ONLINE CONTEXT RECOGNITION WITH MOBILE PHONE SENSING
(AKILLI TELEFONLAR ÜZERİNDE GERÇEK ZAMANLI EYLEM TANIMA)

by

Doruk COŞKUN, B.S.

Thesis

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

in

COMPUTER ENGINEERING

in the

INSTITUTE OF SCIENCE AND ENGINEERING

of

GALATASARAY UNIVERSITY

September 2014

ONLINE CONTEXT RECOGNITION WITH MOBILE PHONE SENSING
(AKILLI TELEFONLAR ÜZERİNDE GERÇEK ZAMANLI EYLEM TANIMA)

by

Doruk COŞKUN, B.S.

Thesis

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

Date of Submission : September 22, 2014

Date of Defense Examination: October 16, 2014

ACKNOWLEDGEMENTS

This study which is in front of you is the result of research in the Graduate School of Science and Engineering at Galatasaray University. This work is supported by the Office of Scientific Research Projects (BAP) at Galatasaray University, number 13.401.002, and by TUBITAK under the project number 113E271.

I want to thank all the people who support me to carry out this work. My first words of thanks go to Assist. Prof. Atay Özgövde and Assist. Prof. Özlem Durmaz İncel for their advices, encouragement, their high availability and attitude of respect and understanding. During my work, I was able to enjoy their ideas and their remarkable intuition.

I cannot forget my supervisors Assoc. Prof. Francois Portet and David Blachon from LIG Lab at Grenoble. My sincere thanks also go to them for their interest and support for my work during my internship in their team.

Last but not the least, I would like to thank my family for their encouragement and support for my efforts throughout my studies.

Doruk COŞKUN
20.09.2014

TABLE OF CONTENTS

ACKNOWLEDGEMENTS……………………………………………………………..ii

TABLE OF CONTENTS…………………………………………………………….iii

LIST OF ACRONYMS/ABBREVIATIONS………………………………………..v

LIST OF FIGURES………………………………………………………………….vi

LIST OF TABLES…………………………………………………………………viii

ABSTRACT……………………………………………………………………......ix

RÉSUMÉ...x

ÖZET……………………………………………………………………………….xi

1. Introduction………………………………………………………………….1

2. Background………………………………………………………………….4

    2.1. Literature Overview……………………………………………………….4

    2.2. Human Activity Modelling and Classification……………………….…...6

    2.3. Evaluation of Human Activity Classifiers……………………………..11

3. Exploratory Activity Recognition Phase……………………………………...17

    3.1. Machine Learning Methods………………….…………………………..17

    3.2. Model Building and Testing……………………………………………...18

    3.3. Clustering Approach……………………………………………………...24

    3.4. Orientation/Position Recognition…….…………………………………..29

    3.5. Android Implementation………………………………………………….33

    3.6. Position-Aware Recognition: Performance Evaluation…..……………….....36

4. ARServ : Advanced Data Collection and Activity Recognition Framework……..40

    4.1. System Overview……………………………………………...………42

5. GSU Activity Set: A New Data Collection Experiments………………………….45

    5.1. Position Recognition..…………………………………………………………47

        5.1.1. Classification with Basic Acceleration Features………………………..47

        5.1.2. Classification with Angular Features…………………………………...48

5.1.3. Classification with Angular and Basic Acceleration Features…………..…49
5.2. Activity Recognition: Performance Evaluation……………………..................50
5.2.1. Position-Independent Activity Classification…………………………..50
5.2.2. Position-Aware Activity Classification……………………...............52
5.2.3. Discussion……………………………………………………………53
6. Conclusion……………………………………………………………………54
References…………………………………………………………………...58
Biographical Sketch………………………………………………………….....61

LIST OF ACRONYMS/ABBREVIATIONS

AR	Activity Recognition
MSP	Mobile Sensing Platform
NB	Naive Bayes
KNN	K-Nearest Neighbors
DT	Decision Tree
SVM	Support Vector Machines
GPS	Global Positioning System
QDA	Qualitative Data Analysis
DHMM	Discrete Hidden Markov Models
RMS	Root Mean Square
SMA	Simplified Memory Bounded A
FFT	Fast Fourier Transform
CHMM	Continuous Hidden Markov Model
KMC	Kinetic Monte Carlo
NN	Nearest Neighbor
SFS	Sequential Forward Selection
SBS	Sequential Backward Selection
GUI	Graphical User Interface

LIST OF FIGURES

Figure 2.1. An off-the-self iPhone 4, representative of the growing class of sensor enabled phones …………………………………………………7
Figure 2.2. Typical steps of activity recognition ……………………………………9
Figure 2.3. Types of features ……………………………………………………10
Figure 3.1. Main Screen of AndroidAcc……………………………………………20
Figure 3.2. Test results of the data without stairs…………………………………...21
Figure 3.3. Test results of the data with stairs………………………………………22
Figure 3.4. Test results of the data with FFT features……………………………...23
Figure 3.5. Test results of the data with “Stairs” activity…………………………..24
Figure 3.6. Test results of K-means cluster analysis………………………………..25
Figure 3.7. Test results of K-means cluster analysis with 3 activities……………...26
Figure 3.8. Test results of AscendingStairs-Walking and AscendingStairs-Running………………………………………………27
Figure 3.9. Test results of DescendingStairs-Running and DescendingStairs-Walking………………………………………….…27
Figure 3.10. Test results of DescendingStairs-AscendingStairs……………………..28
Figure 3.11. Test results of DescendingStairs-Walking and AscendingStairs-Walking…………………….……………………………29
Figure 3.12. Orientation axis of a smartphone……………………………….........30
Figure 3.13. Orientation readings of the both sensor………………………………...31
Figure 3.14. Test results of orientation recognition………………………….........32
Figure 3.15. Test results of the new dataset with pants pocket orientation……........34
Figure 3.16. ARService and service readings………………………………………..35
Figure 4.1. ARServ GUI……………………………………………………………41
Figure 4.2. Sensor Selection Interface……………………………………………...42
Figure 4.3. Online Markup Popup…………………………………………………..43
Figure 5.1. Data Collection Scenario Locations on Google Maps …………………46

Figure 6.1. Summary of activity recognition applications 56

LIST OF TABLES

Table 2.1. Precision results for classifiers…....……………………………………..12
Table 2.2. Recall results for classifiers…..……………………………………...12
Table 2.3. Accuracy of the activity recognition using different feature groups.…..13
Table 2.4. Aggregate recognition rates for activities…..…………………………14
Table 2.5. Comparison of clustered KNN, NB and DT without biking…..………15
Table 2.6. Impact of window size and sampling interval on the accuracy rates of classifiers with biking…...………………………………………...15
Table 3.1. Classification Accuracy of position-independent model……………….36
Table 3.2. Classification accuracy of pants pocket position model………..........37
Table 3.3. Classification accuracy of messenger bag position model……………..37
Table 3.4. Classification accuracy of backpack position model…………………...37
Table 3.5. Classification accuracy of jacket pocket model………………………...38
Table 3.6. Overall results of weighted averages from position model tables……...38
Table 5.1. Phone position recognition with basic acceleration features ……….…48
Table 5.2. Phone position recognition with angular acceleration features ……….48
Table 5.3. Phone position recognition with basic and angular acceleration features……………………………………………………………49
Table 5.4. Position-independent activity classification …………………………...51
Table 5.5. Position-specific activity classification ……………………………..…52

ABSTRACT

ONLINE CONTEXT RECOGNITION WITH MOBILE PHONE SENSING

Activity Recognition (AR) or in other saying Context Recognition is an active area of research in the domain of pervasive and mobile computing that has direct applications about life quality and health of the users. Previous studies aim to classify different daily human activities with high accuracy rates using various types of sensors. Becoming a substantial part in our daily lives with their sensing capabilities, smartphones are becoming increasingly sophisticated and the latest generations of smart cell phones now incorporate many diverse and powerful sensors. Therefore, they are now considered feasible platforms that enable people to make use of AR technologies without being obliged to use or wear some extra devices. Nevertheless, 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 a research based on other works in the literature that analyze the performance of the classification methods for online AR systems on smart phones. The previous studies generally focus on single phone location of the users despite the fact that users carry their phones in various positions. Hence, we also focus on phone position uncertainty problem and compare the classification results with position independent and position dependent classification models. Finally, we propose our own implementations to make and run an activity recognition system on an Android based smartphone.

RÉSUMÉ

RECONNAISSANCE DE CONTEXTE EN LIGNE AVEC UN TÉLÉPHONE PORTABLE DE LA DÉTECTION

Reconnaissance d’activités (AR) ou dans d'autres mots reconnaissance de contexte est un domaine de recherche actif dans le domaine de l'informatique omniprésente et mobile qui a des applications directes sur la qualité de vie et la santé des utilisateurs. Les études précédentes ont pour but de classer les différentes activités humaines quotidiennes avec des taux de haute précision en utilisant différents types de capteurs. Devenir une partie substantielle dans notre vie quotidienne avec leurs capacités de détection, les smartphones sont plus en plus sophistiquées et les dernières générations de smartphones intègrent désormais de nombreux capteurs divers et puissants. Par conséquent, ils sont désormais considérés comme les plates-formes possibles qui permettent aux gens de faire usage de technologies AR sans être obligé d'utiliser ou de porter des appareils supplémentaires. Néanmoins, en raison de la puissance de calcul et les contraintes de ces dispositifs, il devient une tâche difficile à atteindre des résultats précis en utilisant l'énergie et gourmandes en temps processeur classificateurs.

Dans cette étude, nous présentons une recherche basée sur d'autres œuvres dans la littérature qui analysent la performance des méthodes de classification des systèmes AR en ligne sur les téléphones intelligents. Les études antérieures se concentrent généralement sur l'emplacement de téléphone unique des utilisateurs, malgré le fait que les utilisateurs effectuent leurs téléphones dans diverses positions. Par conséquent, nous nous concentrons également sur le problème de l'incertitude de position de téléphone et comparons les résultats de la classification des modèles de la position dépendants et indépendants. Enfin, nous proposons nos propres implémentations de faire et exécuter un système de reconnaissance de l'activité sur un smartphone basé Android.

ÖZET

AKILLI TELEFONLAR ÜZERİNDE GERÇEK ZAMANLI EYLEM TANIMA

Aktif bir bilişim araştırma konusu olan eylem tanıma, hayat kalitesi ve e-sağlık gibi uygulama alanlarında günümüzde artarak kullanılmaktadır. Literatürdeki çalışmalar, çeşitli algılayıcı tipleri kullanarak değişik insan aktivitelerini isabetli bir şekilde sınıflandırmaya çalışmaktadır. Hayatımızın değişilmez bir parçası olmaya başlayan akıllı telefonlar geniş algılama kapasiteleri ile son derece sofistike bir hal almış ve günümüz teknolojisinin en kapsamlı çevresel sensörler ile donatılmaya başlanmıştır. Bu durum akıllı telefonları, insanların eylem tanıma teknolojilerini herhangi başka bir cihaz taşımadan kullanabilmelerine olanak sağlamaktadır. Ancak, akıllı telefonlarda hala varolmakta olan güç ve hesaba dayalı kısıtlamalar, yüksek işlem kabiliyetine ihtiyac duyan sınıflandırma algoritmalarının bu cihazlar üzerinde verimli bir şekilde kullanılmasını engellemektedir.

Bu çalışmada, literatürde yer alan diğer çevrim içi aktivite tanıma çalışmaları incelenmiş, çevrim içi sınıflandırmada kullanılan sınıflandırma algoritmalarının test sonuçları analiz edilmiştir. Ayrıca geçmiş çalışmalarda gerçek hayattan farklı olarak kullanıcın telefonun sadece bir pozisyonda taşıdığı varsayımı üzerine yogunlaştığı görüldüğü için telefon taşınma pozisyon sorununun, eylem tanıma sınıflandırma başarımının üzerindeki etkisi incelenmiş ve telefon pozisyonu bilinerek yapılan sınıflandırma başarımı ile telefon pozisyonu bilinmeden yapılan sınıflandırma başarımı karşılaştırılmıştır. Son olarak bu incelemelerden yola çıkılarak Android işletim sistemine sahip telefonlar için çevrim içi aktivite tanıma sistemi gerçeklenmiştir.

1 – INTRODUCTION

The recent advances in networking and sensor technology has created a growing interest in sensor networks which are necessary technology for the development of the concept of ambient intelligence where the users receive services depending on their context. The concept of "context" is very critical because determining the context of the user; it is a key to create a dynamic and flexible ambient intelligence environment. A good effective ambient intelligence environment must provide three major functionalities; context awareness, ubiquitous access and natural interaction. Classical low cost sensors are not satisfying these requirements. Therefore, the researchers must come up with something new.

In recent years, the researchers started using mobile devices on their works on the ambient intelligence. Mobile devices are becoming increasingly sophisticated and the latest generations of smart cell phones now incorporate many diverse and powerful sensors. During years of usage of the smart phones, the two of the three functionalities of the ambient intelligence, which are the natural interaction and ubiquitous access, have been implemented satisfactorily on the ambient intelligence systems. But the context awareness functionality is the notion that the researchers are still working on.

Context awareness is the idea that computers can sense and react to a user's situation; it has been a popular research topic for a number of years. The mobile devices that are generally used for providing the context awareness for an ambient intelligence system are one of the most ubiquitous tools in the progress of the context awareness. They have an enormous popularity and permeation in to daily life. Therefore, they have a perfect potential for context awareness. The very mobility of these devices creates a new concept called "mobile context awareness", where sensing and reacting is enabled by the device itself.

The release of high-end mobile devices like smart-phones enabled the human activity recognition on the mobile platforms. There are other devices like MSP (Mobile Sensing Platform) equipped by a set of sensors like a barometer and humidity sensors, which aren’t found even on today’s smart-phones, were used for AR but distributing the application and collecting data from the users were a great problem. Hence smartphones have a great advantage over MSPs and this advantage is called the application stores. The market of mobile operating systems is shared by two major operating systems, they are, namely IOS and Android. These mobile operating systems have application stores for the developers to distribute their applications. An AR system is based on machine learning models and to build these models, the large amount of data that gathered from the mobile devices has a crucial importance. Since a developer can reach out thousands of people around the world by using these application stores, data gathering and distributing the application to the people is no longer a problem. These lacks of infrastructure of the other mobile devices like MSP cause great disadvantages for making an AR system. Therefore, the uses of the smart phones are eminent because of its high infrastructure.

Smartphones also embed large resources in terms of computation, storage, battery, which could allow performing online embedded activity recognition. However, according to a survey from Incel et al. (2013), online activity recognition is an under explored area. They report that most studies deal with online classification and that classifiers still require much resource for embedding them on smartphones. Yet, some recent work has started studying online AR classification, using Decision Tree and K-Nearest Neighbors (Reddy et al., 2010; Kose et al., 2012)

Beyond those temporary limitations of resources, other issues need to be tackled with. First, the large variability of sensors does not seem to be standardized yet, which means that one should not make a system depend on the availability of such a sensor on every smartphone. For instance, accelerometer is quite common but proximity sensor or barometer is far less common. Hence, a system of human AR should deal with this variable sensor availability. Also, unlike the previously mentioned study of Bao & Intille (2004), sensor location and orientation may change in time. Indeed, smartphone users can carry them in different locations. A survey (Chon et al., 2012) performed among 55 volunteers reported the preferred locations for users to carry their smartphones. The four most frequent answers were hand, pants pocket, bag and jacket pocket. The change of location may make it more difficult for a system to infer user activity, yet it needs to be taken into account.

I presented a different thesis (Coskun, 2014) based on a different point of view on the online activity recognition by using a sequence model approach different from other works in literature that uses sequential algorithms only as supplementary for the classical classification algorithms. This work also focused on phone location uncertainty problem since the previous studies generally focused on single phone location of the users despite the fact that users carry their phones in various positions. I ran some tests in order to address this uncertainty problem by using accelerometer, audio and accelerometer + audio features together which were extracted from raw sensor data that were collected in three different phone locations.

Apart from my previous thesis, this study focuses on the systems that only use embedded accelerometer sensor on smart phones for building an AR framework. The rest of the study is organized as follows. In chapter 2, state of art is presented. Chapter 3 focused on the exploratory phase of activity recognition. Chapter 4 contains our activity recognition framework and chapter 5 is about experiments on data which was collected by using our framework. Finally, chapter 6 will provide you the conclusion and directions for future research.

2 – BACKGROUND

2.1 LITERATURE OVERVIEW

Previous studies are mainly concentrated on the type of the classification which is online or offline and the performance of these classifiers. In Kononen et al. (2010), a system that contains multiple accelerometers is proposed. These accelerometers are located on wrist and hip. This system is implemented for recognizing activities like biking, soccer, lying, walking, rowing, running, sitting, and standing. This system can recognize very large sets of activities. Authors use Min. Distance algorithm, decision trees and support vector machines as classifiers and they obtain relatively accurate results of recognition with SVM (%80) by using offline classification.

In a similar study, Reddy et al. (2010), a system is proposed but this time authors of this paper added a new sensor called GPS that is also located on smart phones. They tried to recognize transportation activities like stationary, walking, biking and motorized transports. Unlike other researchers, they also used the frequency domain features with the time domain features. For example, the Fast-Fourier-Transform coefficients of a sensor signal are a frequency domain feature, but the standard deviation or a mean of a sensor signal is a time domain feature. The studies about the signal processing show us that the frequency domain features has better information about the characteristic of a signal. Authors of this paper proved that, they obtain high recognition results with decision trees + discrete hidden markov model (%93) and SVM (%91). This system also used the offline classification.

