significant improvement even when 5200 msec windows are used, depending on the results of our tests, it is not possible to claim that these window functions contribute to recognition performance in the scope of this study. As a result, the use of *Rectangular* windows seems to be the most appropriate option since other two window functions do not exhibit any significant superiority and on top of that they require extra processing.

Another important point in these results is that the type of features that are frequently selected by SFS and SBS algorithms slightly differs by the window function used for segmentation. Although the features that are related to FFT coefficients are always among the preferred features for all window functions, the *variance* which is the mostly selected feature when rectangular windows are used is never contained in the best feature selections for Hann or Hamming windows. This is possibly due to the fact that both Hann and Hamming functions decrease the variance of the sample window by attenuating the signal at the window edges, so that classifying the activities by looking at the signal variance becomes less efficient. The results show that, when Hann or Hamming window is used, the *mean* takes the place of the *variance* in the selected feature lists. Since window functions are not expected to effect the *mean* values as much as they effect the *variance*, this substitution in time-domain feaures is reasonable.

5.1.3. Generalized Feature Selection Approach

In our second approach, we applied the same feature selection algorithms considering the average result of a complete cross validation test. In other words, at each iteration of SFS and SBS algorithms, we tested the performance of the current feature set against the whole 10-fold cross validation test which gives an average accuracy for all subjects. Since the whole cross validation test is repeated plenty of times during the test, this second test takes much more time than the previous one.

As in the previous section, we again performed the test for each segmentation configuration (Table 5.1). After testing the system with one of these configurations, we obtain a single feature set which yields the maximum average recognition in leave-one-out cross validation. Since the feature selection performed by the SFS algorithm almost always outperformed SBS in these tests, here we present the results of SFS only.

The best feature sets found by the SFS method for each segmentation configuration and the average recognition rates they achieve are given in Table 5.6. Comparing these results with the results attained in the previous feature selection approach (Table 5.5), it is seen that for some of the configurations SFS converges to a local minimum which we could avoid in the previous section with the help of our first approach. On the other hand, for some other configurations this generalized approach gave us better results than we could reach in the previous section. For example, in our *personalized feature selection* approach, the best feature set discovered for 5200 msec Rectangular windows yielded 95.76% recognition accuracy whereas the best feature set found in this second *generalized* approach could yield 94.38%. However, the selection of features made by these two approaches always gave us very close accuracy rates which differ by at most 1.5%.

Table 5.6. The feature sets selected by SFS in generalized feature selection and the average recognition accuracies they yield.

1350 msec			**2700 msec**			**5200 msec**		
Rectangular	**Hann**	**Hamming**	**Rectangular**	**Hann**	**Hamming**	**Rectangular**	**Hann**	**Hamming**
σ_A^2	$\max\{F_{C_i}\}$	$\max\{F_{C_i}\}$	σ_A^2	$\max\{F_{C_i}\}$	$\max\{F_{C_i}\}$	σ_A^2	$\max\{F_{C_i}\}$	$\max\{F_{C_i}\}$
$F_{C_{10}}$	μ_A	μ_A	F_{C_4}	F_{C_4}	F_{C_4}	F_{C_3}	F_{C_9}	F_{C_9}
$\max\{F_{C_i}\}$	F_{C_8}	F_{C_1}	F_{C_1}	F_{C_1}	μ_A	$I_{\max\{R_i\}}$	F_{C_6}	$F_{C_{10}}$
F_{C_8}	$I_{\max\{F_{C_i}\}}$	F_{C_2}	F_{C_5}	μ_A	F_{C_1}	F_{C_2}	μ_A	μ_A
	F_{C_1}	F_{C_3}	$F_{C_{10}}$	F_{C_2}	$\max\{R_i\}$	F_{C_6}	F_{C_2}	F_{C_4}
	$F_{C_{10}}$	F_{C_8}	F_{C_2}	F_{C_3}	F_{C_6}	F_{C_1}	F_{C_3}	F_{C_5}
	F_{C_7}	$F_{C_{10}}$	F_{C_6}	$\max\{R_i\}$	F_{C_5}		F_{C_8}	$I_{\max\{R_i\}}$
	F_{C_3}	F_{C_6}	F_{C_7}	F_{C_7}	F_{C_3}		$F_{C_{10}}$	F_{C_6}
	F_{C_4}	F_{C_7}	F_{C_3}	$F_{C_{10}}$	F_{C_2}			F_{C_3}
		$I_{\max\{R_i\}}$	F_{C_9}					F_{C_1}
88.62%	**87.32%**	**84.81%**	**95.26%**	**94.77%**	**92.26%**	**94.38%**	**95.93%**	**94.34%**

In Table 5.7, we aggregated the results of our two feature selection approaches that are described above. In this table, the best feature set that achieved the highest classification accuracy in offline tests is given for each different segmentation setup. We also plotted the results in Figure 5.3.

When we look at these final results, we can see that the deductions that we made in the previous section still holds. Regarding the recognition accuracies attained by different window lengths, the unsuitability of short window lengths like 1350 msec for the purposes of this work is again obvious. As mentioned before, this is due to the fact that the frequency of physical motions is inherently low, thus the usage of short windows makes it hard to extract frequency related information.

On the other hand, the sufficiency of a window length of 2700 msec depends on the window function. Although there is no significant difference between the performances attained by 2700 and 5200 windows when rectangular windows are used, the recognition accuracy exhibits a regular increase proportional to window length when we use Hann or Hamming windows. As mentioned in the previous section, the reason behind this behaviour is that applying Hann and Hamming window functions to a short window is expected to mask the signal, since the motion signals analyzed in this study have very low frequency and the windows sometimes contain only a few repeating patterns (i.e. wavelengths).

The performance of Hann windows could reach the performance of rectangular windows only when 5200 msec windows are used, however both Hann and Hamming windows could not yield any significant improvement when compared with rectangular windows. Since 5200 msec Hann windows outperform rectangular windows by only 0.17%, it is not possible to claim that these window functions clearly contribute to the recognition performance in the scope of this study. As a result, the use of rectangular windows seems to be the most appropriate option since other two window functions do not exhibit any significant superiority and on top of that they require extra processing.

Table 5.7. The best feature sets that are found in offline tests.

1350 msec			**2700 msec**			**5200 msec**		
Rectangular	**Hann**	**Hamming**	**Rectangular**	**Hann**	**Hamming**	**Rectangular**	**Hann**	**Hamming**
σ_A^2	$\max\{F_{C_i}\}$	$\max\{F_{C_i}\}$	σ_A^2	$\max\{F_{C_i}\}$	$\max\{F_{C_i}\}$	σ_A^2	$\max\{F_{C_i}\}$	$\max\{F_{C_i}\}$
$F_{C_{10}}$	μ_A	μ_A	F_{C_4}	F_{C_4}	F_{C_4}	F_{C_2}	F_{C_9}	F_{C_9}
$\max\{F_{C_i}\}$	F_{C_8}	F_{C_1}	F_{C_1}	F_{C_1}	μ_A	F_{C_3}	F_{C_6}	$F_{C_{10}}$
F_{C_8}	$I_{\max\{F_{C_i}\}}$	F_{C_2}	F_{C_5}	μ_A	F_{C_1}	F_{C_8}	μ_A	μ_A
	F_{C_1}	F_{C_3}	$F_{C_{10}}$	F_{C_2}	$\max\{R_i\}$	F_{C_6}	F_{C_2}	F_{C_4}
	$F_{C_{10}}$	F_{C_8}	F_{C_2}	F_{C_3}	F_{C_6}	F_{C_9}	F_{C_3}	F_{C_5}
	F_{C_7}	$F_{C_{10}}$	F_{C_6}	$\max\{R_i\}$	F_{C_5}		F_{C_8}	$I_{\max\{R_i\}}$
	F_{C_3}	F_{C_6}	F_{C_7}	F_{C_7}	F_{C_3}		$F_{C_{10}}$	F_{C_6}
	F_{C_4}	F_{C_7}	F_{C_3}	$F_{C_{10}}$	F_{C_2}			F_{C_3}
		$I_{\max\{R_i\}}$	F_{C_9}					F_{C_1}
88.62%	**87.32%**	**84.81%**	**95.26%**	**94.77%**	**92.26%**	**95.76%**	**95.93%**	**94.34%**