We mentioned two papers that used the offline classification to build models and make classifications but there are other studies like (Siirtola & Röning, 2012), (Kononen et al., 2010) that used the online classification. Online classification is a bit different than the offline classification; it basically depends on making classifications in real-time.

In Siirtola & Röning (2012), the authors proposed a system that uses two distinct classifiers for classification process. As you can acknowledge, online classification contains online data streaming and the system must handle the sensor data in real-time. It means that the system makes the classifications on phone and because of the nature of the online streaming and smart phone's limited CPU power; classical classification algorithms can't be used. Generally the researchers use multiple classification algorithms step-by-step just like a single algorithm to handle this online data streaming problem. So in Siirtola & Röning (2012), the authors used decision trees + QDA and decision trees + KNN separately and evaluate their performance. Both methods obtain high recognition results, %95 and %94 respectively.

Similar in Bieber et al. (2010), the authors of this paper proposed a custom classification method called "Cross correlating vertical acceleration waveform with characteristic waveform of sit-to-stand transition". In this research, authors analyze the accelerometer value transition between two activities sit-to-stand and stand-to-sit.

The systems that use online classification have generally one thing in common; they use only one sensor (generally accelerometer) because of the CPU limitation of the smart phones. But due to recent advances in hardware technologies, the researchers on this field believe that the constraints on the CPU limitation can be overcome in time.

Accelerometer appears to be the most popular sensor for the domain. However, we previously noticed in the introduction that smartphone location and orientation might change due to the habits of users that can carry it in different locations. This can have an impact on accelerometer readings as Alanezi & Mishra (2013) report. They collected accelerometer data from two different positions: hand holding and pants pocket. Magnitudes of acceleration hardly reached the value of 15 m/s^2 when in hand while they often exceeded 15 m/s^2 and even reached 20 m/s^2 in pants pocket. The difference was also noticeable on standard deviations of readings. Hence, as the authors concluded, accelerometer data are affected by smartphone position.

2.2 HUMAN ACTIVITY MODELLING AND CLASSIFICATION

In this section, we will explain the classification algorithms and common steps of the AR process. In other words, some activity recognitions systems in the literature will be examined and their methods will be analyzed for using in our implementation of the AR system. A bunch of question must be answered for analyzing a method of an activity recognition system for this purpose. First question that needs to be answered is “What is needed for building an activity recognition system?” Today’s smart phones are not only the key computing or communicational devices; they have also a rich set of sensors such as accelerometer, GPS etc. (Lane et al., 2010). Naturally, these sensors enable new application opportunities across a wide variety of domains. One of these domains is activity recognition.

Sensor-based activity recognition can be performed with mobile devices and today’s technology offers us a new mobile devices; smart phones. A typical smart phone contains many sensors.

But these cheap embedded sensors are not the only reason that researchers use smart phones for their applications. Smart phones are also programmable; they have high computational and communicational resources than other mobiles devices like MSP that are used for activity recognition systems earlier. They have also application stores for distributing the activity recognition application for collecting data from the users of the system.

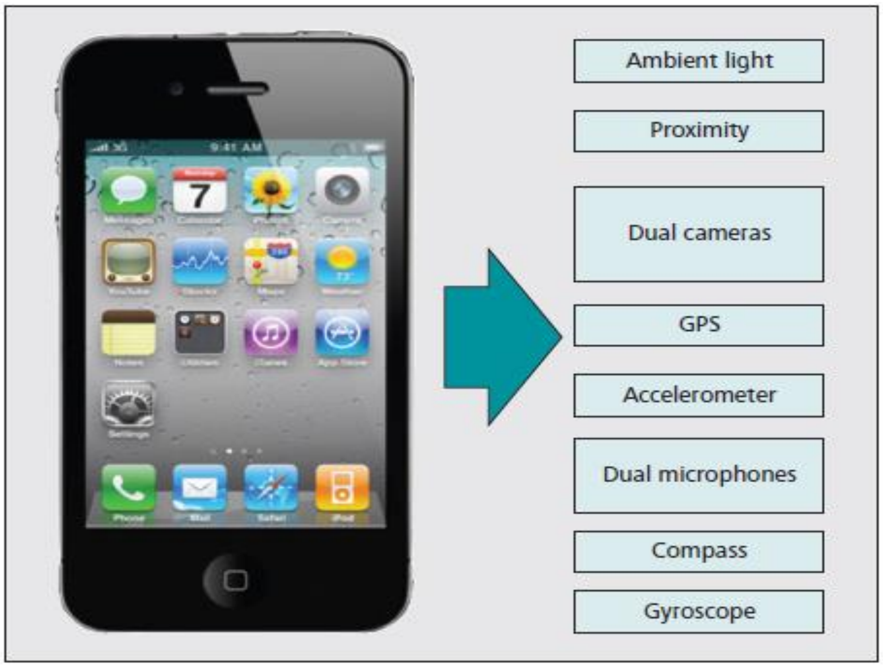


Figure 2.1. An off-the-self iPhone 4, representative of the growing class of sensor enabled phones. (Lane et al., 2010)

Another question that is needed to be answered is “which classification type?” In literature, there are two classification types, offline classification and online classification. Offline classification uses the offline processing, that means the model building and classification process are implemented offline (not real-time) (Kose et al., 2012) . As you can imagine, there are some advantages of using offline classification, for example efficient model building is a computationally challenging task, by doing this offline; you can use the resources of a desktop computer that has a better CPU power than smart phones. On the other hand, online classification is implemented in real-time, that means the model building can be done offline but the classification process must be done in real-time. Another example is in Siirtola & Röning (2012), when we tried to analyze a routine of a daily activities of a person, offline classification can be more appropriate but when it comes to applications such as a fitness coach where the users activities must be observed instantly by a third person, online classification is more appropriate.

Another question that is needed to be answered is “which classification method?” As it was mentioned before, the trendy classification algorithms are not feasible for an activity recognition system, especially the ones that use the online classification. In literature, most of the researchers created a custom classification algorithm by using other classification algorithms step-by-step like in Kose et al. (2012) where authors proposed two-step algorithm by using the decision tree and discrete hidden markov model. Authors created a custom classifier called DT+DHMM and used it in their research. They also obtained good recognition results, but popular classifiers like c4.5 decision tree or support vector machines also obtained high recognition results even when they are used separately. Eventually, it can be deduced the selection of the classification algorithm is important especially when the system is supposed to use the online classification.

A typical online activity recognition system consists of five steps (Incel et al., 2013). After the raw sensor data is collected, the steps of the activity recognition include

I. Preprocessing of sensor data
II. Segmentation
III. Feature extraction
IV. Optionally dimension reduction
V. Classification

The figure 2.2 illustrates these steps in detail.

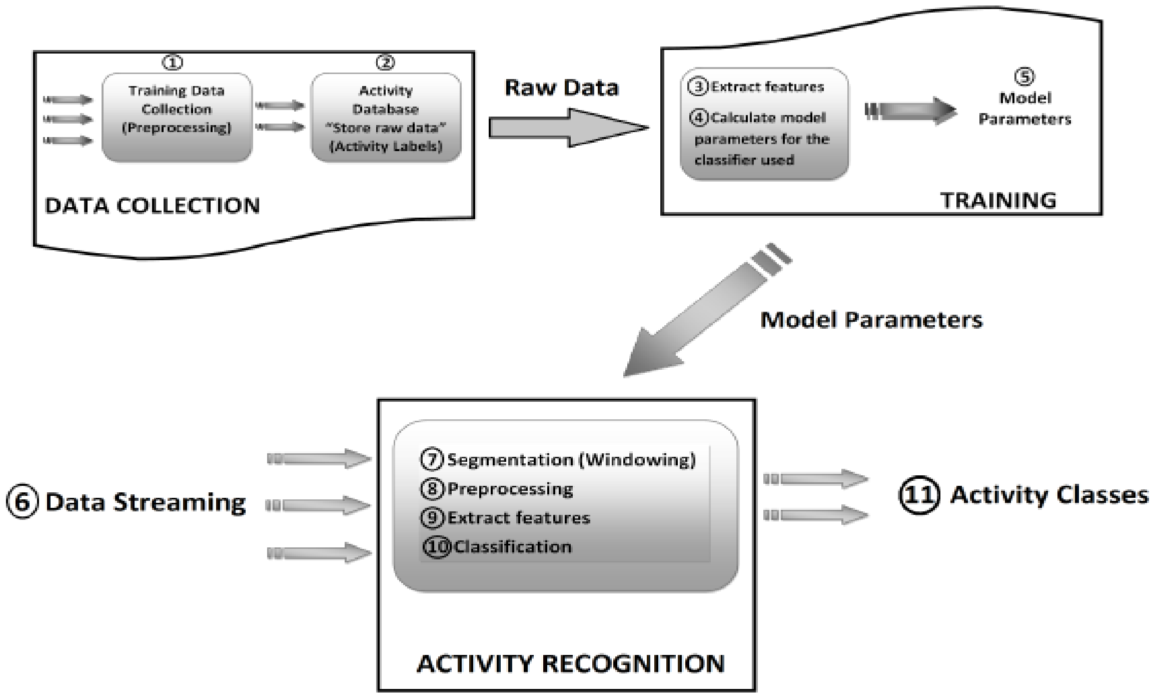


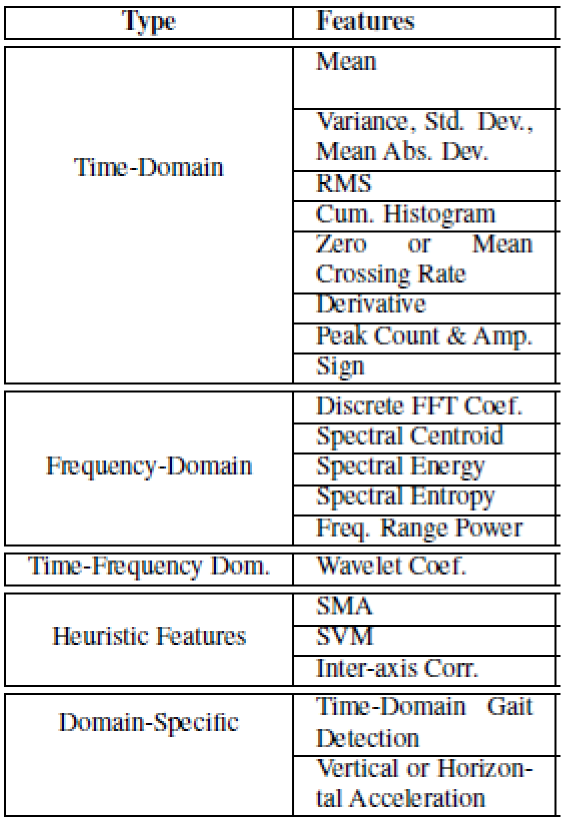
Figure 2.2. Typical steps of activity recognition. (Incel et al., 2013)

The preprocessing step contains data re-organizing algorithms like noise removal or "SMOTE" for preventing the over fitting problem. The segmentation step is applied to continuous data streams of real-time sensor data for dividing the signal in time windows. The feature extraction step is an important step. This step is used for characterizing the raw sensor data. The raw sensor data is itself not feasible for classifying process, so a feature extraction must be applied to raw data to represent the original data in best way (Incel et al., 2013). Dimension reduction step can be applied for removing irrelevant and useless features to decrease the computational cost of the training and classifying process. It also increases the performance of the system (Incel et al., 2013). Last step is the classification step that includes the mapping of the data.

After the examination of these questions that are valid for all of AR systems, the more specific questions are needed to be answered which are highly dependent on the application.

One of these questions is “which activities are needed to be recognized?” It is obvious that these activities are highly dependent on the application. It is decided to recognize coarse-grained activities like running, walking, standing still etc. Coarse-grained activities are much easier to recognize than fine-grained activities like cooking or shopping. Fine-grained activities need various sensors like GPS or thermometer. We used only the accelerometer in our initial prototype, but later we are hoping to change this approach and use other sensors for recognizing more activities.

The other question is “which features that is needed to be used for building the model?” The feature selection is very important in an AR application because you can’t build an efficient machine learning model for recognition purposes. As we said before, many features must be extracted from the raw sensor data in order to understand the underlying meaning of the raw sensor data. As figure 2.3 indicates, there are many feature types in literature that we can use in our system.



Type	Features
Time-Domain	Mean
	Variance, Std. Dev., Mean Abs. Dev.
	RMS
	Cum. Histogram
	Zero or Mean Crossing Rate
	Derivative
	Peak Count & Amp.
	Sign
Frequency-Domain	Discrete FFT Coef.
	Spectral Centroid
	Spectral Energy
	Spectral Entropy
	Freq. Range Power
Time-Frequency Dom.	Wavelet Coef.
Heuristic Features	SMA
	SVM
	Inter-axis Corr.
Domain-Specific	Time-Domain Gait Detection
	Vertical or Horizontal Acceleration

Figure 2.3. Types of features. (Lockhart et al., 2012)

Time domain and frequency domain features are the prominent features which were generally used in the other applications in literature. It has been observed that their accuracy was fine so that’s why these feature types are used in scope of this project, but these feature types have different characteristics and the selection of these features is an issue that must be overcome. There are some concepts like normalization that must be considered during the feature selection. Normalization is generally used in the applications which use the algorithms that need distance calculation. We used c4.5 decision tree so we didn’t need normalization. Another concept is periodicity; we planned to recognize acts like running and walking. These activities contain periodic actions. Periodicity can be detected in a certain activity by applying auto-correlation functions to its raw data. Auto-correlation can be implemented by applying two times FFT to a signal. Therefore it was rational for using FFT coefficients as a feature (Jun-geun et al., 2012). Finally, the usage of the peak points in a certain data window can also determine the periodic actions. We decided to use both peak points and FFT coefficients for determining the periodicity. Below, you can find rest of the features that we used in our system.

- Mean of accelerometer magnitude
- Variance of accelerometer magnitude
- Time Between Peaks(3)(For each accelerometer axis)
- Standard Deviation(3)(For each accelerometer axis)
- FFT Coefficients(1-20th coefficient)(The number depends on the window size)

2.3. EVALUATION OF HUMAN ACTIVITY CLASSIFIERS

Researchers generally evaluate the performance of the classifiers and sensors for performing an experiment on a system. As you can see from the introduction part of the study, in this section we will evaluate experiments on the classifiers in the literature.

Selection of the activities is important as the selection of the classifiers. The performance of a classifier can differ hugely depending on the activities that we want our system to recognize. In literature, there are many activity groups like locomotion activities which include biking, running, walking etc. or daily activities which include cooking, washing hands etc. The feasible recognition of these activities is very contingent upon the selection of the algorithm.

In this paper (Reddy et al., 2010), authors ran the experiments on the locomotion activities. They used GPS + Accelerometer and they evaluated the performance of several classifiers in terms of precision and recall.

Table 2.1. Precision results for classifiers. (Reddy et al., 2010)

	Still	Walk	Run	Bike	Motor	All
DT	**%95**	**%87,6**	**%95,5**	**%84,5**	**%93,9**	**%91,3**
KMC	%54	%81	%98,5	%45,6	%98,9	%75,6
NB	%88,4	%88,1	%93,5	%75,6	%71,3	%83,4
NN	%96,4	%87,3	%93,3	%84,8	%92,7	%90,9
SVM	%90,7	%88,8	%95,9	%81,6	%97,8	%91
CHMM	%89,2	%90	%94,3	%80,5	%77,6	%86,3
DT-DHMM	**%95,5**	**%92,4**	**%96,4**	**%87,9**	**%96,2**	**%93,7**

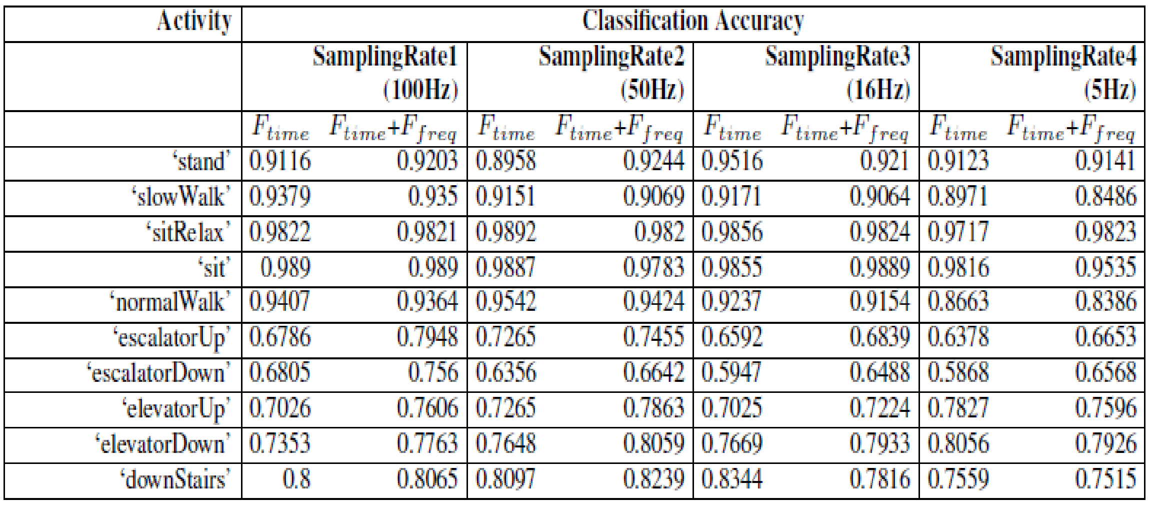
Table 2.2. Recall results for classifiers. (Reddy et al., 2010)

	Still	Walk	Run	Bike	Motor	All
DT	**%97,2**	**%88,4**	**%91,9**	**%85,3**	**%93,4**	**%91,3**
KMC	%99,7	%75,3	%81	%34,8	%63,2	%70,8
NB	%97,2	%77,4	%94,2	%51,2	%95,3	%83
NN	%96,6	%88	%92,9	%84,2	%92,9	%90,9
SVM	%97,4	%86,9	%92,7	%87,1	%89,4	%90,7
CHMM	%97,5	%79	%94,7	%63,5	%95,9	%86,1
DT-DHMM	**%97,8**	**%90,8**	**%94,4**	**%90,6**	**%94,5**	**%93,6**

This system uses the online classification, so as we mentioned before custom classifiers are fabricated for solving the real-time data streaming problem that the online classification caused. As table 2.1 and 2.2 indicate, DT-DHMM classifier performs better than other classification algorithms. The decision tree classifier performs significantly well by itself but nevertheless if you support the decision tree classifier with a different classifier, the performance will increase.

Feature selection is also important in the experiments. As we mentioned in the method section, there are two types of features, time domain features and frequency domain features. In this paper (Yan et al., 2012), authors analyzed the influence of sample rate and feature types to the performance of the system.

Table 2.3. Accuracy of the activity recognition using different feature groups. (Yan et al., 2012)



Activity	Classification Accuracy							
	SamplingRate1 (100Hz)		SamplingRate2 (50Hz)		SamplingRate3 (16Hz)		SamplingRate4 (5Hz)	
	F_{time}	F_{time}+F_{freq}	F_{time}	F_{time}+F_{freq}	F_{time}	F_{time}+F_{freq}	F_{time}	F_{time}+F_{freq}
'stand'	0.9116	0.9203	0.8958	0.9244	0.9516	0.921	0.9123	0.9141
'slowWalk'	0.9379	0.935	0.9151	0.9069	0.9171	0.9064	0.8971	0.8486
'sitRelax'	0.9822	0.9821	0.9892	0.982	0.9856	0.9824	0.9717	0.9823
'sit'	0.989	0.989	0.9887	0.9783	0.9855	0.9889	0.9816	0.9535
'normalWalk'	0.9407	0.9364	0.9542	0.9424	0.9237	0.9154	0.8663	0.8386
'escalatorUp'	0.6786	0.7948	0.7265	0.7455	0.6592	0.6839	0.6378	0.6653
'escalatorDown'	0.6805	0.756	0.6356	0.6642	0.5947	0.6488	0.5868	0.6568
'elevatorUp'	0.7026	0.7606	0.7265	0.7863	0.7025	0.7224	0.7827	0.7596
'elevatorDown'	0.7353	0.7763	0.7648	0.8059	0.7669	0.7933	0.8056	0.7926
'downStairs'	0.8	0.8065	0.8097	0.8239	0.8344	0.7816	0.7559	0.7515

As you can see from Table 2.3, the usage of the both F_{time} and F_{freq} at the same time generally has a better performance than the usage of only F_{time}. F_{time} and F_{freq} are acronyms of time domain features and frequency domain features respectively. You can also see that the increase of the sampling rate increases the classification accuracy.

There is a similar paper (Bao & Intille, 2004) that uses C4.5 decision tree classifier that has a satisfactory performance. Authors proposed a system where the users wear five small biaxial accelerometers. Accelerometer data was collected from 20 subjects. C4.5 decision tree classifier has a performance with a good accuracy rate of %84. 20 different activities are recognized by this system.

Table 2.4. Aggregate recognition rates for activities. (Bao & Intille, 2004)

Activity	Accuracy	Activity	Accuracy
Walking	%89.71	Carrying items	%82.10
Sitting & Relaxing	%94.78	Working on PC	%97.49
Standing still	%95.67	Eating or drinking	%88.67
Watching TV	%77.29	Reading	%91.79
Running	%87.68	Bicycling	%96.29
Stretching	%41.42	Strength-training	%82.51
Scrubbing	%81.09	Vacuuming	%96.41
Folding laundry	%95.14	Lying down	%94.96
Brushing teeth	%85.27	Climbing stairs	%85.61
Riding elevator	%43.48	Riding escalator	%70.56

The systems that have been analyzed previously can recognize 5 or 6 activities max, but it has been observed in table 2.4., the system can recognize 20 activities. This system uses five different accelerometers, each located on different parts of a human body. Therefore, it’s normal that if you increase the diversity of the sensor data (in this case increases the number of the sensors), you can recognize more activities.

Likewise, this paper (Kose et al., 2012) evaluated the performance of the classifier in terms of accuracy, precision, recall and F-measure. This system can recognize four activities and uses the online classification. Results are shown in table 2.5.

Table 2.5. Comparison of clustered KNN, NB and DT without biking. (Kose et al., 2012)

	Clustered KNN	**Naïve Bayes**	**Decision Tree**
Accuracy	%92,13	%75,23	%85,52
Precision	%92,45	%70,08	%83,56
Recall	%92,09	%75,07	%81,22
F-Measure	%92,27	%72,10	%82,37

Authors of this paper also pointed out that they made two different experiments. The accuracy results of running, walking, standing and sitting is above. In their second experiment they added another activity called biking and they performed their experiment again. After the second experiment, they observed and significant decrease on the accuracy results. They tied this observation to activity similarity between biking and running. They confirmed this result by looking at the accelerometer readings. This similarity leaded to misclassification. They also indicated that they can overcome this problem by extracting the frequency domain features since their system used only the time domain features. As it has been observed previous sections of this report, they are proven right. A system can detail the nature of a sensor signal by using the frequency domain.

They also analyzed the performance of the classifier dependent on the sample rate and window size.

Table 2.6. Impact of window size and sampling interval on the accuracy rates of classifiers with biking. (Kose et al., 2012)

Window size (sec.)	0.5			1			2		
Sample Rate	10	50	100	10	50	100	10	50	100
C-KNN	61.6%	59.3%	61.2%	64.9%	66.6%	65.8%	71.5%	**71.7%**	70.7%
DT	64.7%	67.2%	58.0%	63.0%	67.8%	65.1%	68.9%	**76.2%**	71.2%
NB	41.0%	44.4%	43.6%	43.3%	42.9%	43.2%	45.8%	45.7%	**46.1%**

It has been observed that the window size is a more dominant system parameter than the sampling rates (Kose et al., 2012). In general, bigger window sizes obtain better results regardless of the significant effects of the sampling rates. In addition to that, smaller sample rates revealed better results. Generally, increasing the sampling rates should be detailed the nature of the activities, yet for this experiment it created a negative effect on the results (Kose et al., 2012). You can see from Figure 2.6, the accuracy rates of 50 Hz are better than the 100 Hz.

3 – EXPLORATORY ACTIVITY RECOGNITION PHASE

The design and implementation of an effective AR system need to be informed by a holistic understanding of machine learning models. We started the modelling process by discussing several considerations in model development, such as purpose of our study, the availability of the knowledge of smartphone orientation and feasible human activities which were recognizable by wearable sensors.

In this chapter, we presented an exploratory phase towards the activity recognition. We ran some experiments in order to take a closer look to elements of online activity recognition systems.

The design and implementation of an effective AR system need to be informed by a holistic understanding of machine learning models. We started the modelling process by discussing several considerations in model development, such as purpose of our study, the availability of the knowledge of smartphone orientation and feasible human activities which were recognizable by wearable sensors.

In this chapter, we presented an exploratory phase towards the activity recognition. We ran some experiments in order to take a closer look to elements of online activity recognition systems.

3.1. MACHINE LEARNING MODELS

Machine learning is a type of artificial intelligence that provides the ability to learn without being explicitly programmed to the computers. This ability basically allows computers to handle new situations via analysis, self-training, observation and experience (Domingos, 2012). Therefore we can ask ourselves the following question “How can we create systems to automatically learn and to improve with more data?” The learning in this context is recognizing complex patterns and makes intelligent decisions based on data. The difficulty lies in here that the set of all possible decisions given all possible inputs is too complex for us to understand (Domingos, 2012). The field of Machine Learning develops algorithms that discover knowledge and some patterns from specific data and experience, based on sound statistical and computational principles (Domingos, 2012). An activity recognition system uses these algorithms in order to recognize the activities. In this study we used Decision Trees and Random Forest as machine learning algorithms.

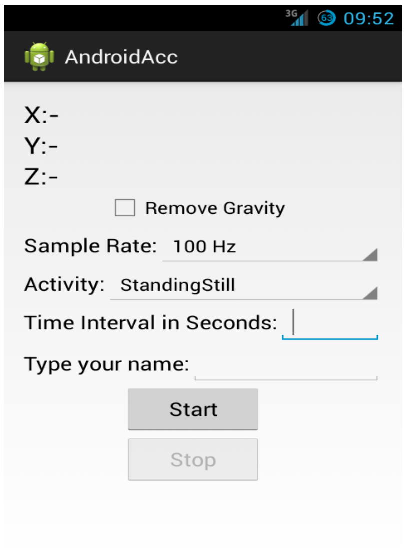
A decision tree is a graph that uses a branching method to illustrate every possible outcome of a decision. Therefore we can say that decision trees are graphical representations of a classification process. A classical DT is consisted of 3 parts; ”nodes” represent a test on an attribute, “link” represents a possible value for the tested attribute and ”leafs” are estimated class for the considered instance. The general algorithm of the decision trees are based on creating purer and purer instance subsets recursively. Therefore the recursive process continues until all instances in the subset belong to the same class. Steps of the DT algorithm contain initializing of the tree, creating a new node and selecting an attribute for this node. The selection of the attribute is the most important step. There various selection criterias are available but the most common one is the information gain. Information gain is a measure based on information theory principles and DTs learn the decision boundary by recursively partitioning the space in a manner that maximizes this information gain. The concept of entropy is used to calculate the information gain. Entropy basically measures the average quantity of information over the different possible realizations, weighted by their probability. We used C4.5 decision tree which is developed by Ross Quinlan (Quinlan, 1986). This version of the decision trees differs from classical one by its ability to handle continuous data and enables the pruning after the creation of tree.

Random forests are an ensemble learning method for classification that operate by constructing multiple decision trees at training time and outputting the class that is the mode of the classes output by individual trees[1]. Basically, a classification of a instance is made by all trees in the forests and Leo Breiman and Adele Cutler who are the creators of Random Forests call this process “Voting”. The classification process is finalized by the forest chooses the class that has the most votes. The parameters such as ”number of trees” and ”number of tried attributes” must be set before applying the algorithm. The general algorithm of Random Forests is based on bootstrap aggregating. Bootstrap aggregating is a machine learning meta-algorithm for improving the accuracy of the machine learning algorithm by using the resampling with replacement. It basically creates multiple datasets from one dataset. Bootstrap aggregating, also known as “Bagging”, is commonly used in the field of machine learning, especially in decision trees. Random Forests creating a fixed number of bootstrap subsets from original training data by using bootstrap aggregating. The algorithms start with the dispersion of original data into two. About two-third of the data is used for constructing each tree of the forest by using a different bootstrap subset from the original data. Remaining training data are used to estimate error and variable importance. At the each node of the trees, a randomly selected subset of features is used for splitting. Finally the class assignment of is made by the number of votes from all of the trees.

3.2. MODEL BUILDING AND TESTING

Before getting into the machine learning models we must understand the concept of “model” in this context. The process of activity recognition is some kind of supervised classification and it is known that the supervised classification algorithms use the models for classifying the instances. Like in every supervised classification process, data needs to be labelled. Labelled data is also known as training data and training data refers to the data used for “building the model”. The training data is an important requirement for the application; therefore we implemented simple android application called “AndroidAcc”. This application is used for logging the user’s accelerometer readings. AnroidAcc has a very simple user interface.

[1] Random Forests. URL: http://stat-www.berkeley.edu/users/breiman/RandomForests/. [accessed June 2014]



09:52

AndroidAcc

X:-

Y:-

Z:-

☐ Remove Gravity

Sample Rate: 100 Hz

Activity: StandingStill

Time Interval in Seconds:

Type your name:

Start

Stop

Figure 3.1. Main Screen of AndroidAcc

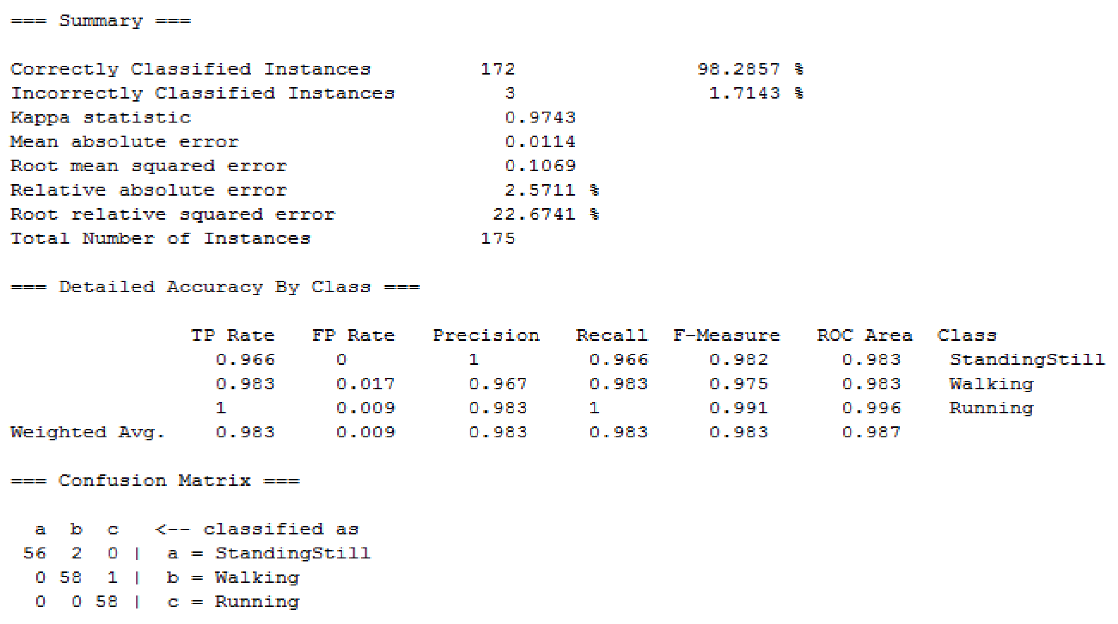
The user chooses the time interval, the sample rate and the activity that the user is about to perform from the graphic interface. When the user presses the start button, the application gives “Five seconds” of preparation time for the user to put the smartphone in their pocket and perform the action. The application logs the data to csv text file.

We collected sensor data from single person that consisted of five different activities with 100Hz sample rate. Each activity was performed for 60 seconds. When we merged the data of five activities with one second window size, we had a raw labeled accelerometer data with 291 instances. We used WEKA program[2] for the model building. As we said at the beginning, we used c4.5 decision tree as the supervised classification algorithm. The usage of WEKA is simple, but there are some settings which must be set. The most important setting is the count of folds of the cross-validation. The cross-validation is a model validation technique. It is used to estimate how accurately a machine learning model will perform in action. In practice, the training set is used for building models but in addition to that extra data which is generally called test data must be used for validating the model. Validation of the model

[2] Machine Learning Group at University of Waikato, (2012). Weka Machine Learning Toolkit. URL: http://www.cs.waikato.ac.nz/ml/index.html[accessed February 2014]

is crucial for an efficient machine learning model. The collection of the test data is another challenge and one of the main reasons that we used cross-validation in our system, is to overcome this challenge. This is simply a way of coming up with partition of your entire dataset into training and test data. For example, if you do 10-fold cross-validation, your entire dataset is partitioned into 10 sets of equal parts. Nine of these are combined and used for training; the remaining one is used for testing. This process is repeated with nine different sets combined for training and so on until all of the ten individuals have been used for training. Therefore we say that training/test sets and cross-validation are conceptually doing the same thing; cross-validation simply takes a more rigorous approach by averaging over the entire dataset.