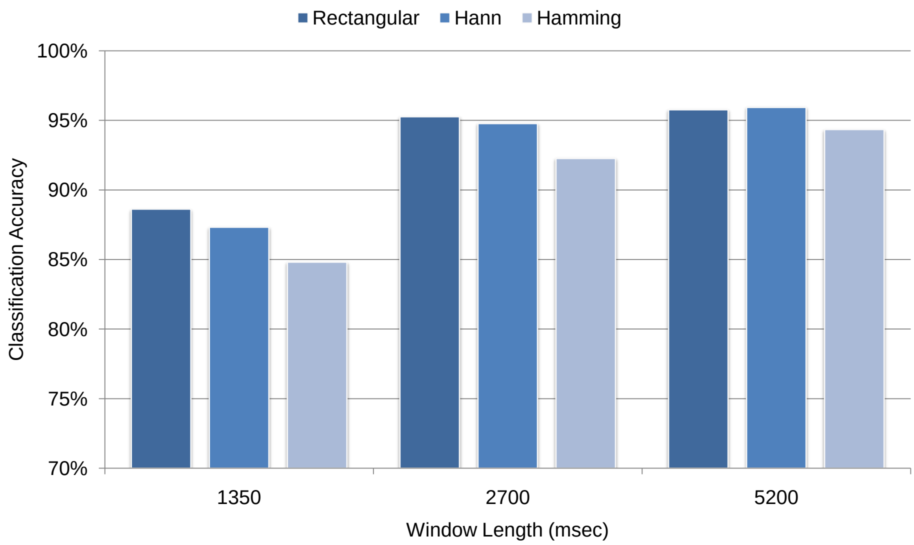


Figure 5.3. Best recognition rates attained by different segmentation parameters in offline tests with feature selection.

5.1.4. Training Dataset for Online Classifier

As a result of the offline tests that are described in Section 5.1.3 in detail, it is shown that segmenting the input signal into 2700 and 5200 msec rectangular windows and 5200 msec Hann windows yield the highest recognition accuracies which are very close (95.26%, 95.76% and 95.93% respectively). However, 2700 msec windows require 256-point FFT computation whereas 5200 msec windows require 512-point FFT which has higher computational complexity. Since our target platforms are mobile phones with limited resources, we decided to use 2700 msec rectangular windows at the cost of the negligible performance difference that is shown by the tests above. The best feature set that is found to yield the highest performance consists of the features σ_A^2, F_{C_4}, F_{C_1}, F_{C_5}, $F_{C_{10}}$, F_{C_2}, F_{C_6}, F_{C_7}, F_{C_3} and F_{C_9}. In Table 5.8, the confusion matrix of the leave-one-out cross validation that is carried out using this segmentation configuration and feature set is given.

Compared to the related studies, especially those considering the same set of activities [23, 29] which report 92% and 95% accuracy rates, the results obtained in this study are similar where we experiment with more users and select the features that maximize the recognition rate.

Table 5.8. The confusion matrix of leave-one-out cross validation obtained by using the best feature set determined for 2700 msec Rectangular windows.

		CLASSIFICATION					**ACCURACY**	**AVERAGE**
		Stationary	**Motorized**	**Bicycling**	**Walking**	**Running**		
GROUND TRUTH	**Stationary**	978	20	0	0	0	98.00%	95.26%
	Motorized	42	912	48	0	0	91.02%	
	Bicycling	0	65	910	20	1	91.37%	
	Walking	0	0	17	973	3	97.99%	
	Running	0	0	1	20	973	97.89%	

The confusion matrix shows that the misclassifications are made mostly between the activity pairs like *motorized transport-bicycling* and *stationary-motorized transport*. This is very similar to the confusion made by the system when testing it without feature selection (Table 5.2). As mentioned in Section 5.1.1, the confusion between *motorized transport* and *stationary* activities is possibly due to the sampling periods when the motor vehicle is stuck in traffic and the confusion between *bicycling* and *motorized transport* may be caused by the coasting on bicycle rather than pedaling.

The final step of our performance evaluation involves performing online, real-time tests after tuning the system according to the feature set and the segmentation setup that are selected according to the results achieved in offline tests. Before deploying the configured application to a mobile platform, we should form a training dataset for the online version of the system. Since the size of the training dataset directly effects the memory and CPU usage of the KNN classification algorithm, a reasonable dataset size should be chosen for an efficient consumption of the mobile phone resources so that the application leaves enough space for other important tasks running on the device like phone calls and internet browsing. In order to determine the right dataset size, we observed the resource utilization of our application by using training datasets with different sizes. In Table 5.9, the average memory and CPU utilization of the application for different sizes of training dataset is presented. In this table, the first column shows the number of feature vectors contained in the training dataset. In order

Table 5.9. The effect of training dataset size on the resource utilization.

Dataset Size	Average Memory Usage	Average CPU Usage
5000	12.3 MB	28.6%
4000	11.1 MB	20.4%
3000	10.2 MB	18.2%
2000	9.2 MB	12.9%
1000	**8.3 MB**	**9.8%**
500	7.8 MB	8.2%

to keep the CPU usage of our solution below 10%, we preferred using the dataset size of 1000 which is the largest dataset size that keeps the CPU usage below this level. This dataset corresponds to $\frac{1}{6}$ of the total data that we collected from participants.

The training dataset is extracted from the set of labeled features that are computed from the samples which we used in offline tests. For this purpose, we employed an algorithm which helped us in selecting an appropriate subset which is a representative one for the whole dataset. As illustrated in Figure 5.4, the algorithm runs classification tests in which the classifier is trained with the subset that is selected for the online version, and the remainder of the dataset is used for testing. After repeating this test with a sufficient number of randomly selected subsets, the one that gave the best results is deployed to the smartphone which is used for online tests. In this phase, we selected 50 as the number of randomly selected datasets which is symbolized with n in the algorithm. In order to see if 50 is a sufficient number for this purpose or not, we tried generating 75 or 100 datasets as well. As a result, it is found that generating more than 50 random datasets does not provide any significant contribution. For example, the performance difference between the best one found in 100 randomly selected datasets and the best one in 50 datasets never exceeded 0.1%.

Using this method, we selected 10 sets of labeled features as training datasets which are represented by T_i in the algorithm. While creating each one of these training datasets we excluded the samples of a different subject (d_i). Using these 10 training

```
Let D be the set of all labeled features
Let n_s be the number of subjects (10)
T := An empty list of training datasets
n_t := Training dataset size

for i = 1 to n_s do
  d_i := data collected from i^th subject
  D_i := D − d_i
  R_max := 0
  for j = 1 to n do
    S_j := Select n_t feature vectors from D_i randomly
    Train KNN classifier with S_j
    Accuracy := Test the classifier D_i − S_j
    if  Accuracy > R_max then
      R_max := Accuracy
      T_i := S_j
    end if
  end for
end for
```

Figure 5.4. The algorithm for extracting training datasets for online tests.

datasets, we could perform online test for each of the same 10 subjects by training the system with a data set that do not contain the subject's previous samples. So, each training dataset is prepared for a different subject and while performing an online test we trained the system with the corresponding dataset for that subject. In Section 5.2, we will present the results of the online tests that are carried out by deploying these training datasets to the mobile phone.

5.2. Online Tests and Results

As the last stage of the performance evaluation, we performed online tests using the set of features that are selected as the result of the offline tests (Table 5.8). Online tests are performed by the same 10 participants, however for each subject, we used a different training dataset which is prepared using the samples of 9 other subjects that were collected in the beginning of performance evaluation part.

This time participants are asked to perform each activity for 1 minute. Before a subject starts performing an activity, the application on the phone is informed about the type and the duration of that activity and it is placed in either left or right pocket of the subject without any orientation restriction. When the activity duration ends, the application displays the classification accuracy on the screen (Figure 4.9).

Performing the test with 10 participants and five activities, we carried out a total of 50 test sessions. Summing up the confusion matrices of these sessions, we obtained the matrix given in Table 5.10. The average recognition accuracy achieved in online tests is nearly 93% which is very close to the results obtained in offline tests. This result suggests that the online classifier is modeled successfully on the desktop application and the achieved performance is really a competitive one when compared with the related studies.

Table 5.10. Confusion matrix of online tests.

		CLASSIFICATION					**ACCURACY**	**AVERAGE**
		Stationary	**Motorized**	**Bicycling**	**Walking**	**Running**		
GROUND TRUTH	**Stationary**	568	28	4	0	0	94.67%	92.67%
	Motorized	5	578	17	0	0	96.33%	
	Bicycling	0	72	514	12	2	85.67%	
	Walking	0	1	32	567	0	94.50%	
	Running	0	0	5	42	553	92.17%	

Table 5.11. Confusion matrix of the longer online tests.

		CLASSIFICATION					ACCURACY	AVERAGE
		Stationary	Motorized	Bicycling	Walking	Running		
GROUND TRUTH	Stationary	839	52	9	0	0	93.22%	91.98%
	Motorized	67	791	40	2	0	87.89%	
	Bicycling	0	48	823	26	3	91.44%	
	Walking	0	2	38	857	3	95.22%	
	Running	0	0	4	67	829	92.11%	

The confusion matrix of online tests is not so different from the matrix in Table 5.8 which presents the results of the offline tests. Again there is a significant confusion between *bicycling* and *motorized transport* activities which is evaluated in the previous sections. However, the confusion between *motorized transport* and *stationary* activities that we experienced in offline tests is not seen in online test results. The reason behind this difference should be the fact that during online tests for the activity of motorized transport, the vehicle has never stopped. Since each online test session was only one minute long, it was easier to keep moving for such a short time before the motor vehicle is stopped by something like red light, traffic or a pedestrian crossing the street. In order to verify this, we carried out another online test in which 3 subjects performed each activity for 5 minutes. The confusion matrix of this test is given in Table 5.11.

The most significant difference between the results of these two tests is decrease in recognition rate of the activity of *motorized transport*. As mentioned, as long as the motorized vehicle keeps moving the activity is recognized with high accuracy, however if there are intervals during which vehicle stops moving for some reason (e.g. traffic, red light) then the accuracy decreases and the activity is confused with *stationary*. In the second test, such intervals were experienced more than in the first one since the duration of the test is longer and it was inevitable to stop moving from time to time. Moreover, among the subjects that are participated in the second test, one of them performed the activity of motorized transport by traveling on a bus which frequently stops at bus stops.

In order to filter out such short periods of confusion during a continuous performance of an activity, a long-term intelligence can be implemented on a higher level. Analyzing the outputs of the classifier over a long period like one minute, it can be determined whether there is a dominant activity throughout this period. If it is likely that the same activity is performed during this period, then the individual misclassifications or the short-term confusions can be eliminated. Although a higher level approach like this may increase overall classification rate remarkably for similar systems, we could not employ such techniques in this study since one of our objectives was to give real-time output.

On the other hand, the recognition rates for the activities of *stationary*, *walking* and *running* are very close in both tests. However, the recognition rate of *bicycling* is increased in the second test. The recognition accuracy for *bicycling* is mostly effected by the confusion of with *motorized transport*. As mentioned, the confusion of *bicycling* with *motorized transport* is very likely to be caused by the periods of coasting on bicycle rather than pedaling and the decrease in this confusion shows that the participants of second test kept pedaling on bicycle more than the participant of the first test did.

5.3. A Probabilistic Approach with Prior Probabilities

The classification accuracies reported in the previous sections are computed with the assumption that a priori probabilities of all activities are equal. However, that is not the case in real world. For example, it is much more likely that a person is walking rather than running at a random moment of the day. That means recognition accuracy for walking activity contributes more to the actual performance of the system than accuracy for running activity does.

In this section, we present a probabilistic version of KNN that considers the prior probabilities of activities that are to be classified. Unlike plain KNN algorithm, in this modified algorithm given in Figure 5.5, the classifier takes both the number of occurrence of a class in k nearest neighbors and the specified prior probability of that class. As shown in the pseudocode, the likelihood of selecting a class increases with

```
C := Set of classes
D := Training dataset
t := Input test sample
p_c := Prior probability of class c

p_max := 0
r := NULL
N_k := Find k nearest neighbors of t in D
for all c ∈ C do
  o_c := Number of occurrences of class c in N_k
  p := o_c · p_c
  if  p > p_max then
    p_max := p
    r := c
  end if
end for
return  r
```

Figure 5.5. A probabilistic version of KNN which takes prior probabilities into account.

both the number of occurrence in the nearest neighbors and the prior probability of that class. By the help of this algorithm we analyzed the effect of prior probabilities of the activities on the overall recognition performance of the system in real world.

For evaluating the system performance, we used two different vectors that contain prior probabilities for the five activities that are classified in this study. While constructing these vectors we analyzed the share of different transportation modes that are preferred by the people of two famous metropolises, Istanbul and Amsterdam. We have extracted the percentages of motorized transport, bicycling and walking in the total transportation time of people by looking at the figures presented in [45] and [46] for these cities. As shown in Table 5.12, there is a significant difference about bicycle usage between Istanbul and Amsterdam. Since the share of bicycling is very small

Table 5.12. Percentage of transportation times for Istanbul and Amsterdam.

	Istanbul	**Amsterdam**
Motorized	75%	62%
Walking	24%	21%
Bicycling	1%	17%

for Istanbul, the classification accuracy of the proposed solution will have very little impact on the real performance in that city.

Unlike the transportation options mentioned above, the ratios of idling and running activities during a day does not vary that much between different cities. So we will make logical assumptions for the share of these two activities. In today's world, most people stay physically idle during most of their work hours and when they are at home. If we assume that a person is awake and active for 12 hours, does exercise for one hour and spends two hours for transportation or travelling every day in average, in the remaining nine hours he can be considered physically idle (working at the desk, watching TV, resting, etc.). We divided the time for exercise between running and bicycling equally and distributed the transportation time according to the percentages given for Istanbul and Amsterdam. As a result, we obtained two vectors composed of a priori probabilities of activities for these cities as shown in Table 5.13.

Using the prior probabilities given in Table 5.13, we performed leave-one-out cross validation tests by employing the modified KNN algorithm (Figure 5.5). In the previous offline tests, we tested the system with equal amount of data from each activity. But

Table 5.13. A priori probabilities of activities for Istanbul and Amsterdam.

	Istanbul	**Amsterdam**
Idling	75%	75%
Motorized	12.4%	10%
Bicycling	4.2%	7.5%
Walking	4.2%	3.3%
Running	4.2%	4.2%

Table 5.14. Real world recognition accuracy with different prior probabilities.

	KNN	**Modified KNN**
Uniform	95.26%	95.26%
Istanbul	97.17%	97.32%
Amsterdam	97.34%	97.49%

in this section, in order to simulate the prior probabilities of the activities, we adjusted the amount of test data for each activity so that the occurrence rate of each activity in the test data is proportional to its prior probability.