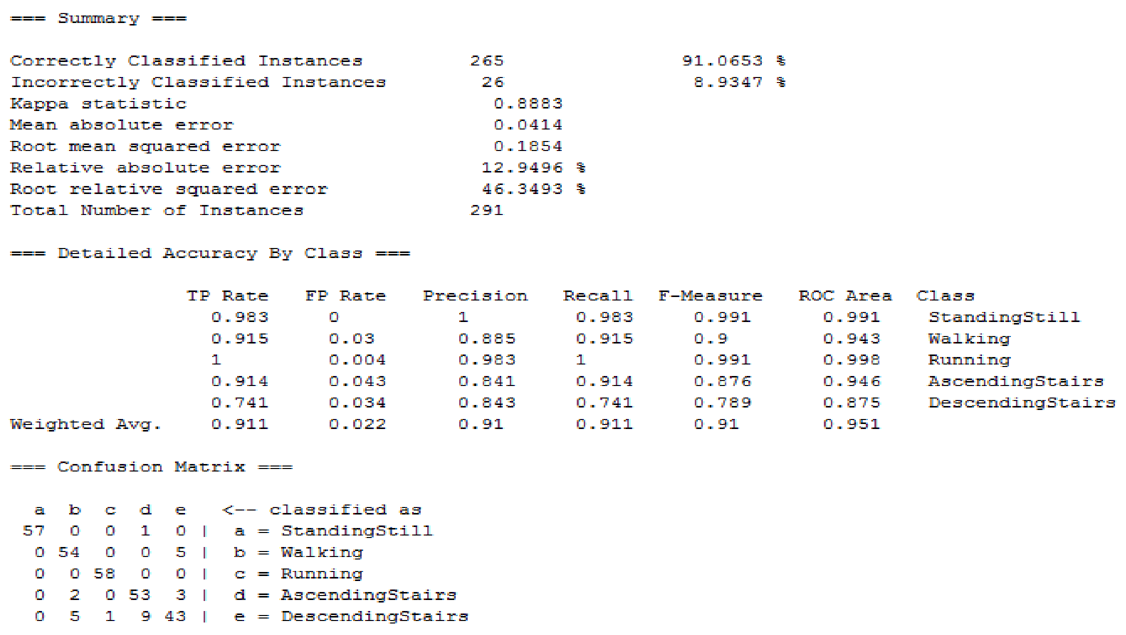
In our initial tests, we used 10-fold cross-validation. Our first tests without FFT features were split into two. In the first one, we left out the stairs data. You can see the results in Figure 3.2.



```
=== Summary ===

Correctly Classified Instances         172               98.2857 %
Incorrectly Classified Instances         3                1.7143 %
Kappa statistic                          0.9743
Mean absolute error                      0.0114
Root mean squared error                  0.1069
Relative absolute error                  2.5711 %
Root relative squared error             22.6741 %
Total Number of Instances              175

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.966     0          1         0.966     0.982      0.983    StandingStill
                 0.983     0.017      0.967     0.983     0.975      0.983    Walking
                 1         0.009      0.983     1         0.991      0.996    Running
Weighted Avg.    0.983     0.009      0.983     0.983     0.983      0.987

=== Confusion Matrix ===

  a  b  c   <-- classified as
 56  2  0 |  a = StandingStill
  0 58  1 |  b = Walking
  0  0 58 |  c = Running
```

Figure 3.2. Test results of the data without stairs

In the second one, we added the stairs data and ran another test.

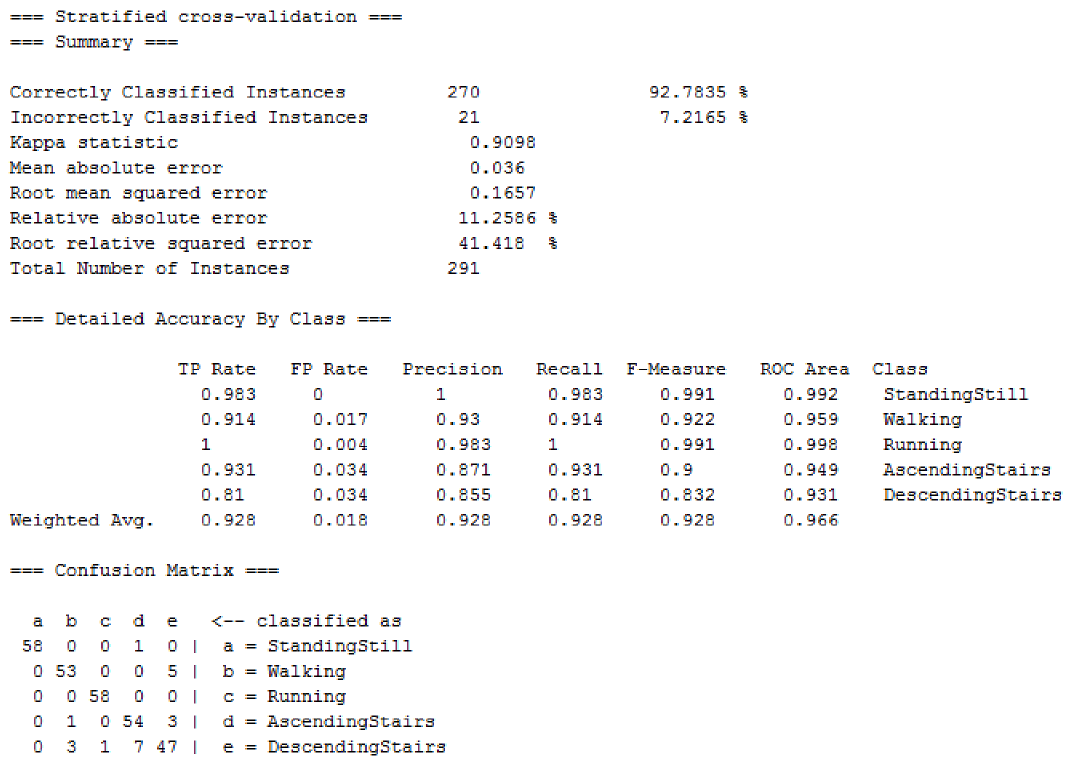


```
=== Summary ===

Correctly Classified Instances         265               91.0653 %
Incorrectly Classified Instances        26                8.9347 %
Kappa statistic                          0.8883
Mean absolute error                      0.0414
Root mean squared error                  0.1854
Relative absolute error                 12.9496 %
Root relative squared error             46.3493 %
Total Number of Instances              291

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.983     0          1         0.983     0.991      0.991    StandingStill
                 0.915     0.03       0.885     0.915     0.9        0.943    Walking
                 1         0.004      0.983     1         0.991      0.998    Running
                 0.914     0.043      0.841     0.914     0.876      0.946    AscendingStairs
                 0.741     0.034      0.843     0.741     0.789      0.875    DescendingStairs
Weighted Avg.    0.911     0.022      0.91      0.911     0.91       0.951

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 57  0  0  1  0 |  a = StandingStill
  0 54  0  0  5 |  b = Walking
  0  0 58  0  0 |  c = Running
  0  2  0 53  3 |  d = AscendingStairs
  0  5  1  9 43 |  e = DescendingStairs
```

Figure 3.3. Test results of the data with stairs

As you can see from the results in figure 3.2 and 3.3, when we added the stairs data, the accuracy of the classification decreased. It has been observed that the stairs activity especially the "DescendingStairs" decreases the accuracy by observing the confusion matrix from the results. It is a normal thing since the similarity between stairs and walking activities is indisputable. We proved this with these tests.

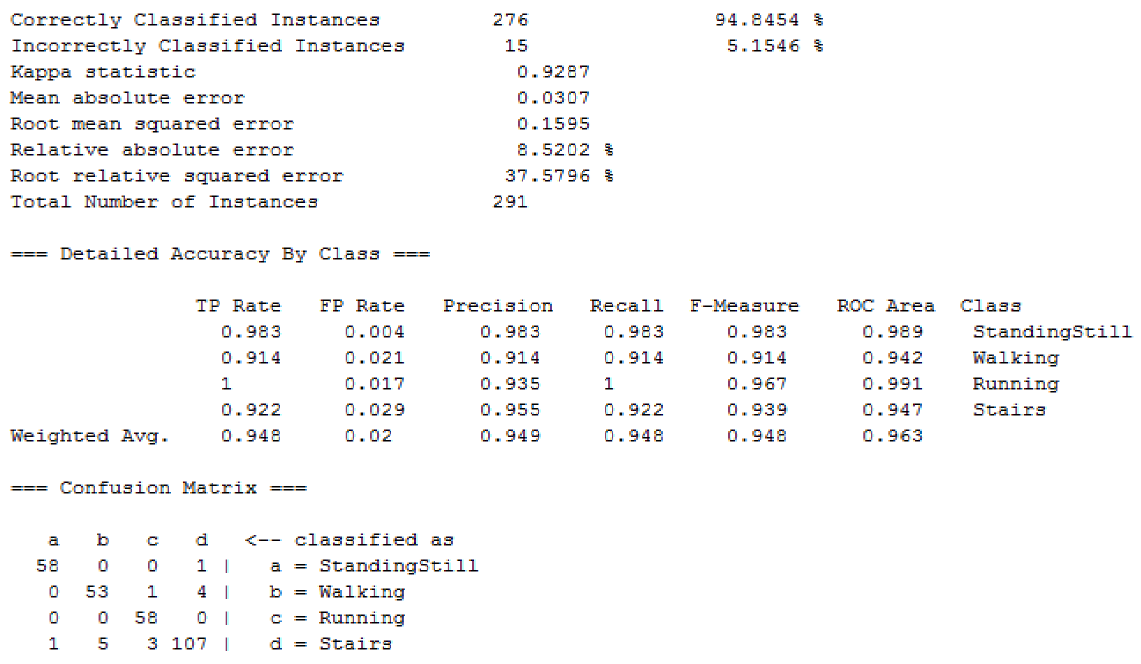
For the last test we added FFT features to the feature set that we used in our dataset.



```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         270               92.7835 %
Incorrectly Classified Instances        21                7.2165 %
Kappa statistic                          0.9098
Mean absolute error                      0.036
Root mean squared error                  0.1657
Relative absolute error                 11.2586 %
Root relative squared error             41.418  %
Total Number of Instances              291

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.983     0           1         0.983     0.991      0.992    StandingStill
                 0.914     0.017       0.93      0.914     0.922      0.959    Walking
                 1         0.004       0.983     1         0.991      0.998    Running
                 0.931     0.034       0.871     0.931     0.9        0.949    AscendingStairs
                 0.81      0.034       0.855     0.81      0.832      0.931    DescendingStairs
Weighted Avg.    0.928     0.018       0.928     0.928     0.928      0.966

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 58  0  0  1  0 |  a = StandingStill
  0 53  0  0  5 |  b = Walking
  0  0 58  0  0 |  c = Running
  0  1  0 54  3 |  d = AscendingStairs
  0  3  1  7 47 |  e = DescendingStairs
```

Figure 3.4. Test results of the data with FFT features

We observed an increase of 1 point to %92.78 but this increase didn’t satisfy us. When we examined the confusion matrix, we were able to see that there were some misclassifications between the stairs activities. Hence, we merged the stairs activities into one activity called “Stairs”. In other words, “AscendingStairs” and “DescendingStairs” were merged into “Stairs” activity. We ran another test after this merger.



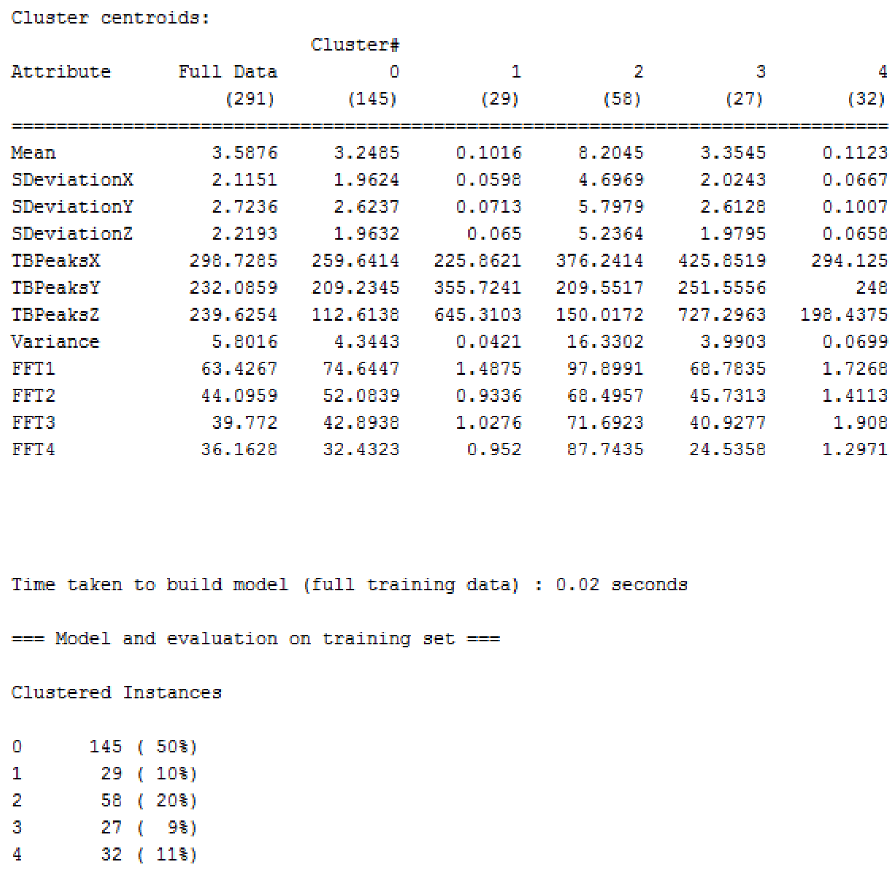
```
Correctly Classified Instances         276               94.8454 %
Incorrectly Classified Instances        15                5.1546 %
Kappa statistic                          0.9287
Mean absolute error                      0.0307
Root mean squared error                  0.1595
Relative absolute error                  8.5202 %
Root relative squared error             37.5796 %
Total Number of Instances              291

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.983     0.004      0.983     0.983     0.983      0.989    StandingStill
                 0.914     0.021      0.914     0.914     0.914      0.942    Walking
                 1         0.017      0.935     1         0.967      0.991    Running
                 0.922     0.029      0.955     0.922     0.939      0.947    Stairs
Weighted Avg.    0.948     0.02       0.949     0.948     0.948      0.963

=== Confusion Matrix ===

   a   b   c   d   <-- classified as
  58   0   0   1 |   a = StandingStill
   0  53   1   4 |   b = Walking
   0   0  58   0 |   c = Running
   1   5   3 107 |   d = Stairs
```

Figure 3.5. Test results of the data with “Stairs” activity

The accuracy of the classifier was increased by 2.8 points to %94.8454. This is a very good accuracy but it can be much better since when we merged the stairs activities, the instance count of the “Stairs” activity was two time bigger than the other activities. This could cause an overfitting problem. But this accuracy is sufficient for us in this point.

3.3. CLUSTERING APPROACH

The results in the previous section were successful, but it has been displayed that there are some misclassifications. As we said before, the activities like walking, ascending stairs and descending stairs are similar, therefore this means they had some similarities between their numerical feature values. It has been decided to make a cluster analysis in my dataset for examining these similarities and detecting the features which cause the misclassifications. For these purposes, we used a distance based clustering algorithm called “Simple K-Means”. K-means is a clustering algorithm that uses the Euclidean distance. The process of the algorithm follows an easy way to classify a given data set through a certain number of clusters (K clusters) fixed a priori. The main idea is to define “K” centroids, one for each cluster, and create clusters based on the distances between these centroids and other instances. Then the algorithm defines a new centroid based on its cluster members and creates new clusters again. This process continues till fixed amount of iterations. WEKA has its own k-means implementation hence we decided to run the tests on WEKA again. This algorithm needs the cluster count at the beginning; we had five activities so we determined the cluster count as five. The result of the first test is below.



```
Cluster centroids:
                           Cluster#
Attribute       Full Data          0          1          2          3          4
                    (291)      (145)       (29)       (58)       (27)       (32)
================================================================================
Mean               3.5876     3.2485     0.1016     8.2045     3.3545     0.1123
SDeviationX        2.1151     1.9624     0.0598     4.6969     2.0243     0.0667
SDeviationY        2.7236     2.6237     0.0713     5.7979     2.6128     0.1007
SDeviationZ        2.2193     1.9632      0.065     5.2364     1.9795     0.0658
TBPeaksX         298.7285   259.6414   225.8621   376.2414   425.8519    294.125
TBPeaksY         232.0859   209.2345   355.7241   209.5517   251.5556        248
TBPeaksZ         239.6254   112.6138   645.3103   150.0172   727.2963   198.4375
Variance           5.8016     4.3443     0.0421    16.3302     3.9903     0.0699
FFT1              63.4267    74.6447     1.4875    97.8991    68.7835     1.7268
FFT2              44.0959    52.0839     0.9336    68.4957    45.7313     1.4113
FFT3               39.772    42.8938     1.0276    71.6923    40.9277      1.908
FFT4              36.1628    32.4323      0.952    87.7435    24.5358     1.2971



Time taken to build model (full training data) : 0.02 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       145 ( 50%)
1        29 ( 10%)
2        58 ( 20%)
3        27 (  9%)
4        32 ( 11%)
```

Figure 3.6. Test results of K-means cluster analysis

The results were not satisfying, as you see the distribution must be %20 for each cluster since we had same amount of instances for each activity. This test gave us nothing in the terms of feature examination.

After some research, we realized that the separate clustering could work better. In this case, separate clustering means that testing the activities in dual or triple groups. First we tested three activities “Running, Walking and StandingStill”.