After forming the test data according to prior probabilities, we tested the system with both the plain KNN algorithm that was used throughout the study and the modified probabilistic version which is presented in this section (Figure 5.5). Testing with the first classifier is expected to show us how the overall recognition performance of the system is effected when the prior probabilities of activities are not uniform and the second test will indicate whether the described probabilistic KNN approach contributes to that performance. Throughout these tests, we used the feature set and the segmentation setup that are shown to perform best in the offline tests. This was also the configuration that we used for online tests.

The results of the tests are given in Table 5.14. In this table, first column shows the recognition accuracies attained by using simple KNN algorithm and the second one contains the results for the probabilistic KNN algorithm that was described early in this section. When the prior probabilities of activities are assumed to be equal, both algorithms behave the same as shown in the first row. So, 95.26% is actually the classification performance that we attained in the offline tests where the prior probabilities are supposed to be equal.

Looking at the next rows of the table, we can see how the prior probabilities of activities effect the real performance of the system. For both Istanbul and Amsterdam, these probabilities provided a performance increase around 2%. The reason behind this increase is the higher prior probabilities of the activities that can be classified more accurately by the system. For example, as the results in the previous sections suggest, *idling* is one of the most successfully classified activities and for both Istanbul and Amsterdam it is by far the most probable activity of a person during the day. On the other hand, the probabilities of other four activities slightly differ between Istanbul and Amsterdam and this accounts for the small performance difference between these cities.

The second KNN algorithm that we used in this section was a very simple modification of the basic algorithm that makes use of prior probabilities of classes and in this section we tried to demonstrate the effect of such a classifier when the probabilities of the activities are not uniform. Table 5.14 indicates that the probabilistic version of the KNN algorithm provides slightly better performance for both Istanbul and Amsterdam. So, it seems that making use of prior probabilities within the classification algorithm helps in increasing the classification accuracy of the system.

In this section, we presented a brief demonstration of how prior probabilities of activities change the real performance of the system and whether it is possible to increase the recognition accuracy by considering prior probabilities in the classification algorithm. As the results suggest, the performance of the system in the real world is very much effected by these probabilities and it is possible to increase the classification accuracy by adapting the classifier with different priors.

5.4. Classification Smoothing Analysis

Smoothing the classifier output is often useful for filtering out singular misclassifications by looking at the dominant (presumably correct) classification around each output. In this section, we analyzed the contribution of classification smoothing on the overall system performance.

The smoothing is performed by applying a majority filter. In this operation, a fixed-size window is slided over the classification output and at each iteration the classification output in the center of the window is changed to the majority class in that window. If there is no majority class, the classification in the center is not changed. Here, the majority class is defined as the class that constitutes more than a given percentage (threshold) of the window.

In Figure 5.6, the smoothing procedure is illustrated. In this example, window size is taken as 5 and the threshold percentage for majority class is 50%. As shown in this figure, two classifier outputs of class M are filtered out due to the dominant output class S.

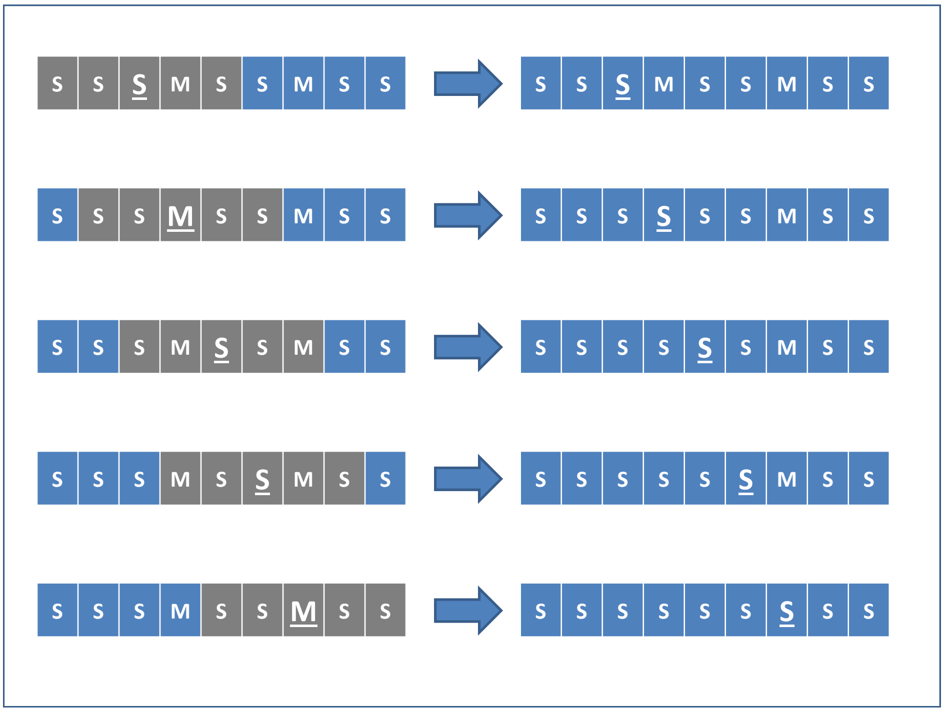


Figure 5.6. Classification smoothing.

In order to see the progress that can be attained by using classification smoothing, we performed offline leave-one-out cross validation tests with the samples of 10 individuals and applied the smoothing procedure to classification output. In these tests, we used the feature set and the segmentation setup that we preferred for online tests. As mentioned, there are two important parameters for smoothing. One of them is the window size for which the majority filter is applied and the second one is the threshold rate that is required for a class to be considered as major class. We evaluated the performance for four different window sizes (5, 10, 20, 30 classifications) and five different threshold rates (0.5, 0.6, 0.7, 0.8). Since our system produces 1 classification output per second, it can be more clear if we think of these window sizes as seconds. For example, if a window size of 10 classifications is used, then at each iteration the smoothing algorithm will consider the majority class in a period of 10 seconds and that means, as a results of the smoothing, each classification result will be effected by the classifications done during the 5 seconds before and after it.

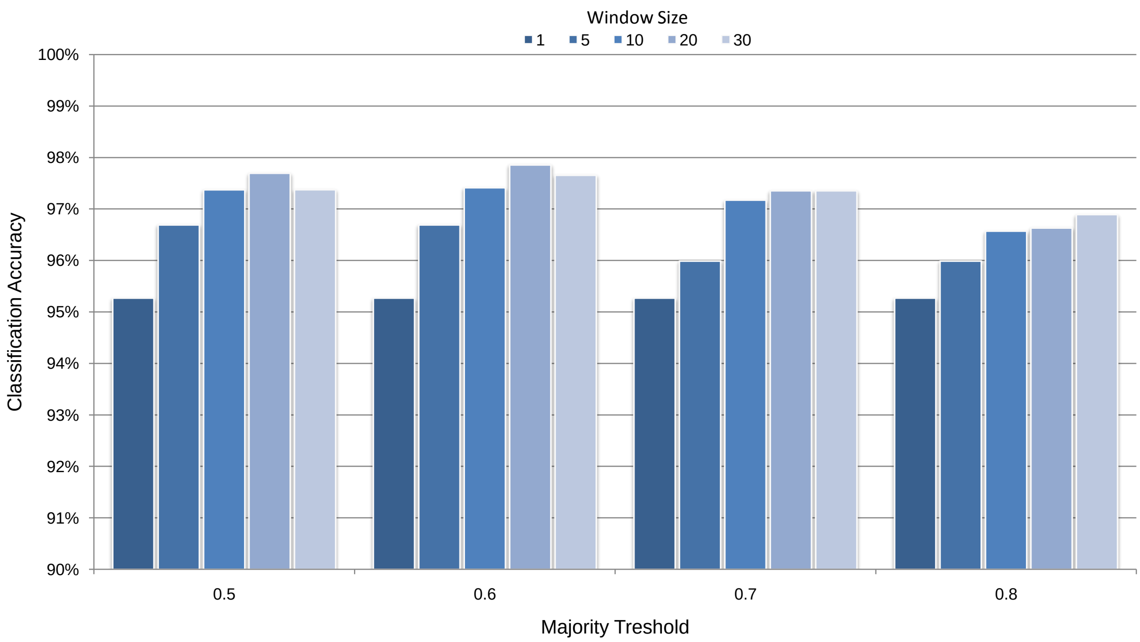


Figure 5.7. Classification performance with smoothing.