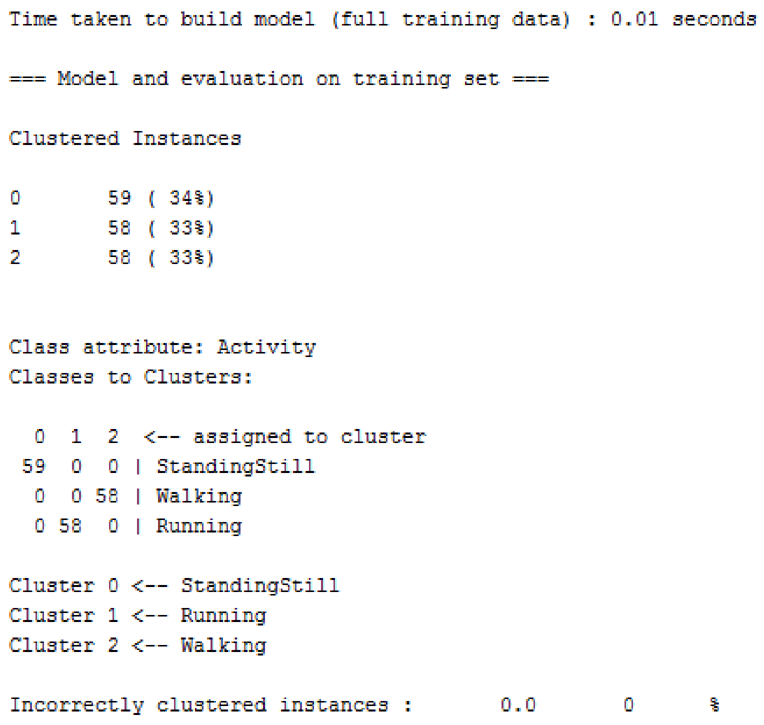
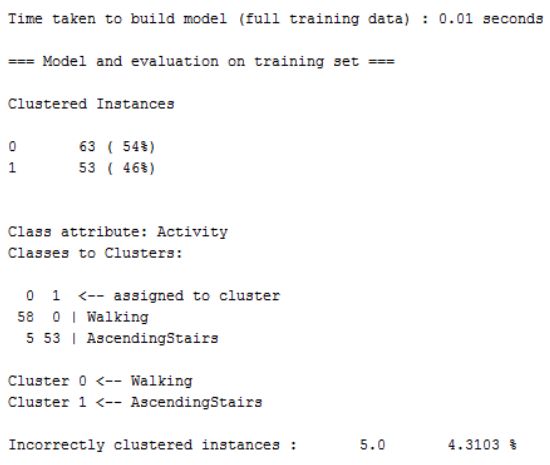
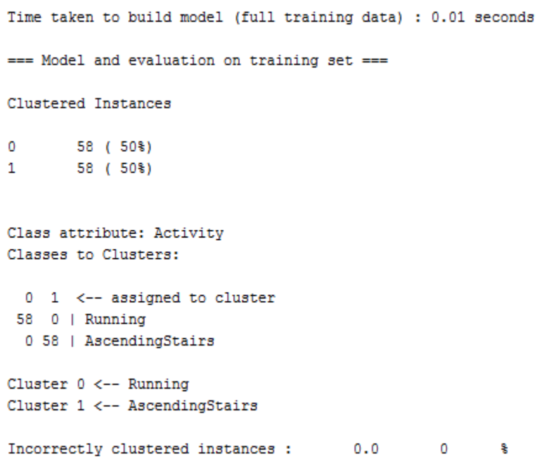

```
Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       59 ( 34%)
1       58 ( 33%)
2       58 ( 33%)


Class attribute: Activity
Classes to Clusters:

  0  1  2  <-- assigned to cluster
 59  0  0 | StandingStill
  0  0 58 | Walking
  0 58  0 | Running

Cluster 0 <-- StandingStill
Cluster 1 <-- Running
Cluster 2 <-- Walking

Incorrectly clustered instances :       0.0       0      %
```

Figure 3.7. Test results of K-means cluster analysis with 3 activities

It can be seen that there is no misclustering, this means there are no similarities between their features of these activities. Then we started to run dual groups tests.

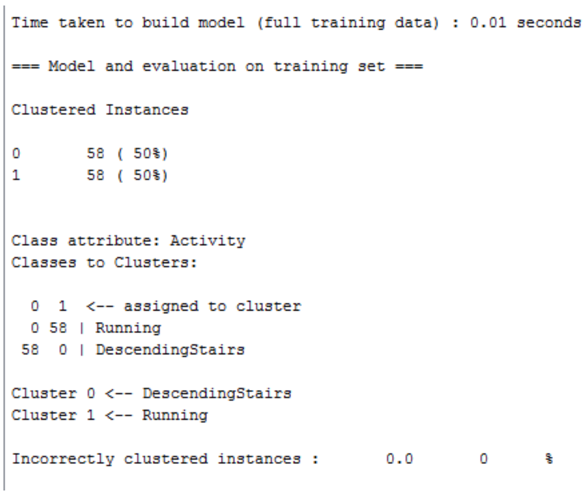


```
Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       63 ( 54%)
1       53 ( 46%)


Class attribute: Activity
Classes to Clusters:

  0  1  <-- assigned to cluster
 58  0 | Walking
  5 53 | AscendingStairs

Cluster 0 <-- Walking
Cluster 1 <-- AscendingStairs

Incorrectly clustered instances :       5.0       4.3103 %
```

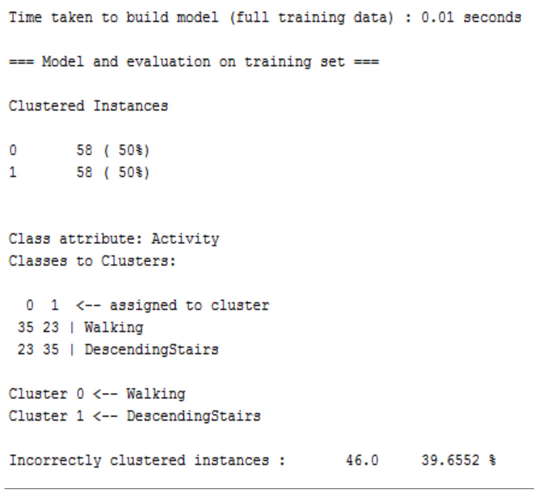


```
Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       58 ( 50%)
1       58 ( 50%)


Class attribute: Activity
Classes to Clusters:

  0  1  <-- assigned to cluster
 58  0 | Running
  0 58 | AscendingStairs

Cluster 0 <-- Running
Cluster 1 <-- AscendingStairs

Incorrectly clustered instances :       0.0       0      %
```

Figure 3.8. Test results of AscendingStairs-Walking and AscendingStairs-Running

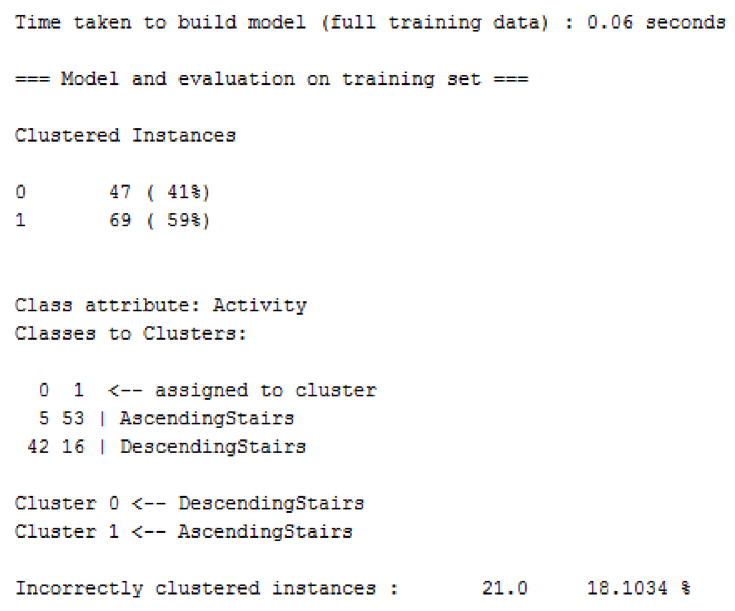


```
Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       58 ( 50%)
1       58 ( 50%)


Class attribute: Activity
Classes to Clusters:

  0  1  <-- assigned to cluster
  0 58 | Running
 58  0 | DescendingStairs

Cluster 0 <-- DescendingStairs
Cluster 1 <-- Running

Incorrectly clustered instances :       0.0       0      %
```



```
Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       58 ( 50%)
1       58 ( 50%)


Class attribute: Activity
Classes to Clusters:

  0  1  <-- assigned to cluster
 35 23 | Walking
 23 35 | DescendingStairs

Cluster 0 <-- Walking
Cluster 1 <-- DescendingStairs

Incorrectly clustered instances :       46.0     39.6552 %
```

Figure 3.9. Test results of DescendingStairs-Running and DescendingStairs-Walking

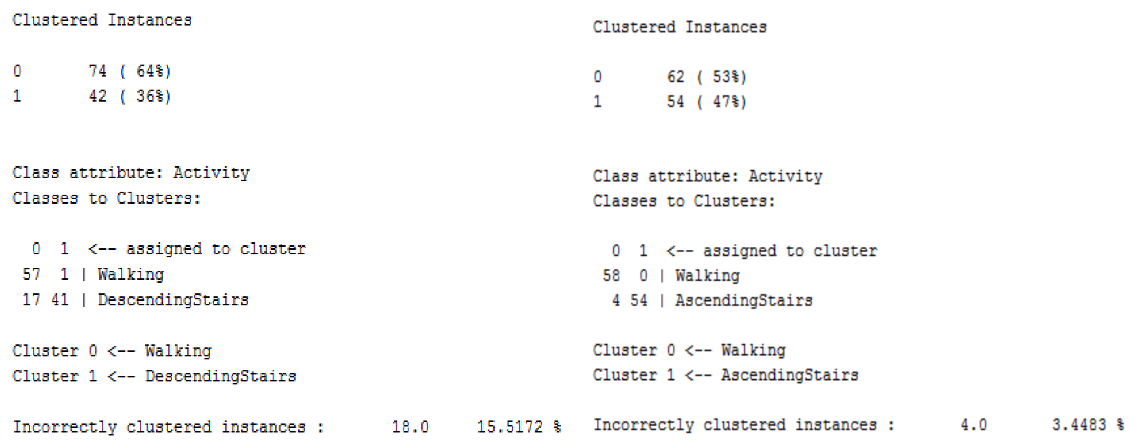


```
Time taken to build model (full training data) : 0.06 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       47 ( 41%)
1       69 ( 59%)


Class attribute: Activity
Classes to Clusters:

  0  1  <-- assigned to cluster
  5 53 | AscendingStairs
 42 16 | DescendingStairs

Cluster 0 <-- DescendingStairs
Cluster 1 <-- AscendingStairs

Incorrectly clustered instances :       21.0     18.1034 %
```

Figure 3.10. Test results of DescendingStairs-AscendingStairs

It has been observed that the outcome was consisted as with earlier predictions, the clustering groups of AscendingStairs-Walking, DescendingStairs-Walking and DescendingStairs-AscendingStairs have some instances of misclustering. This means that there are some similarities between the features of these activities and these similarities cause the misclassification in our model building and testing. A way to increase the accuracy of this cluster analysis was needed to be found.

We had 12 features in our dataset and each feature had its own importance. There are some feature reduction methods in literature that also increases the accuracy of a system. It is known that k-means uses the Euclidean distance calculation and the distance calculation between the less important features can cause a misclustering, therefore we decided to make a feature selection in a set of features and observed the results. There are several methods that we can use like SFS or SBS but instead of implementing these complex methods, we thought to implement a very simple approach. In the previous section, we used c4.5 decision tree as the classification algorithm. Every implementation of a decision tree uses entropy for building the classification model. In this case the model is a tree and simply the root of the tree must be feature that gives us the most information about the dataset than the others. In other words, features are ranked in the tree based on their information potential. When we examined our tree, it has been seen that there are six features took part in the tree instead of 12. This means our classification model uses only six of the features. Thusly we made another test, but this time we used only six of the features. We observed that there are some improvements in certain groups in figure 3.11.



```
Clustered Instances

0       74 ( 64%)
1       42 ( 36%)


Class attribute: Activity
Classes to Clusters:

  0  1  <-- assigned to cluster
 57  1 | Walking
 17 41 | DescendingStairs

Cluster 0 <-- Walking
Cluster 1 <-- DescendingStairs

Incorrectly clustered instances :       18.0     15.5172 %
```

```
Clustered Instances

0       62 ( 53%)
1       54 ( 47%)


Class attribute: Activity
Classes to Clusters:

  0  1  <-- assigned to cluster
 58  0 | Walking
  4 54 | AscendingStairs

Cluster 0 <-- Walking
Cluster 1 <-- AscendingStairs

Incorrectly clustered instances :       4.0      3.4483 %
```

Figure 3.11. Test results of DescendingStairs-Walking and AscendingStairs-Walking

3.4. ORIENTATION/POSITION RECOGNITION

In the previous sections, it has been observed that an activity recognition system can be implemented with good accuracy by using raw accelerometer data. The dataset that the model is built on and ran the tests was collected by a smartphone which is located in a pants pocket. However, one of the major problems in the activity recognition systems during everyday life is that peoples carry their smartphones in various places like bag or jacket pockets. The model that we created in the previous sections was just for the pants pocket. So this model isn’t applicable when the smartphones is carried in different orientations. For solving this issue, orientation recognition functionality added to our AR system. In addition, we had to log raw accelerometer data in various orientations for building the models in each orientation. In other words, we had to collect every activity in each orientation for building the new models. But this collection of data is very challenging, so we used a new dataset (Ustev et al., 2013) from WiSe lab (Wireless Sensor Networks Research Group) from Bogazici University, Turkey. This dataset contains data from 10 people (five male, five female) in five different activities with 4 different orientations.

Today's smartphones have a sensor called orientation sensor. This sensor is actually not a hardware sensor. This sensor uses the combination of the accelerometer and gravity sensor. You can get azimuth, pitch and roll data with this sensor.

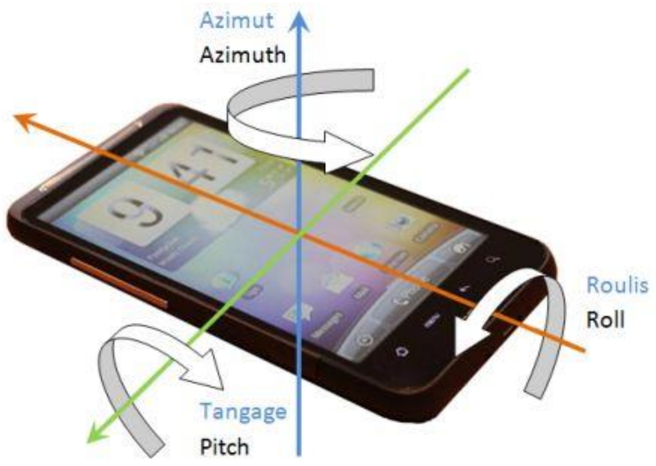


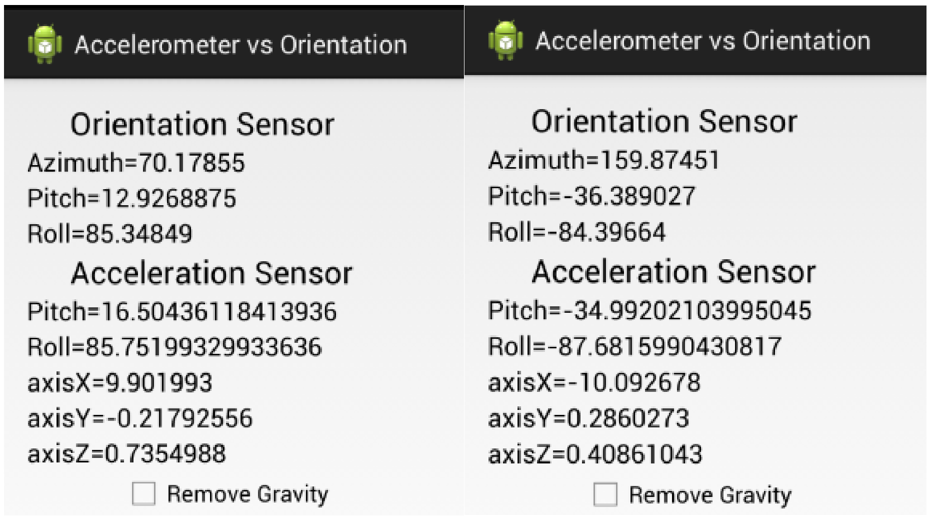
Figure 3.12. Orientation axis of a smartphone

This sensor can be used in our AR system for orientation recognition functionality but this means an addition of a new sensor to our system. This system is needed to be implemented in energy efficient way and because of that as few sensors as possible must be used. Also this sensor was deprecated in Android 2.2(API level 8). After some research, a way to calculate the pitch and the roll values by using the accelerometer was found. Roll and pitch values are sufficient for the orientation recognition since the azimuth value isn't needed for the model.

$$\beta = \frac{180}{\pi} . \tan^{-1}(y/g, z/g) \qquad (1)$$

$$\alpha = \frac{180}{\pi} . \tan^{-1}(x/g, z/g) \qquad (2)$$

We created a simple program that displayed the results of the orientation sensor and calculated pitch and roll values by using (1) and (2). You can find the readings in various phone positions in Figure 3.13.



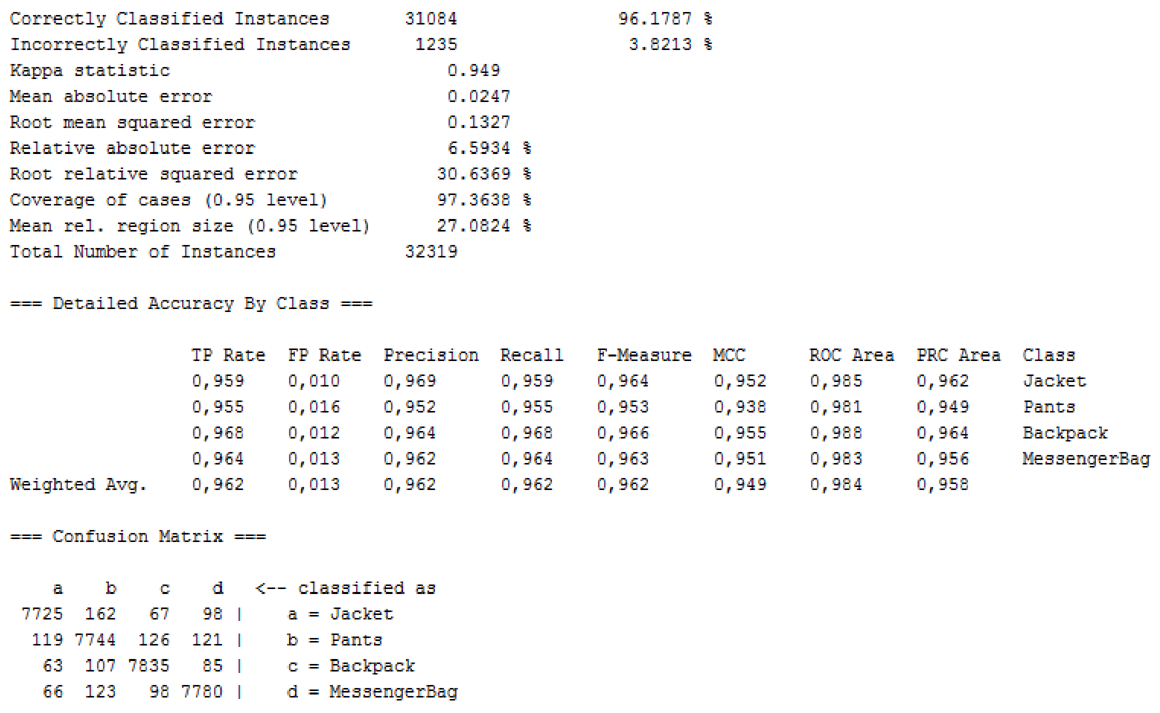
```
Accelerometer vs Orientation
Orientation Sensor
Azimuth=70.17855
Pitch=12.9268875
Roll=85.34849
Acceleration Sensor
Pitch=16.50436118413936
Roll=85.75199329933636
axisX=9.901993
axisY=-0.21792556
axisZ=0.7354988
Remove Gravity
```

```
Accelerometer vs Orientation
Orientation Sensor
Azimuth=159.87451
Pitch=-36.389027
Roll=-84.39664
Acceleration Sensor
Pitch=-34.99202103995045
Roll=-87.6815990430817
axisX=-10.092678
axisY=0.2860273
axisZ=0.40861043
Remove Gravity
```

Figure 3.13. Orientation readings of the both sensor

As you see from the figures, the values were close. There are some differences but these differences weren’t effect the recognition.

After these tests that validate the formula, we built a dataset from the raw dataset. The features that we used in this data were pitch and roll values and their standard deviation in one second window. Our final dataset had four different orientation (Pants pocket, backpack bag pocket, messenger bag pocket and jacket pocket) and 32319 instances. We applied c4.5 decision tree algorithm to this dataset and results are in figure 3.14.



```
Correctly Classified Instances       31084               96.1787 %
Incorrectly Classified Instances      1235                3.8213 %
Kappa statistic                          0.949
Mean absolute error                      0.0247
Root mean squared error                  0.1327
Relative absolute error                  6.5934 %
Root relative squared error             30.6369 %
Coverage of cases (0.95 level)          97.3638 %
Mean rel. region size (0.95 level)      27.0824 %
Total Number of Instances            32319

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,959    0,010    0,969      0,959    0,964      0,952    0,985     0,962     Jacket
                 0,955    0,016    0,952      0,955    0,953      0,938    0,981     0,949     Pants
                 0,968    0,012    0,964      0,968    0,966      0,955    0,988     0,964     Backpack
                 0,964    0,013    0,962      0,964    0,963      0,951    0,983     0,956     MessengerBag
Weighted Avg.    0,962    0,013    0,962      0,962    0,962      0,949    0,984     0,958

=== Confusion Matrix ===

    a    b    c    d   <-- classified as
 7725  162   67   98 |    a = Jacket
  119 7744  126  121 |    b = Pants
   63  107 7835   85 |    c = Backpack
   66  123   98 7780 |    d = MessengerBag
```

Figure 3.14. Test results of orientation recognition

The results were very positive. These satisfying results show us the orientation recognition can be done by using only the accelerometer. There is no need for an extra sensor for this purpose and this is going to save plausible amount of energy in the AR system.

In “Model Building and Testing” section, it has been seen that the orientation and activity recognition can be implemented efficiently by using the accelerometer sensor. We also run some clustering tests for examining the relation between the activities and detecting the similar ones. In addition to that, a simple feature reduction technique was applied and its effect was examined. In the next section, these models and test will be carried out in Android environment.

3.5. ANDROID IMPLEMENTATION

After the implementation on the PC like model building and testing with WEKA, the implementations were carried out in Android platform, but there were some issues that are need to be solved.

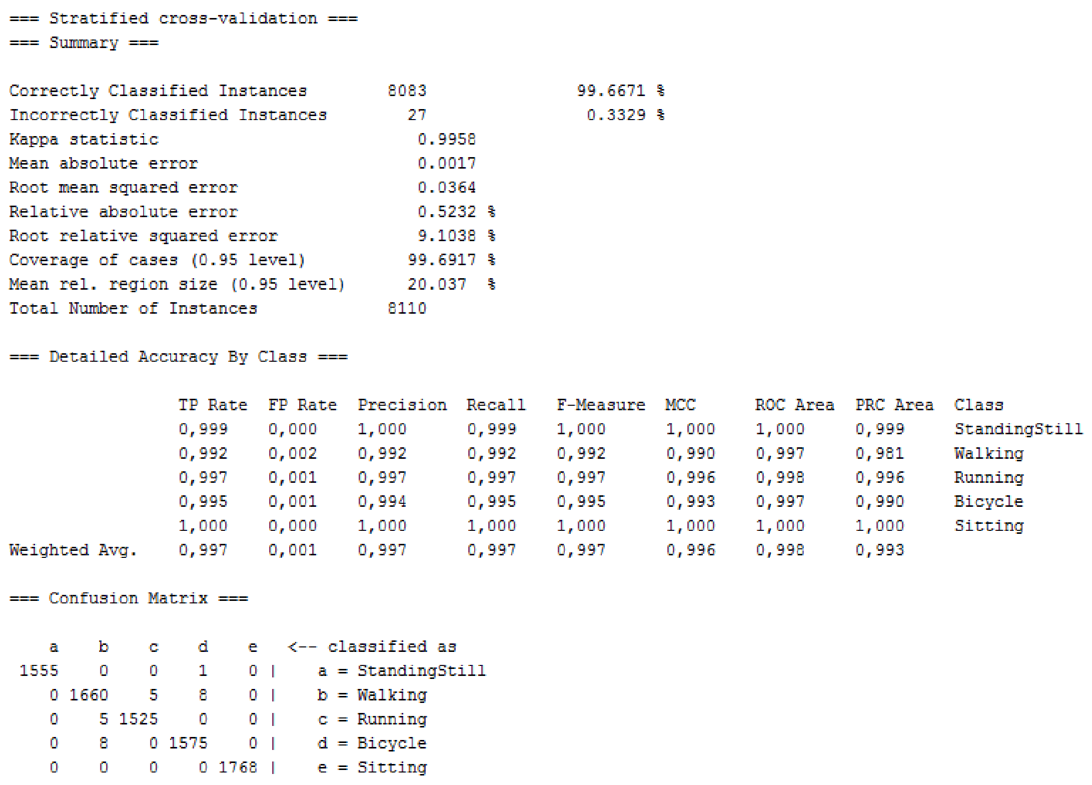
First of this issues was an implementation of a machine learning algorithm to Android platform. At initial implementations, we used c4.5 decision tree and we were very satisfied with its performance so we decided to use this algorithm on Android. Formally, WEKA on the PC for both model building and testing was used, WEKA is an open-source program; therefore their source code are available publicly. WEKA and Android platform are both implemented by JAVA programming language; hence it can possible to port any machine learning implementation of WEKA to Android platform. Initially, it was decided to build our implementation of machine algorithms but at the later stages as similar open-source project[3] was found. They ported all of WEKA algorithms to Android and created a jar library that was compatible with Android environment. It was decided to use this library.

Another issue was the usage of the model. At PC, it is possible to create a model on WEKA by using a dataset, save it and even reuse it another time. We had to be able to use this reusable model on Android since WEKA and Android both use the same programming language but the results were contradicting. The model creation and saving it on WEKA is some kind of serialization of a file and serialization of a file depends on certain things like computer environment, operating system etc. In other words, you can’t just use the model which is created on PC with WEKA on Android platform. Hence, our dataset must be carried out in Android smartphone and the model building must be implemented on the smartphone.

In previous section, it has been observed that we had new dataset which had more instances than ours. This dataset contained 5 different activities (Running, Walking,

[3] WEKA for Android. URL: https://github.com/rjmarsan/Weka-for-Android [accessed February 2014]

StandingStill, Biking, Sitting) with 4 different orientations (jacket, messenger bag, backpack, pants) from 10 different people (5 male, 5 female). For the implementation in this report, only the data with pants pocket orientation was used. Before using the dataset on Android, a test on WEKA was ran in order to examine the accuracy. The result was in figure 3.15.

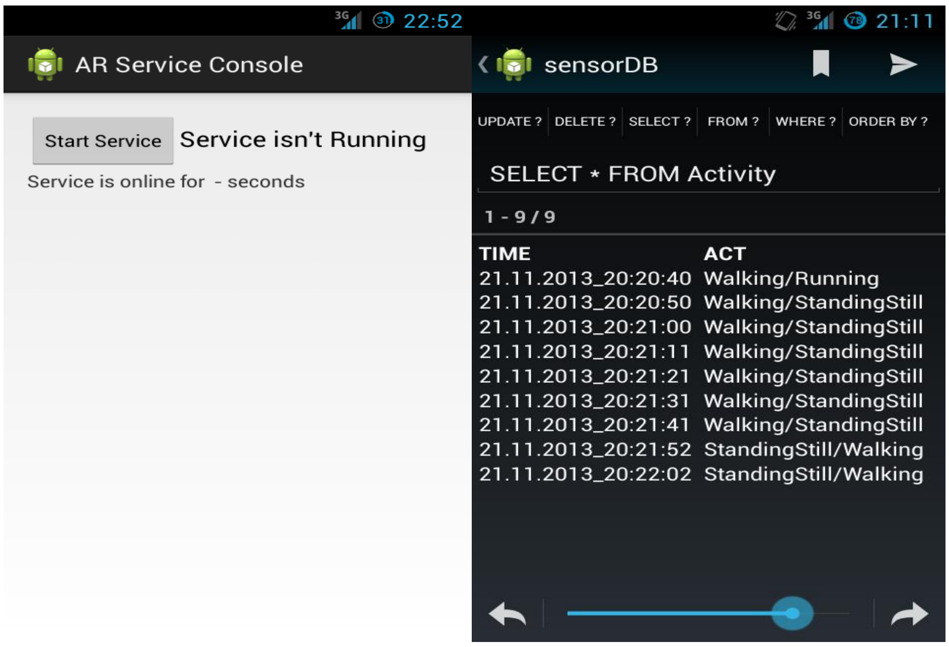


```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        8083               99.6671 %
Incorrectly Classified Instances        27                0.3329 %
Kappa statistic                          0.9958
Mean absolute error                      0.0017
Root mean squared error                  0.0364
Relative absolute error                  0.5232 %
Root relative squared error              9.1038 %
Coverage of cases (0.95 level)          99.6917 %
Mean rel. region size (0.95 level)      20.037  %
Total Number of Instances             8110

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,999    0,000    1,000      0,999    1,000      1,000    1,000     0,999     StandingStill
                 0,992    0,002    0,992      0,992    0,992      0,990    0,997     0,981     Walking
                 0,997    0,001    0,997      0,997    0,997      0,996    0,998     0,996     Running
                 0,995    0,001    0,994      0,995    0,995      0,993    0,997     0,990     Bicycle
                 1,000    0,000    1,000      1,000    1,000      1,000    1,000     1,000     Sitting
Weighted Avg.    0,997    0,001    0,997      0,997    0,997      0,996    0,998     0,993

=== Confusion Matrix ===

    a    b    c    d    e   <-- classified as
 1555    0    0    1    0 |    a = StandingStill
    0 1660    5    8    0 |    b = Walking
    0    5 1525    0    0 |    c = Running
    0    8    0 1575    0 |    d = Bicycle
    0    0    0    0 1768 |    e = Sitting
```

Figure 3.15. Test results of the new dataset with pants pocket orientation

It has been displayed that the results were satisfying for carrying the dataset to smartphone since the accuracy of the classifier was almost perfect.

The model building on Android can be made by a JAVA code. WEKA has a perfect Javadoc for these purposes. How to build a model and how to use this model for classifying the instances were clearly explained in the Javadoc. This facilitated code writing and implementation to Android platform. We also created an Android application called “ARService”. ARService is generally an android service with a small GUI that the user can control the service. The infrastructure of the application is simple. In the user’s first run the application, the application creates a model from the dataset and saves the model to its own running directory. Thus the application doesn’t need model building anymore. This technique saved a considerable amount of time. After the model building, ARService uses the accelerometer sensor for making the online activity recognition and saves the recognized acts to the local SQL database called SQLite. This application is only for the testing and observing the performance of the classifier. In the real application, the readings of the recognized act will be sent to a different server. The screenshots of the ARService GUI and SQLite database can be found below.



Service isn't Running

Service is online for - seconds

```
SELECT * FROM Activity
1 - 9 / 9
TIME                 ACT
21.11.2013_20:20:40  Walking/Running
21.11.2013_20:20:50  Walking/StandingStill
21.11.2013_20:21:00  Walking/StandingStill
21.11.2013_20:21:11  Walking/StandingStill
21.11.2013_20:21:21  Walking/StandingStill
21.11.2013_20:21:31  Walking/StandingStill
21.11.2013_20:21:41  Walking/StandingStill
21.11.2013_20:21:52  StandingStill/Walking
21.11.2013_20:22:02  StandingStill/Walking
```

Figure 3.16. ARService and service readings

The recognized act was split into two. The first one is the real recognized activity and the second one is the first one’s alternative. This implies that the recognized act of the users is mostly the first one, but it can be the second one either.

3.6. POSITION-AWARE RECOGNITION: PERFORMANCE EVALUATION

It has been observed in Figure 3.15, the accuracy which was generated by only using the pants pocket position was good but on the other hand, people carry their phones at various positions like bag or jacket pocket. This situation causes a challenging task to attain accurate results by just using directly one classification model based on every phone position. In this section, we focused on phone position uncertainty problem and also we compared the classification results with position -independent and position-dependent classification models.

The dataset that was used in previous section allowed us to perform these position-dependent and position-independent tests. These tests were consisted of the usage of the classification models that was created from the data which was gathered by different persons who carried their phones at different positions (messenger bag, backpack, jacket pocket, pants pocket).The data of nine people was used for learning phase and the data of one person was used for testing phase. This method is known as “leave-one-out” approach. The goal in here was to secure a good comparison between the results of position-dependent and position-position independent tests by using user-independent way. The algorithms of KNN and decision trees were used for creating the models in the learning phase.