Figure 5.7 shows the change in the classification performance of the system after applying the smoothing procedure with different parameters. In this graph, the window size of 1 stands for the classification without smoothing and that is actually the result that we attained in Section 5.1. As the graph suggests, there is an obvious increase in the classification accuracy after applying smoothing on the classification outputs. Having an already high classification accuracy without smoothing (95%), an improvement of 1-2% can be considered to be a remarkable contribution.

When we look at the effect of window size for smoothing, we can see that even a window of size 5 provides a performance increase. Although the windows with the size of 10 or more improve the performance further, there is no significant difference between them. From this result, we can conclude that most of the time a smoothing window of 10 classifications is enough for taking a good advantage of smoothing. That may be due to the fact that temporal misclassification periods generally last less than 10 seconds which can be handled with a smoothing window of size 10.

Another factor that effects the contribution of the smoothing is the threshold rate of occurrence that we used for specifying the major class in a window. As Figure 5.7 suggests, using higher thresholds decrease the performance of smoothing after some point. That is reasonable since misclassifications of the system does not always occur as singular confusions, instead they sometimes occur in a temporal fashion during short periods of time. So, if we use a high threshold for occurrence rate, we can determine major classes for only the windows that contain very few, individual confusions and that will prevent us from recovering temporal confusions that decrease the occurrence rate of the major class in that window.

The results that we attained in this section shows that smoothing classification outputs can improve the performance of the system significantly. However, in this technique, we need to process former and subsequent outputs of the system before deciding if a classification output is correct or should be changed. In other words, the improved result can only be produced with some latency which depends on the window size which is selected for applying the majority filter. So, future works can take advantage of such techniques by relaxing real-time constraints of the system.

6. CONCLUSION AND FUTURE WORK

In this study, we proposed an online activity recognition system which is shown to achieve high recognition accuracies in classifying five daily activities that are walking, running, bicycling, motorized transport and stationary state (sitting/standing/lying) by employing an efficient selection of features. While developing the system, we also evaluated the performance of the system against several different configurations for segmentation of the input data.

Throughout the study, in order to perform recognition online, we used a lightweight classifier (KNN) with different feature sets that contain well-known features together with some specific features. In order to keep the system energy-efficient, all features are extracted from the readings of a single accelerometer on a smartphone. Moreover, we extracted the features from the magnitude of the acceleration vector rather than considering each axis separately and this provided us a system that does not depend on the orientation of the phone.

As a result, we achieved an average recognition rate above 95% in the offline tests and above 92% in the online tests. This is a very promising result when the recognition rates offered by similar studies that are mentioned in Chapter 3 are considered. There have been very few studies that could attain classification performances close to the level achieved by this work, however they lack important advantages offered by this solution like real-time classification on a smartphone, using only a single accelerometer for data collection and orientation independency for the smartphone.

As a future work, the effect of the proposed features with other lightweight classifiers can be analyzed. For example, the performance of the system may be enhanced by using KNN in combination with DT as described in [23]. The confusion matrices presented above supports this idea since there are some activities which are never confused like *running* and *motorized transport*. For such activities, reducing the size of the target activity set with a DT before applying KNN can increase the true classification rate.

On the other hand, the performance of the system can be evaluated with a larger set of activities. Although we evaluated the general daily activities in this study, this can be extended to more specific activities like playing soccer and jumping. The effect of mobile phone positions is another topic that can be studied. We evaluated our solution assuming that the phone is always carried in the trouser pocket and we used relatively tight trousers like jeans for better reading of the body part acceleration. So, how the system behaves for looser clothes is also an open question for future studies.

APPENDIX A: RESULTS OF PERSONALIZED FEATURE SELECTION

Table A.1. Selection rate of features with 1350 msec Rectangular windows and the highest recognition accuracies with mostly-selected features.

Feature	**Selection Rate**			**Highest Recognition Accuracies**
	SFS	**SBS**	**Overall**	
$\boldsymbol{\sigma_A^2}$	10/10	9/10	19/20	
$\boldsymbol{F_{C_{10}}}$	6/10	10/10	16/20	**88.24%**
F_{C_9}	3/10	9/10	12/20	88.00%
$zcr\{R\}$	3/10	9/10	12/20	87.59%
F_{C_3}	3/10	8/10	11/20	88.06%
F_{C_5}	3/10	8/10	11/20	88.00%
μ_A	2/10	9/10	11/10	84.84%
F_{C_6}	1/10	10/10	11/20	
F_{C_1}	2/10	8/10	10/20	
F_{C_8}	2/10	8/10	10/20	
F_{C_7}	1/10	9/10	10/20	
$I_{\max\{F_{C_i}\}}$	3/10	6/10	9/20	
F_{C_2}	3/10	6/10	9/20	
$\max\{R_i\}$	1/10	8/10	9/20	
F_{C_4}	8/10	0/10	8/20	
$\max\{F_{C_i}\}$	2/10	5/10	7/20	
$I_{\max\{R_i\}}$	2/10	4/10	6/20	

Table A.2. Selection rate of features with 1350 msec Hann windows and the highest recognition accuracies with mostly-selected features.

Feature	**Selection Rate**			**Highest Recognition Accuracies**
	SFS	**SBS**	**Overall**	
$\boldsymbol{F_{C_1}}$	7/10	10/10	17/20	
$\mathbf{max}\{\boldsymbol{F_{C_i}}\}$	9/10	6/10	15/20	
$\boldsymbol{\mu_A}$	6/10	9/10	15/10	
$\boldsymbol{F_{C_{10}}}$	5/10	9/10	14/20	
$\boldsymbol{F_{C_7}}$	4/10	10/10	14/20	
$\boldsymbol{I_{\max\{F_{C_i}\}}}$	5/10	7/10	12/20	
$\boldsymbol{F_{C_3}}$	4/10	8/10	12/20	
$\boldsymbol{F_{C_8}}$	3/10	9/10	12/20	86.59%
$\boldsymbol{F_{C_9}}$	3/10	8/10	11/20	86.65%
$\boldsymbol{F_{C_2}}$	3/10	8/10	11/20	86.33%
$\boldsymbol{F_{C_6}}$	3/10	8/10	11/20	85.99%
$\boldsymbol{F_{C_4}}$	2/10	9/10	11/20	**86.67%**
$\max\{R_i\}$	2/10	8/10	10/20	86.51%
F_{C_5}	2/10	7/10	9/20	
$I_{\max\{R_i\}}$	1/10	8/10	9/20	
σ_A^2	3/10	4/10	7/20	
$zcr\{R\}$	2/10	5/10	7/20	

Table A.3. Selection rate of features with 1350 msec Hamming windows and the highest recognition accuracies with mostly-selected features.

Feature	Selection Rate			Highest Recognition Accuracies
	SFS	SBS	Overall	
$\boldsymbol{\mu_A}$	8/10	9/10	17/10	
$\mathbf{max}\{\boldsymbol{F_{C_i}}\}$	8/10	8/10	16/20	
$\boldsymbol{F_{C_1}}$	5/10	10/10	15/20	
$\boldsymbol{F_{C_{10}}}$	6/10	8/10	14/20	
$\boldsymbol{F_{C_5}}$	5/10	9/10	14/20	
$\boldsymbol{F_{C_9}}$	5/10	8/10	13/20	
$\boldsymbol{F_{C_2}}$	4/10	8/10	12/20	83.50%
$\boldsymbol{F_{C_4}}$	4/10	8/10	12/20	83.64%
$\boldsymbol{F_{C_6}}$	4/10	7/10	11/20	83.96%
$\boldsymbol{F_{C_8}}$	3/10	8/10	11/20	83.92%
$\boldsymbol{F_{C_7}}$	2/10	8/10	10/20	**84.14%**
$I_{\max\{F_{C_i}\}}$	4/10	5/10	9/20	83.27%
F_{C_3}	2/10	7/10	9/20	
$I_{\max\{R_i\}}$	1/10	7/10	8/20	
$zcr\{R\}$	1/10	6/10	7/20	
σ_A^2	2/10	2/10	4/20	
$\max\{R_i\}$	0/10	3/10	3/20	

Table A.4. Selection rate of features with 2700 msec Rectangular windows and the highest recognition accuracies with mostly-selected features.

Feature	Selection Rate			Highest Recognition Accuracies
	SFS	SBS	Overall	
$\boldsymbol{\sigma_A^2}$	10/10	9/10	19/20	
$\boldsymbol{F_{C_1}}$	7/10	9/10	16/20	
$\boldsymbol{F_{C_4}}$	5/10	9/10	14/20	
$\boldsymbol{F_{C_9}}$	3/10	10/10	13/20	
$\boldsymbol{F_{C_5}}$	5/10	7/10	12/20	
$\boldsymbol{F_{C_6}}$	4/10	8/10	12/20	94.73%
$\boldsymbol{F_{C_7}}$	3/10	9/10	12/20	94.67%
$\boldsymbol{F_{C_{10}}}$	2/10	10/10	11/20	94.63%
$\boldsymbol{F_{C_2}}$	2/10	9/10	11/20	95.05%
$\boldsymbol{F_{C_3}}$	2/10	8/10	10/20	**95.26%**
$zcr\{R\}$	0/10	9/10	9/20	93.03%
μ_A	0/10	9/10	9/10	
$I_{\max\{F_{C_i}\}}$	4/10	4/10	8/20	
$\max\{F_{C_i}\}$	1/10	6/10	7/20	
F_{C_8}	0/10	7/10	7/20	
$\max\{R_i\}$	0/10	5/10	5/20	
$I_{\max\{R_i\}}$	1/10	2/10	3/20	

Table A.5. Selection rate of features with 2700 msec Hann windows and the highest recognition accuracies with mostly-selected features.

Feature	**Selection Rate**			**Highest Recognition Accuracies**
	SFS	**SBS**	**Overall**	
$\boldsymbol{F_{C_1}}$	8/10	10/10	18/20	
$\boldsymbol{F_{C_4}}$	7/10	8/10	15/20	
$\boldsymbol{F_{C_3}}$	4/10	9/10	13/20	
$\boldsymbol{F_{C_6}}$	4/10	9/10	13/20	
$\mathbf{max}\boldsymbol{\{F_{C_i}\}}$	5/10	7/10	12/20	
$\boldsymbol{\mu_A}$	4/10	8/10	12/10	
$\boldsymbol{F_{C_2}}$	4/10	8/10	12/20	
$\boldsymbol{F_{C_7}}$	3/10	9/10	12/20	94.45%
$\boldsymbol{F_{C_8}}$	3/10	9/10	12/20	94.47%
$\boldsymbol{F_{C_5}}$	2/10	8/10	10/20	**94.71%**
F_{C_9}	2/10	8/10	10/20	94.59%
$I_{\max\{R_i\}}$	1/10	8/10	9/20	94.61%
$F_{C_{10}}$	1/10	8/10	9/20	94.55%
$\max\{R_i\}$	0/10	9/10	9/20	
$I_{\max\{F_{C_i}\}}$	3/10	2/10	5/20	
$zcr\{R\}$	1/10	4/10	5/20	
σ_A^2	3/10	1/10	4/20	

Table A.6. Selection rate of features with 2700 msec Hamming windows and the highest recognition accuracies with mostly-selected features.

Feature	**Selection Rate**			**Highest Recognition Accuracies**
	SFS	**SBS**	**Overall**	
$\mathbf{max}\{\boldsymbol{F_{C_i}}\}$	9/10	6/10	15/20	
$\boldsymbol{\mu_A}$	7/10	8/10	15/10	
$\boldsymbol{F_{C_4}}$	7/10	8/10	15/20	
$\boldsymbol{F_{C_1}}$	6/10	9/10	15/20	
$\boldsymbol{F_{C_6}}$	5/10	9/10	14/20	
$\boldsymbol{F_{C_3}}$	5/10	8/10	13/20	
$\mathbf{max}\{\boldsymbol{R_i}\}$	5/10	7/10	12/20	
$\boldsymbol{F_{C_2}}$	4/10	8/10	12/20	91.73%
$\boldsymbol{F_{C_7}}$	4/10	8/10	12/20	91.69%
$\boldsymbol{F_{C_5}}$	3/10	8/10	11/20	**91.97%**
F_{C_8}	3/10	6/10	9/20	91.83%
$F_{C_{10}}$	3/10	5/10	8/20	91.62%
F_{C_9}	1/10	7/10	8/20	91.54%
$zcr\{R\}$	1/10	5/10	6/20	
σ_A^2	2/10	3/10	5/20	
$I_{\max\{R_i\}}$	0/10	4/10	4/20	
$I_{\max\{F_{C_i}\}}$	1/10	0/10	1/20	

Table A.7. Selection rate of features with 5200 msec Rectangular windows and the highest recognition accuracies with mostly-selected features.

Feature	Selection Rate			Highest Recognition Accuracies
	SFS	SBS	Overall	
$\boldsymbol{\sigma_A^2}$	10/10	10/10	20/20	
$\boldsymbol{F_{C_2}}$	4/10	10/10	14/20	91.47%
$\boldsymbol{F_{C_3}}$	4/10	10/10	14/20	92.95%
$\boldsymbol{F_{C_8}}$	3/10	10/10	13/20	94.10%
$\boldsymbol{F_{C_6}}$	4/10	7/10	11/20	95.67%
$\boldsymbol{F_{C_9}}$	3/10	7/10	10/20	**95.76%**
μ_A	2/10	8/10	10/10	85.07%
F_{C_4}	1/10	9/10	10/20	
F_{C_5}	2/10	7/10	9/20	
$\max\{R_i\}$	1/10	8/10	9/20	
F_{C_7}	0/10	9/10	9/20	
$\max\{F_{C_i}\}$	1/10	7/10	8/20	
$F_{C_{10}}$	1/10	7/10	8/20	
F_{C_1}	0/10	8/10	8/20	
$I_{\max\{F_{C_i}\}}$	3/10	4/10	7/20	
$zcr\{R\}$	0/10	7/10	7/20	
$I_{\max\{R_i\}}$	1/10	4/10	5/20	

Table A.8. Selection rate of features with 5200 msec Hann windows and the highest recognition accuracies with mostly-selected features.