Table 3.1. Classification Accuracy of position-independent model

	KNN		Decision Tree	
	Precision	F-measure	Precision	F Measure
StandingStill	%47,5	%51,7	%53,9	%56,9
Walking	%99,2	%90,9	%98,9	%93,8
Running	%87,7	%93,5	%89,2	%94,3
Biking	%91,1	%94,6	%95,8	%96,4
Sitting	%65,6	%61	%66,5	%63,3
Weighted Average	%79,2	%79,1	%81,7	%81,7

Table 3.2. Classification accuracy of pants pocket position model

	KNN		Decision Tree	
	Precision	F Measure	Precision	F Measure
StandingStill	%100	%100	%100	%100
Walking	%100	%92,6	%100	%96,8
Running	%100	%100	%93,3	%96,6
Biking	%82,4	%90,4	%99,4	%99,7
Sitting	%100	%100	%100	%100
Weighted Average	%96,5	%96,4	%98,5	%98,5

Table 3.3. Classification accuracy of messenger bag position model

	KNN		Decision Tree	
	Precision	F Measure	Precision	F Measure
StandingStill	%19,7	%32,8	%61,2	%74,4
Walking	%100	%98,1	%99,5	%99,5
Running	%96,9	%98,4	%100	%100
Biking	%98,8	%99,4	%98,2	%98,8
Sitting	%99,4	%72,1	%98,1	%83,5
Weighted Average	%84,1	%81,2	%92,0	%91,7

Table 3.4. Classification accuracy of backpack position model

	KNN		Decision Tree	
	Precision	F Measure	Precision	F Measure
StandingStill	%84,1	%75,4	%87,4	%76,1
Walking	%100	%99,7	%98,4	%98,6
Running	%99,4	%99,7	%100	%99,1
Biking	%100	%100	%98,8	%99,4
Sitting	%63,4	%71,1	%60,2	%70
Weighted Average	%89,8	%89,7	%89,3	%89,2

Table 3.5. Classification accuracy of jacket pocket model

	KNN		Decision Tree	
	Precision	F Measure	Precision	F Measure
StandingStill	%1,3	%2	%0	%0
Walking	%100	%94,8	%100	%100
Running	%88,1	%93,7	%100	%100
Biking	%92,1	%95,9	%100	%100
Sitting	%70,6	%51,7	%81,3	%58,7
Weighted Average	%72,1	%69,3	%77,9	%73,6

Table 3.6. Overall results of weighted averages from position model tables

	KNN		Decision Tree	
	Precision	F Measure	Precision	F Measure
Position Independent	%79,2	%79,1	%81,7	%81,7
Pants Pocket	%96,5	%96,4	%98,5	%98,5
Messenger bag	%84,1	%81,2	%92,0	%91,7
Backpack	%89,8	%89,7	%89,3	%89,2
Jacket Pocket	%72,1	%69,3	%77,9	%73,6

It has been observed from the results, the tests that were performed with knowing the phone position, was better than the test that was performed without knowing the phone position except the jacket pocket position. In addition to that, it can be seen that the classification accuracies of "StandingStill" and "Sitting" activities were poorer at messenger bag, backpack and jacket pocket positions. When the confusion matrix of tests of these positions was analyzed, it has been observed that "StandingStill" activity was generally labeled as "Sitting". This situation decreased the classification accuracy. As a matter of fact, the classification accuracy of "Sitting" activity in jacket pocket position was observed as %0. This was an expected situation since "Sitting" and "StandingStill" is very similar activities and misclassification was normal in these circumstances. On the other hand, all of the activities were almost perfectly classified at pants pocket position because of the nature of the leg movement.

As an improvement for these misclassifications, it was possible to merge “Sitting” and “StandingStill” activity into one called “Stationary”. When this modification was implemented to the tests at Table 3.1, precision and F measure values were recorded as %97.7 and %96.2 for KNN, %98.5 and %98.7 for decision tree, respectively. As you can acknowledge that was a very considerable improvement.

4 –ARSERV: ADVANCED DATA COLLECTION AND ACTIVITY RECOGNITION FRAMEWORK

In this chapter, the work of the exploratory phase was taken one step further. In the exploratory phase, the implemented android application was only capable of recording accelerometer data. As an improvement, the new set of sensors like GPS, gyroscope etc. was added to the system. Besides these sensors, the ability of reading logs of messages, calls and phone state was also added to this new implementation. By using these logs and new set of sensors, a transition from the notion of "context" to the notion of "scene" can be secured which was very important because more information about the user (current location, call and message logs etc.) is a key to create a dynamic and flexible ambient intelligence environment.

Another development on the previous system was about the phone location. The previous system used only the data with pants pocket orientation which means the system was only capable of recognizing the activities when the phone was carried in the pants pocket. However, as you can acknowledge in real life, people carry their phones at different locations. Therefore, it was necessary to add position aware recognition ability to our new system. In brief; you will find the improvements on our previous systems based on these topics.

4.2. SYSTEM OVERVIEW

We mentioned some issues that are need to be solved in order to implement a feasible activity recognition system in the section of initial implementations. In this section, we will present our new system which is an improved version of the previous activity recognition system. The new system which is called "ARServ", was built on our previous system with some important modifications.

ARServ has some important differences comparing to old system. The old system was basically an Android service with a very simple GUI. ARServ has more sophisticated GUI.

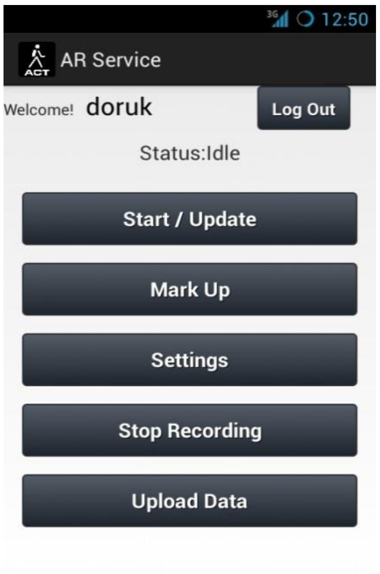


Figure 4.1. ARServ GUI

ARServ has four main android services which work in the background. These services are

- Activity and Smartphone Location Service
- Online Markup Service
- Data Upload Service

Activity and Smartphone Location Recognition Service is the main service of the system. This service is basically same as the previous one. However this service has several different abilities. First one is the ability of collecting data from other sensors and smartphone logs. The system gives the user a selection among this sensors and logs before starting the service.

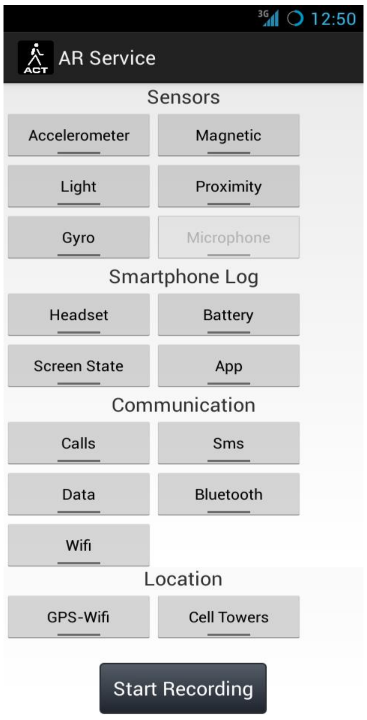
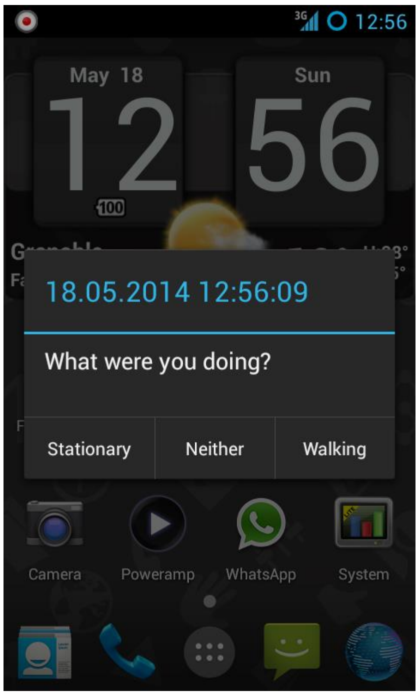


Figure 4.2. Sensor Selection Interface

There are four main data group which are sensor, smartphone log, communication and location. The user can select desired toggle button and starts the service. The service collects the data from selected sensors or logs. The microphone sensor is disabled for now.

The other ability is the ability of recognizing phone location. As you can acknowledge from the section of "Position-aware recognition", this ability can have a great impact on the recognition results since the knowledge of phone location during the activity recognition can increase the accuracy of the system considerably. The logic of position-aware recognition is implemented in two steps in this service. The main idea is to recognize the position of the phone then make the classification of the activity. There are five classification models that were created by the system for this purpose. One of these models is phone location model. This model uses 4 features that are extracted from only the accelerometer data. These features are the same feature (pitch and roll angles + their standard deviation over a window) which were mentioned at the section of "Orientation". Other four models are for each phone location (messenger bag, backpack, pants pocket, jacket pocket). The process of recognizing an activity starts with arriving of the accelerometer data to the service and two feature extraction (one for recognizing the phone location, one for recognizing the activity) methods is applied to this raw accelerometer data. These features are the same feature that we mentioned at the previous sections. There are two feature extraction methods. The service first recognizes current phone location of the user then uses the relevant classification model for recognizing the activity.

Online Markup Service is the service for marking the current activity. This system is kind of a validation system. It has been observed in the previous sections, the overall classification accuracy of our models is good. Nevertheless these results don't mean that the online classification results will be satisfying too; therefore there is a need for a validation system. This service basically creates popups when the recognition service is online. These popups ask user to annotate their activity based on their previous activities which also are recognized by the system.


18.05.2014 12:56:09

What were you doing?

Stationary | Neither | Walking

Figure 4.3. Online Markup Popup

The interval of popup appearance can be set on the settings. Each popup displays two most occurring activities which are recognized by the system in last one minute. User can choose relevant activity or simply can choose neither of them. These user annotations are recorded by the service. We can have an idea about the accuracy of the online recognition by comparing the user annotations and activity results. In addition to that user does not have to wait for the popup since the manual markup is available within the system.

Lastly, *Data Upload Service* is responsible for the data transfer between Android smartphone and server. The data that is gathered from the system is sent to a server for future examinations and offline recognition. This service works at intervals and these intervals can be set at the settings in the application. There is also a button for the manual upload for instant upload of the data. Data upload service is only able to operate when the phone is connected to a Wi-Fi connection. The server side of the system is also changeable at the settings. For now we use a simple MySQL database and an apache server for storing the data on the web.

5 – GSU ACTIVITY SET: A NEW DATA COLLECTION EXPERIMENTS

In this chapter, we collected data by using our ARSERV framework in order to analyze more complex data than the previous one. Previous data collection scenario was rather simple compared to our new data collection in terms of natural interaction between users and phones. In our data collection scenario, participants also performed activities such as making a phone call, interacting with an application on the smart phone, sending an SMS, using the phone carried in the hand. The phones carried in the backpack and the pocket were stationary whereas the participants held freely the phone in the hand, could put it on the table while sitting, could shake the arm while walking, running, or keep it static, or change the hand holding the phone.

We collected activity data from 15 participants carrying three phones in different positions, in a backpack, in the hand and in a pocket. In total, 700 minutes of activity data is collected from the participants, performing primary activities of walking, running, sitting, standing, climbing up/down stairs and transportation with a bus. In the data collection scenario, participants also performed activities such as making a phone call, interacting with an application on the smart phone, sending a SMS, using the phone carried in the hand. The phones carried in the backpack and the pocket were stationary whereas the participants held freely the phone in the hand, could put it on the table while sitting, could shake the arm while walking, running, or keep it static, or change the hand holding the phone. The data collected was then processed using the WEKA machine learning toolkit, using the Random Forest[4] classifier.

In the data collection process, 15 participants were included, eight male and seven female. The age range among participants was between 20 and 40 who are research

[4] Random Forests. URL: http://stat-www.berkeley.edu/users/breiman/RandomForests/. [accessed June 2014]

assistants, faculty and students in the Engineering Faculty at Galatasaray University. The data was collected using two different phone models: Samsung Galaxy S2 and Samsung Galaxy S3-Mini. Three phones (two Samsung Galaxy S3 Mini and one Samsung Galaxy S2) were carried by the participants in three different positions: one in the hand, one in the backpack and one in the pocket. A scenario was predefined and the participants followed the same scenario during the experiments. In the scenario, we had two types of activities: primary activities and secondary activities. Primary activities include stationary (sitting and standing), walking, stairs (up and down), jogging and transportation (with a bus), whereas secondary activities include making a phone call, sending an SMS, opening an application. Secondary activities were included to make the scenario more realistic, reflecting the daily usage of the phone. During the data collection, all the activities were tagged by an annotator who was with the participants during the data collection for keeping the ground truth data. The list and sequence of activities were as follows: stand, walk and exit the office, go upstairs, stop and stand while making a phone call, go downstairs, walk to the cafeteria, sit in the cafeteria while interacting with an app, stand, run to the campus entrance (100 meters), walk to the second entrance of the campus (100 meters), cross the street to get to the bus stop, get on the bus, get off at the next stop (250 meters), walk to the traffic lights (50 meters), stand at the traffic lights, run to the campus entrance (100 meters), stop at the entrance, walk back to the office, sit and send an SMS. The scenario locations are presented in Figure 5.1 on Google Maps, in Galatasaray University campus.

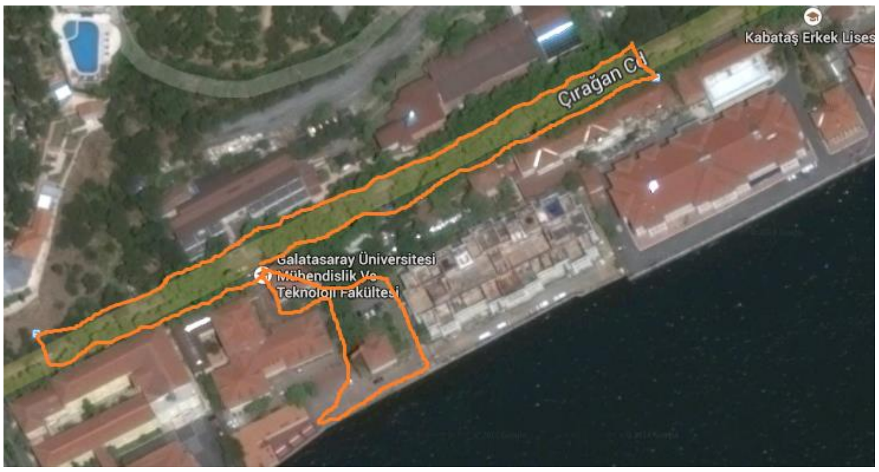


Figure 5.1. Data Collection Scenario Locations on Google Maps

In the data collection experiments, data from all the sensors available on the phones were collected. Accelerometer was the primary sensor and it was sampled at 100Hz. Data from gyroscope, magnetic field, proximity, microphone and light sensors were also captured. Besides these, we also collected location data using Google Location services, communication data such as WiFi access point SSIDs, Bluetooth scans, as well as the phone state data such as battery information, screen state (turned on and off), app usage and headset information (plugged in and out). However, in this paper we only utilize the information from accelerometer for activity and position recognition. Other data is planned to be exploited in other studies and will be made available for other researchers.

5.1. POSITION RECOGNITION

In the evaluations, we considered the worst-case scenario and applied leave-one-subject-out cross validation, hence in the training phase we included the data from 14 participants from 3 different positions as well as the data from the test person, from the positions other than the data used in the test phase. In the test phase, data from one person from a specific position is used. Hence, in the results average value calculated from 45 (15 participants*3 positions) tests is reported. This was due to the fact that, we are interested in developing a practical application that works in real time and we aim to test user generalization with these experiments.

5.1.1. Classification with Basic Acceleration Features

In the first experiment, our objective was to examine the position recognition performance using basic acceleration features that are also used for the activity recognition purpose. Hence, instead of extracting additional features for the position recognition task, we used the same set of features that are given in Section 2.2**.** In Table 5.1, the results of the experiment are provided. The average recognition accuracy was reported as 77.34%, while the F-measure was found to be 86.43%. Instead of computing average values, weighted averages are given where the number of instances for each specific case is taken into account. When we consider the position-specific results, they are similar, only differing by 2-3%. Using the same set of features that are also used in the activity identification, acceptable position recognition accuracy is achieved in this set of experiments.

Table 5.1. Phone position recognition with basic acceleration features

	Accuracy	F-Measure
Hand	%78,84	%88,03
Bag	%77,58	%86,61
Pocket	%75,59	%84,66
Weighted Average	%77,34	%86,43

5.1.2. Classification with Angular Features

In the second experiments, instead of using basic features that are also used for the activity recognition, we focused on using angular features that are extracted from pitch and roll values. The intuition was that these features can provide more information about the position of the phone based on the orientation changes.

Table 5.2. Phone position recognition with angular acceleration features

	Accuracy	F-Measure
Hand	%83,64	%90,69
Bag	%81,65	%86,82
Pocket	%68,07	%75,89
Weighted Average	%77,78	%84,46

In Table 5.2, the results of the experiment are provided. The average recognition accuracy was reported as 77.78%, while the F-measure was found to be 84.46%. Looking at the position-specific results, the highest accuracy was achieved with the hand position whereas the pocket position was identified with a fairly lower accuracy.

We believe that, this is due to the tight positioning of the phone in the pocket. The changes in the orientation of the phone were less compared with the hand and bag positions. Compared with the results achieved using only the basic features, the average results are similar, however, bag and hand positions were recognized with a higher accuracy, around 4-5%.

5.1.3. Classification with Angular and Basic Acceleration Features

As a final experiment, we considered using both the angular features and the basic features to investigate the combined effect. In Table 5.3., the results are provided. The average position recognition accuracy is reported as 85.04% in this case, while the F-measure is 90.01%. Compared to the results achieved with using basic features presented in Table 5.1 and the results of using angular features presented in Table 5.2, in each position higher recognition performance is achieved. On average, 8% increase in accuracy and 5.5% increase in F-measure is reported.