<table>
<tr><th rowspan="2">Feature</th><th colspan="3">Selection Rate</th><th rowspan="2" colspan="6">Highest Recognition Accuracies</th></tr>
<tr><th>SFS</th><th>SBS</th><th>Overall</th></tr>
<tr><td>$\mathbf{max\{\boldsymbol{F_{C_i}}\}}$</td><td>10/10</td><td>10/10</td><td>20/20</td><td rowspan="8">94.94%</td><td rowspan="9">**95.76%**</td><td rowspan="10">95.67%</td><td rowspan="11">95.56%</td><td rowspan="12">95.65%</td><td rowspan="13">95.69%</td></tr>
<tr><td>$\boldsymbol{F_{C_5}}$</td><td>6/10</td><td>8/10</td><td>14/20</td></tr>
<tr><td>$\boldsymbol{F_{C_3}}$</td><td>4/10</td><td>9/10</td><td>13/20</td></tr>
<tr><td>$\boldsymbol{F_{C_8}}$</td><td>3/10</td><td>9/10</td><td>12/20</td></tr>
<tr><td>$\boldsymbol{F_{C_2}}$</td><td>4/10</td><td>7/10</td><td>11/20</td></tr>
<tr><td>$\boldsymbol{F_{C_6}}$</td><td>4/10</td><td>7/10</td><td>11/20</td></tr>
<tr><td>$\boldsymbol{F_{C_4}}$</td><td>3/10</td><td>8/10</td><td>11/20</td></tr>
<tr><td>$\boldsymbol{F_{C_9}}$</td><td>3/10</td><td>8/10</td><td>11/20</td></tr>
<tr><td>$\boldsymbol{\mu_A}$</td><td>2/10</td><td>9/10</td><td>11/10</td><td></td></tr>
<tr><td>$\max\{R_i\}$</td><td>1/10</td><td>10/10</td><td>11/20</td><td></td><td></td></tr>
<tr><td>F_{C_1}</td><td>1/10</td><td>9/10</td><td>10/20</td><td></td><td></td><td></td></tr>
<tr><td>$F_{C_{10}}$</td><td>2/10</td><td>7/10</td><td>9/20</td><td></td><td></td><td></td><td></td></tr>
<tr><td>$I_{\max\{R_i\}}$</td><td>1/10</td><td>6/10</td><td>7/20</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>$zcr\{R\}$</td><td>0/10</td><td>7/10</td><td>7/20</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>F_{C_7}</td><td>0/10</td><td>7/10</td><td>7/20</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>σ_A^2</td><td>1/10</td><td>5/10</td><td>6/20</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>$I_{\max\{F_{C_i}\}}$</td><td>3/10</td><td>1/10</td><td>4/20</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

Table A.9. Selection rate of features with 5200 msec Hamming windows and the highest recognition accuracies with mostly-selected features.

Feature	**Selection Rate**			**Highest Recognition Accuracies**
	SFS	**SBS**	**Overall**	
$\boldsymbol{\max\{F_{C_i}\}}$	8/10	9/10	17/20	
$\boldsymbol{F_{C_3}}$	5/10	10/10	15/20	
$\boldsymbol{F_{C_2}}$	6/10	8/10	14/20	
$\boldsymbol{zcr\{R\}}$	4/10	10/10	14/20	
$\boldsymbol{F_{C_5}}$	5/10	8/10	13/20	
$\boldsymbol{F_{C_9}}$	4/10	8/10	12/20	
$\boldsymbol{F_{C_6}}$	3/10	9/10	12/20	
$\boldsymbol{F_{C_{10}}}$	3/10	9/10	12/20	92.56%
$\boldsymbol{I_{\max\{R_i\}}}$	3/10	8/10	11/20	92.80%
$\boldsymbol{\mu_A}$	2/10	9/10	11/10	**93.66%**
F_{C_1}	2/10	8/10	10/20	93.60%
F_{C_8}	2/10	8/10	10/20	93.64%
F_{C_4}	4/10	5/10	9/20	93.66%
$\max\{R_i\}$	0/10	9/10	9/20	
F_{C_7}	1/10	7/10	8/20	
σ_A^2	6/10	1/10	7/20	
$I_{\max\{F_{C_i}\}}$	2/10	1/10	3/20	

REFERENCES

1. Apple, *iOS Developer Library*, 2012, `http://developer.apple.com/library/IOS`, accessed at June 2012.

2. Preece, S. J., J. Y. Goulermas, L. P. J. Kenney, D. Howard, K. Meijer and R. Crompton, "Activity Identification Using Body-Mounted Sensors — A Review of Classification Techniques", *Physiological Measurement*, Vol. 30, No. 4, pp. R1–R33, 2009.

3. Lane, N., M. Mohammod, M. Lin, X. Yang, H. Lu, S. Ali, A. Doryab, E. Berke, T. Choudhury and A. Campbell, "BeWell: A Smartphone Application to Monitor, Model and Promote Wellbeing", *Pervasive Health 2011*, 2011.

4. Wu, W. H., A. A. T. Bui, M. A. Batalin, L. K. Au, J. D. Binney and W. J. Kaiser, "MEDIC: Medical Embedded Device for Individualized Care.", *Artificial Intelligence in Medicine*, Vol. 42, No. 2, pp. 137–152, 2008.

5. Yavuz, G., M. Kocak, G. Ergun, H. O. Alemdar, H. Yalcin, O. D. Incel and C. Ersoy, "A Smartphone Based Fall Detector with Online Location Support", *Proceedings of PhoneSense 2010*, Zurich, Switzerland, 2010.

6. Najafi, B., K. Aminian, A. Paraschiv-Ionescu, F. Loew, C. J. Bula and P. Robert, "Ambulatory System for Human Motion Analysis Using a Kinematic Sensor: Monitoring of Daily Physical Activity in the Elderly", *IEEE Transactions on Biomedical Engineering*, Vol. 50, No. 6, pp. 711–723, 2003.

7. Long, X. L. X., B. Y. B. Yin and R. M. Aarts, "Single-Accelerometer-Based Daily Physical Activity Classification.", *Conference Proceedings of the International Conference of IEEE Engineering in Medicine and Biology Society*, Vol. 2009, pp. 6107–6110.

8. Lockhart, J. W., T. Pulickal and G. M. Weiss, "Applications of Mobile Activity Recognition", *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, pp. 1054–1058, ACM, New York, NY, USA, 2012.

9. Bigargaddi, N., A. Sarela, L. Klingbeil and M. Karunanithi, "Detecting Walking Activity in Cardiac Rehabilitation by Using Accelerometer", *International Conference on Intelligent Sensors, Sensor Networks and Information Processing' 07*, pp. 555–560, 2007.

10. Zeng, H. and Y. Zhao, "Sensing Movement: Microsensors for Body Motion Measurement", *Sensors*, Vol. 11, No. 1, pp. 638–660, 2011.

11. Pentland, A., "Looking at People: Sensing for Ubiquitous and Wearable Computing", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, pp. 107–119, 2000.

12. Devaul, R. W. and S. Dunn, "Real-Time Motion Classification for Wearable Computing Applications", *Analysis*, Vol. 112, No. 11, pp. 1–14, 2001.

13. Avci, A., S. Bosch, M. Marin-Perianu, R. Marin-Perianu and P. Havinga, "Activity Recognition Using Inertial Sensing for Healthcare, Wellbeing and Sports Applications: A Survey", *23th International Conference on Architecture of Computing Systems, ARCS 2010*, pp. 167–176, 2010.

14. Miluzzo, E., C. T. Cornelius, A. Ramaswamy, T. Choudhury, Z. Liu and A. T. Campbell, "Darwin Phones: The Evolution of Sensing and Inference on Mobile Phones", *Proceedings of the 8th international conference on Mobile systems, applications, and services*, MobiSys '10, pp. 5–20, ACM, New York, NY, USA, 2010.

15. Saponas, T. S., J. Lester, J. Froehlich, J. Fogarty and J. Landay, *iLearn on the iPhone: Real-Time Human Activtity Classification on Commodity Mobile Phones*, Tech. Rep. UW-CSE-08-04–02, 2008.

16. Karantonis, D. M., M. R. Narayanan, M. Mathie, N. H. Lovell and B. G. Celler, "Implementation of a Real-Time Human Movement Classifier Using a Triaxial Accelerometer for Ambulatory Monitoring", *IEEE Transactions on Information Technology in Biomedicine*, pp. 156–167, 2006.