Table 5.3.. Phone position recognition with basic and angular acceleration features

	Accuracy	F-Measure
Hand	%90,86	%95,14
Bag	%85,80	%89,63
Pocket	%78,47	%85,27
Weighted Average	%85,04	%90,01

Although it is difficult to compare the results of our experiment with related studies that aim to recognize phone positions, due to the fact that not exactly the same experiment scenarios were followed, 85% recognition accuracy with three different positions and six different activities can be considered relatively high. For instance in Jun-geun et al. (2012), bag, ear, hand and pocket positions were considered and the average recognition accuracy was reported to be 94% using leave-one-participant-out method. However, only the walking activity was performed by the participants. In Lane et al. (2010), 80%

accuracy was reported for in-pocket versus out of pocket recognition. Nevertheless, the number of positions to be recognized was limited and the microphone on a smart phone platform was used and the number of subjects was not provided. In Martin et al. (2013), the best position recognition accuracy was reported as 92.94% with best set of all features and 66% with selected features. However, in that study not only the accelerometer was used but also light, proximity, magnetometer, gyroscope, gravity and linear acceleration were also integrated and as a future work position recognition with only accelerometer was targeted. As mentioned, in the experiments we also collected different modalities with different sensors, such as gyroscope, microphone, and in future work we aim to investigate the combined effect of using different sensors for position recognition and study the tradeoffs between accuracy and battery consumption levels.

5.2. ACTIVITY RECOGNITION: PERFORMANCE EVALUATION

Since the accelerometer generates different signals when the phone is carried in different positions, by knowing the position/placement of the phone a position-specific algorithm can be used to improve the recognition accuracy. Although this has been investigated in previous studies (Martin et al., 2013), (Jun-geun et al., 2012), (Yang et al., 2013), there is no clear conclusion whether position-specific physical activity recognition provides improved accuracy compared to the position-independent recognition. In this section, we focus on this issue and first study position-independent classification and next position-specific classification providing the results for each activity and phone position.

5.2.1. Position-Independent Activity Classification

In this set of experiments, activity recognition is performed without knowing the phone position. Random forest classifier is used and leave-one-subject-out cross validation is applied similar to the position recognition experiments with 45 tests in total. The average recognition accuracy considering all position and activities is 74.51% and the detailed results are presented in Table 5.4.

Table 5.4. Position-independent activity classification

	Hand		Bag		Pocket	
	Accuracy	F-measure	Accuracy	F-measure	Accuracy	F-measure
Sit	%26,55	%33,96	%49,81	%54,07	%74,38	%72,53
Stand	%77,75	%71,13	%83,37	%77,56	%83,67	%80,60
Walk	%87,86	%82,69	%90,06	%85,55	%88,39	%85,36
Stairs	%18,14	%22,62	%29,77	%36,32	%37,01	%42,59
Run	%88,21	%91,02	%93,14	%93,07	%94,93	%92,56
Bus	%31,65	%33,46	%36,46	%43,28	%33,02	%43,91
Weighted Average	%69,05	%67,01	%75,82	%74,12	%78,64	%77,41
Weighted Average Accuracy (All Positions)	%74,51					

Considering the weighted average results, recognition performance with the hand position is relatively lower. The reason is that in the hand position, it is challenging to extract body-movement related information and in the experiments, the users freely held the phone, could put it on the table while sitting, could shake the arm while walking, running, or keep it static, or change the hand holding the phone. Additionally, the participants performed the secondary activities (sending SMS, making a phone call, starting an app) using the phone in the hand. Considering the activity-specific activity recognition results, the recognition accuracy for the sitting activity was considerably low for hand and bag cases. When we examined the detailed confusion matrices, we realized that sitting activity was confused with the standing activity since the users are stationary in both cases. However, in the pocket case since the posture of the leg can be captured, this effect was lower. In the hand case, while the participants were sitting or standing, they were also performing the secondary activities and this increases the confusion rate. Walking and running activities were recognized with a high accuracy. Analyzing the confusion matrices, we also realized that the stairs activity is confused with the walking activity while the transportation (bus) activity is confused with walking and standing activities. Confusion of the transportation activity was expected since the users walked and stood still while travelling in the bus. In the future work, we aim to increase the recognition of transportation activity with using linear acceleration.

5.2.2. Position-Aware Activity Classification

In the last set of experiments, random forest classifier was trained for each position. Leave-one-subject-out method is applied similar to the other experiments. The average recognition accuracy considering all position and activities is 74.88% and the detailed results are presented in Table 5.5.

Table 5.5. Position-specific activity classification

	Hand		Bag		Pocket	
	Accuracy	F-measure	Accuracy	F-measure	Accuracy	F-measure
Sit	%39,04	%42,65	%46,18	%45,76	%75,64	%76,71
Stand	%78,62	%72,50	%79,45	%76,38	%86,03	%83,84
Walk	%87,04	%82,29	%87,65	%84,77	%89,60	%85,07
Stairs	%15,62	%19,44	%38,53	%40,88	%35,67	%40,62
Run	%91,14	%92,7	%93,47	%94,59	%89,60	%91,70
Bus	%22,93	%28,4	%40,59	%46,50	%48,22	%54,51
Weighted Average	%69,98	%67,90	%74,40	%73,24	%80,24	%79,32
Weighted Average Accuracy (All Positions)	%74,88					

Similar to the results of position-independent recognition presented in Table 5.4, sitting, stairs and transportation activities were recognized with a lower accuracy and stand, walk and run activities revealed a higher accuracy. Considering activities individually, in the case of sitting activity improved accuracies are achieved for hand and pocket positions, but in the pocket position the increase is not significant. For the standing and walking activities, results were not significantly different. In the case of running activity, results were not significantly different for hand and bag positions, but in the pocket case the accuracy dropped. For the transportation (bus) activity there is a significant increase in bag and pocket positions but there is a significant decrease in the hand position. Finally for the stairs activity, there is a significant increase in the bag position. Hence, the effect of the position information on classifier performance is dependent on the activities and the locations involved. These findings are consistent with previous results found in the literature (Thiemjarus et al., 2013), (Jun-geun et al., 2012).

5.2.3. Discussion

Overall, with a perfect position classification assumption, the position-specific classification performs nearly the same with the position-independent recognition accuracy, revealing 74% average accuracy, whereas it performs slightly better by 2% in average for the pocket case. In previous studies, particularly in Jun-geun et al. (2012), Thiemjarus et al. (2013), similar results were reported, and our study also confirms that position-specific activity recognition does not perform significantly superior compared to the position independent recognition. However, it contradicts with the findings in Martin et al. (2013) where it was reported that there is strong evidence that the classifiers that take into consideration the position outperform the ones that classify regardless of it according to the applied Mann-Whitney U-test to the data. As a future work, the different datasets should be explored with the same set of features and classifiers to provide concrete conclusions on the use of position-specific recognition compared to position independent recognition.

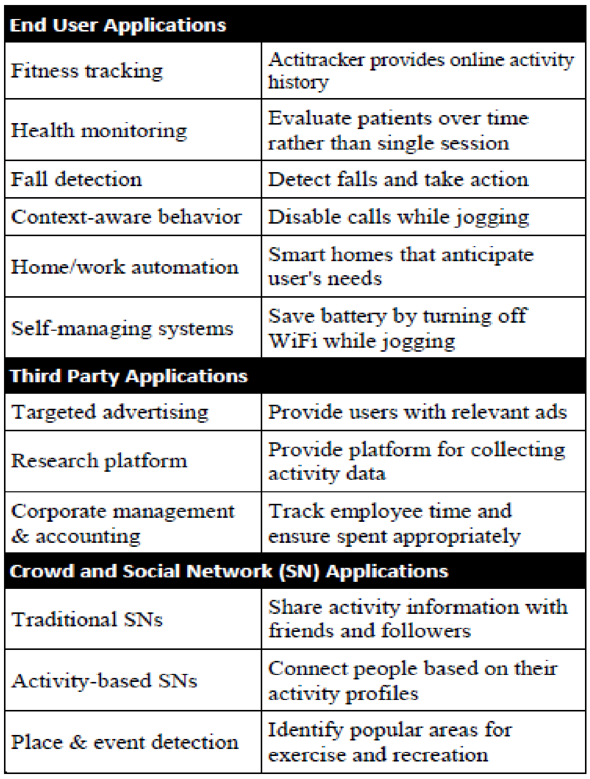
6 – CONCLUSION

In this study, we evaluated the impact of the classifiers and some other characteristics like feature types on the performance of online activity recognition systems. Accordingly, the performance of some classical classifiers like naïve Bayes, decision trees etc. and custom classifiers like Clustered-KNN, DT-DHMM in the context of activity recognition was evaluated. It has been seen that the custom classifiers generally exhibited a much better performance than the classical classifiers. We also saw that this result is associated with the online data streaming property of the online activity recognition systems. But we must indicate that C4.5 decision tree classifiers performed just well as other custom classifiers. This is the reason that nearly every work in literature has an experiment based on decision trees and as you can observed that we also used C4.5 decision tree in our AR system as the classifier.

Another theme in our work is centered on phone position. It is known that phone position/placement information can provide valuable context data for mobile sensing applications. As an example different sensor readings may be triggered depending on the position of the phone and also improved resource management can be achieved using algorithms that use phone position as input. In this study one challenge that we pursued is to successfully identify phone positions using accelerometer only data without additional sensors. Moreover, in order to evaluate how much performance gain the position information can provide, we focused on the accelerometer-based physical activity recognition scenario. Specifically, we evaluated the gain that can be achieved in the activity recognition accuracy by using position-specific classifiers compared to position-independent classifiers. For this purpose, we collected physical activity data from 15 participants from three different phone positions, and the participants followed a daily life scenario instead of performing an artificial and disconnected series of activities. To make the data collection more realistic, participants performed secondary activities, such as making a phone call, sending an SMS and opening an application on the phone. The collected data is processed with the Random Forest classifier. According to the results of position recognition, using basic accelerometer features which are also used in the activity recognition, can achieve an accuracy of 77.34%, and this ratio increases to 85% when basic features are combined with angular features calculate from the orientation of the phone.

After the position recognition experiments, activity recognition experiments are performed to explore the impact of position information. In our experiments, we showed that the effect of the location information on classifier performance is dependent on the activities and the locations involved, which is consistent with previous results found in the literature. On average, the recognition accuracies are similar for position specific and position independent recognition. Only for the pocket case, slight increase is achieved.

Even if online activity recognition is a primitive research field, when this research field begins to mature, high quality and innovative commercial applications would be developed instantly (Lockhart et al., 2012). Because of this research field is in its infancy, these applications are rare at the moment, although the researchers believe that these application will arrive shortly due to the ubiquity and robustness of the mobile devices (Lockhart et al., 2012). There are many application fields that use the activity recognition systems. The researchers believe that the future work of this field will take form in these application fields that you can see in figure 6.1.



End User Applications	
Fitness tracking	Actitracker provides online activity history
Health monitoring	Evaluate patients over time rather than single session
Fall detection	Detect falls and take action
Context-aware behavior	Disable calls while jogging
Home/work automation	Smart homes that anticipate user's needs
Self-managing systems	Save battery by turning off WiFi while jogging
Third Party Applications	
Targeted advertising	Provide users with relevant ads
Research platform	Provide platform for collecting activity data
Corporate management & accounting	Track employee time and ensure spent appropriately
Crowd and Social Network (SN) Applications	
Traditional SNs	Share activity information with friends and followers
Activity-based SNs	Connect people based on their activity profiles
Place & event detection	Identify popular areas for exercise and recreation

Figure 6.1. Summary of activity recognition applications (Lockhart et al., 2012).

As you can acknowledge from this study there are many issues on this field. Some of these issues are caused because of the lack of infrastructure; some of them are caused because of the algorithmic problems but it can be said that all of these issues are an obstacle to build a stable system and because of these issues the activity recognition and mobile sensing were still in its infancy. It our belief that once these issues are overcome, this field will advance quickly, acting as a disruptive technology across many domains including social networking, health and energy.

For future works, we plan to improve our AR system. At the moment, we have an Android application which uses accelerometer sensor for recognizing the user’s acts. Its accuracy is satisfying. As an improvement, new sets of sensor like GPS and microphone was implemented to the system and also Android platform gives us the opportunity to use the phone logs. The phone logs include the state of the phone like network connection status, the phone idle or not, call logs, message logs etc. For now, we are only able to collect them but we believe that by using these logs and new sets of sensors, we can make a transition from the notion of “context” to the notion of “scene” which is able to give us more information about the user’s current state. As it has been said in the first place, the concept of “context” is very critical because determining a context of a user, is a key to create a dynamic and flexible ambient intelligence environment and it’s know that the more information about the user’s current state, in other words the user’s current scene means more dynamic and flexible ambient intelligence environment.

As another future work we also plan to investigate our findings at chapter 5 on different datasets and with other sensor modalities available in our dataset. When other sensor modalities are taken into account, it will be interesting to investigate the resource, particularly battery, consumption and analyze the tradeoffs between recognition accuracy and resource usage.

REFERENCES

Alanezi K. and Mishra S.: Impact of Smartphone Position on Sensor Values and Context Discovery, Computer Science Technical Reports. paper 1030, 2013.

Avci, Akin; Bosch, Stephan; Marin-Perianu, Mihai; Marin-Perianu, Raluca; Havinga, Paul, Activity Recognition Using Inertial Sensing for Healthcare, Wellbeing and Sports Applications: A Survey, Architecture of Computing Systems (ARCS), 2010 23rd International Conference , pp.1-10, 2010.

Bao, L.; Intille, S.S.; Activity Recognition from User-Annotated Acceleration Data, Pervasive Computing Lecture Notes in Computer Science, pp.1-17, 2004.

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

Chon Y., Talipov E., and Cha H.: Autonomous management of everyday places for a personalized location provider. Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, 42(4), pp. 518-531, 2012.

Coskun, D. (2014). Online activity recognition for smart-phones from accelerometer and audio data: a sequence model learning approach. M.Sc. Thesis. Joseph Fourier University: France.

Domingos,P.. A few useful things to know about machine learning. Commun. ACM, 55(10), pp. 78–87, 2012

Incel, O.D., Kose, M., Ersoy, C. : A Review and Taxonomy of Activity Recognition on Mobile Phones, Vol. 3, No. 2, pp. 145-171, 2013.

Jun-geun, P., Ami, P., Dorothy, C., Teller, S. and Ledlie, J.: Online pose classification and walking speed estimation using handheld devices. *In* Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp '12). ACM, New York, NY, USA, pp. 113–122, 2012.

Kononen, V., Mantyjarvi J., Simila H., Parkka J. and Ermes M., "Automatic feature selection for context recognition in mobile devices", Pervasive and Mobile Computing, Vol. 6, No. 2, pp. 181-197, 2010.

Kose, M. ; Incel, O.D. and Ersoy, C. : Performance Evaluation of Classification Methods for Online Activity Recognition on Smart Phones, Signal Processing and Communications Applications Conference (SIU), pp.1- 4, 2012.

Kose, M., O. D. Incel and Ersoy, C: 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.

Lane, N.D. ; Miluzzo, E. ; Hong Lu ; Peebles, D. ; Choudhury, T. ; Campbell, A.T. : "A Survey of Mobile Phone Sensing", vol.48, no.9, pp.140,150, 2010.

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

Martin H., Bernardos A. M., Iglesias J., and Casar J. R., "Activity logging using lightweight classification techniques in mobile devices," Personal and Ubiquitous Computing, vol. 17, no. 4, pp. 675–695, 2013.

Quinlan J.R.. Induction of decision trees. Machine Learning, 1(1):81–106, 1986.

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

Siirtola, P.; Röning, J.; “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, 2012.

Thiemjarus S., Henpraserttae A., and Marukatat S., “A study on instance-based learning with reduced training prototypes for device context-independent activity recognition on a mobile phone,” in 2013 IEEE International Conference on Body Sensor Networks (BSN), pp. 1–6, 2013.

Ustev, Y.E., Ersoy, C. and Ö. Durmaz Incel, User, Device and Orientation Independent Human Activity Recognition on Mobile Phones: Challenges and a Proposal. In Adj. Proc. UbiComp'13, Ubiquitous Mobile Instrumentation, 2013.

Yan, Z.; Subbaraju, V.; Chakraborty, D.; Misra, A. ; Aberer, K. : Energy-Efficient Continuous Activity Recognition on Mobile Phones An Activity-Adaptive Approach, 16th International Symposium , pp.17- 24, 2012.

Yang J., Munguia-Tapia E., and Gibbs S., “Efficient in-pocket detection with mobile phones,” in Proceedings of the 2013 ACM conference on Pervasive and ubiquitous computing adjunct publication., pp. 31–34, 2013.

BIOGRAPHICAL SKETCH

Doruk Coşkun was born in 1989 in Rize, Turkey. He finished Burak Bora Anatolian High School in 2008 and received his Bachelor's degree of Computer Engineering in 2012 from Galatasaray University in Istanbul, Turkey. He is currently pursuing a Master's degree in Computer Engineering at the same university.

His first publication is a conference paper written under the supervision of Assist. Prof. Dr. Özlem Durmaz Incel. Its title is "Position-aware activity recognition on mobile phones" (Coskun et al., 2014), and it was presented at the Signal Processing and Communications Applications Conference (SIU 2014). He wrote another paper called "On-line Context Aware Physical Activity Recognition from the Accelerometer and Audio Sensors of Smartphones" with his other supervisors Assoc. Prof. Francois Portet and David Blachon from Laboratoire d'informatique de Grenoble in France. It was presented at European Conference on Ambient Intelligence (AMI 2014) international conference.