17. Pirttikangas, S., K. Fujinami and T. Nakajima, "Feature Selection and Activity Recognition from Wearable Sensors", *Ubiquitous Computing Systems, Third International Symposium, UCS 2006*, pp. 516–527, 2006.

18. Khan, A., Y. Lee, S. Lee and T. Kim, "Human Activity Recognition via An Accelerometer-Enabled-Smartphone Using Kernel Discriminant Analysis", *Future Information Technology (FutureTech), 2010 5th International Conference on*, pp. 1–6, 2010.

19. Berchtold, M., M. Budde, D. Gordon, H. Schmidtke and M. Beigl, "ActiServ: Activity Recognition Service for Mobile Phones", *Wearable Computers (ISWC), 2010 International Symposium on*, pp. 1–8, 2010.

20. Könönen, V., J. Mäntyjärvi, H. Similä, J. Pärkkä and M. Ermes, "Automatic Feature Selection for Context Recognition in Mobile Devices", *Pervasive and Mobile Computing*, Vol. 6, No. 2, pp. 181–197, 2010.

21. Yang, J., "Toward Physical Activity Diary: Motion Recognition Using Simple Acceleration Features with Mobile Phones", *Proceedings of the 1st international workshop on Interactive multimedia for consumer electronics IMCE '09*, pp. 1–10, 2009.

22. Reddy, S., D. Estrin and M. Srivastava, "Recruitment Framework for Participatory Sensing Data Collections", *Proceedings of the 8th International Conference on Pervasive Computing*, pp. 138–155, Berlin, Heidelberg, 2010.

23. Siirtola, P. and J. Röning, "Recognizing Human Activities User-independently on Smartphones Based on Accelerometer Data", *International Journal of Interactive*

Multimedia and Artificial Intelligence, Vol. 1, No. 5, pp. 38–45, 06/2012 2012.

24. Huynh, T. and B. Schiele, “Analyzing Features for Activity Recognition”, *2005 joint conference on Smart objects and ambient intelligence: innovative context-aware services: usages and technologies (sOcEuSAI2005)*, pp. 159–163, ACM Press New York, NY, USA, ACM Press New York, NY, USA, Grenoble, France, 2005.

25. Preece, S. J., J. Y. Goulermas, L. P. J. Kenney and D. Howard, “A Comparison of Feature Extraction Methods for the Classification of Dynamic Activities from Accelerometer Data”, *IEEE Transactions on Biomedical Engineering*, Vol. 56, No. 3, pp. 871–879, 2009.

26. Kose, M., O. D. Incel and C. Ersoy, “Online Human Activity Recognition on Smart Phones”, *Workshop on Mobile Sensing: From Smartphones and Wearables to Big Data (colocated with IPSN 2012)*, pp. 11–15, Beijing, China, 2012.

27. Yang, J.-Y., J.-S. Wang and Y.-P. Chen, “Using Acceleration Measurements for Activity Recognition: An Effective Learning Algorithm for Constructing Neural Classifiers”, *Pattern Recognition Letters*, Vol. 29, No. 16, pp. 2213–2220, 2008.

28. Krishnan, N. C., D. Colbry, C. Juillard and S. Panchanathan, “Real Time Human Activity Recognition Using Tri-Axial Accelerometers”, *In Sensors Signals and Information Processing Workshop*.

29. Reddy, S., M. Mun, J. Burke, D. Estrin, M. Hansen and M. Srivastava, “Using Mobile Phones to Determine Transportation Modes”, *ACM Transactions on Sensor Networks*, Vol. 6, No. 2, pp. 13:1–13:27, 2010.

30. Bao, L. and S. S. Intille, “Activity Recognition from User-Annotated Acceleration Data”, *Pervasive Computing*, Vol. 3001, pp. 1–17, 2004.

31. Mansfield, A. and G. M. Lyons, “The Use of Accelerometry to Detect Heel Contact Events for Use as a Sensor in FES Assisted Walking”, *Medical Engineering Physics*,

Vol. 25, No. 10, pp. 879–885, 2003.

32. Nyan, M. N., F. E. H. Tay, K. H. W. Seah and Y. Y. Sitoh, "Classification of Gait Patterns in the Time-Frequency Domain.", *Journal of Biomechanics*, Vol. 39, No. 14, pp. 2647–2656, 2006.

33. Könönen, J., J. Mantyjarvi, H. Simila, J. Parkka and M. Ermes, "Automatic Feature Selection for Context Recognition in Mobile Devices", *Pervasive and Mobile Computing*, Vol. 6, pp. 181–197, 2010.

34. Ward, J. A., P. Lukowicz and G. Troster, "Gesture Spotting Using a Wrist Worn Microphone and 3-axis Accelerometer", *EUSAI*, 2005.

35. Junker, H., O. Amft, P. Lukowicz and G. Tröster, "Gesture Spotting with Body-Worn Inertial Sensors to Detect User Activities", *Pattern Recognition*, Vol. 41, No. 6, pp. 2010–2024, 2008.

36. Krishnan, N. C., C. Juillard, D. Colbry and S. Panchanathan, "Recognition of Hand Movements Using Wearable Accelerometers", *Journal of Ambient Intelligence Smart Environment*, Vol. 1, No. 2, pp. 143–155, 2009.

37. Fabian, A., N. Gyorbiro and G. Homanyi, "Activity Recognition System for Mobile Phones Using the MotionBand Device", *Proceedings of the 1st international conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications*, MOBILWARE '08, pp. 41:1–41:5, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, 2007.

38. Bieber, G., P. Koldrack, C. Sablowski, C. Peter and B. Urban, "Mobile Physical Activity Recognition of Stand Up and Sit Down Transitions for User Behaviour Analysis", *Proceedings of the 3rd International Conference on Pervasive Technologies Related to Assistive Environments, PETRA*, 2010.

39. Fitz-Walter, Z. and D. Tjondronegoro, "Simple Classification of Walking Activities

Using Commodity Smart Phones", *OZCHI '09 Proceedings of the 21st Annual Conference of the Australian Computer-Human Interaction Special Interest Group: Design: Open 24/7*, pp. 409–412, 2009.

40. Mladenov, M. and M. Mock, "A Step Counter Service for Java-Enabled Devices Using a Built-in Accelerometer", *Proceedings of the 1st International Workshop on Context-Aware Middleware and Services: affiliated with the 4th International Conference on Communication System Software and Middleware (COMSWARE 2009)*, CAMS '09, pp. 1–5, ACM, New York, NY, USA, 2009.

41. Iso, T. and K. Yamazaki, "Gait Analyzer Based on a Cell Phone with a Single Three-Axis Accelerometer", *MobileHCI '06 Proceedings of the 8th conference on Human-computer interaction with mobile devices and services*, pp. 141–144, 2006.

42. Farrahi, K. and D. Gatica-Perez, "Daily Routine Classification from Mobile Phone Data", *MLMI '08 Proceedings of the 5th international workshop on Machine Learning for Multimodal Interaction*, pp. 173–184, 2008.

43. Google, *Android Developer Tutorial*, 2012, `http://developer.android.com`, accessed at June 2012.

44. Chen, K. Y., K. F. Janz, W. Zhu and R. J. Brychta, "Redefining the Roles of Sensors in Objective Physical Activity Monitoring.", *Medicine & Science in Sports & Exercise*, Vol. 44, No. 1 Suppl 1, pp. 13–23, 2012.

45. Rietveld, P. and V. Daniel, "Determinants of bicycle use: do municipal policies matter?", *Transportation Research Part A: Policy and Practice*, Vol. 38, No. 7, pp. 531–550, 2004.

46. Hennig, M., *Sustainable Urban Mobility: The Example of Istanbul*, Case study, Turkish Ministry of Transportation, Ankara, 2011.