Approximate Ensemble Methods for Physical Activity Recognition Applications

A dissertation submitted by **Pierluigi Casale** at Universitat de Barcelona to fulfil the degree of **Ph.D. in Applied Mathematics**.

Barcelona, November, 2011

Director: **Oriol Pujol and Petia Radeva**
Dept. of Applied Mathematics and Analysis, University of Barcelona
& Computer Vision Center

This document was typeset by the author using $\LaTeX\ 2_\varepsilon$.

The research described in this book was carried out at the Computer Vision Center and University of Barcelona.

ISBN 978–84–938351–6–3
Deposito Legal: B-37.482-2011
Printed by Ediciones Gráficas Rey, S.L.

A Mireia

0.1 Acknowledgements

I would like to thank the Computer Vision Center and projects TIN2006-15308-C02, FIS-PI061290, CONSOLIDER- INGENIO 2010 (CSD2007-00018), MI 1509/2005, TIN2009-14404-C02, La Maratò de TV3 082131 for supporting this research.

I would thank Petia and Oriol, my supervisors. Without them this work would not have been possible. Many thanks Petia. Many thanks Oriol. I am very grateful.

También quiero agradecer a todas las personas que he conocido en estos años, a todos los compañeros y compañeras de la Universidad de Barcelona y del Centro de Visión por Computador, a los estudiantes y a todos aquellos con que he tenido la oportunidad de intercambiar ideas y opiniones. Muchas gracias a todos. Un agradecimiento especial, y no se en que idioma darselo, se lo debo a Francesco, con el cual he compartido mucho a lo largo de este viaje. Muchas gracias Fra.

No tinc tantes paraules per expressar els meus agraïments a la Mireia. Aquest treball és seu també. I amb la Mireia, un agraïment especial a la Carmeta, al Vicenç i a l' Antonio. Moltes, moltes gracies per tot.

Ultimi, non per importanza, i miei cari a Sorbello. Nonna, mamma, papá, zio Fausto, Milena, Saverio, Simon e Angelo. Grazie di tutto ancora una volta. Questa volta, almeno, avete avuto la fortuna di potervi risparmiare lo stress del rush finale!

0.2 Abstract

The main interest of this thesis focuses on computational methodologies able to reduce the degree of complexity of learning algorithms and its application to physical activity recognition.

Random Projections will be used to reduce the computational complexity in Multiple Classifier Systems. A new boosting algorithm and a new one-class classification methodology have been developed. In both cases, random projections are used for reducing the dimensionality of the problem and for generating diversity, exploiting in this way the benefits that ensembles of classifiers provide in terms of performances and stability. Moreover, the new one-class classification methodology, based on an ensemble strategy able to approximate a multidimensional convex-hull, has been proved to over-perform state-of-the-art one-class classification methodologies.

The practical focus of the thesis is towards Physical Activity Recognition. A new hardware platform for wearable computing application has been developed and used for collecting data of activities of daily living allowing to study the optimal features set able to successful classify activities.

Based on the classification methodologies developed and the study conducted on physical activity classification, a machine learning architecture capable to provide a continuous authentication mechanism for mobile-devices users has been worked out, as last part of the thesis. The system, based on a personalized classifier, states on the analysis of the characteristic gait patterns typical of each individual ensuring an unobtrusive and continuous authentication mechanism.

τρῐ́ποδας γάρ ἐείκοσι πάντας ἔτευχεν
ἑστάμεναι περί τοῖχον ἐυσταθέος μεγάροιο
χρύσεα δέ σφ’ ὑπό κύκλα ἑκάστω πυθμένι θῆκεν
ὄφρα οι αὐτόματοι θεῖο δυσαίατ ἀγῶνα
ἠδ’ αὖτις πρός δῶμα εοίατο, θαῦμα ιδέσται [1]

Homer, *Iliad*, XVIII, 373-377

[1] “For he was fashioning tripods, twenty in all, to stand around the wall of his well-builded hall, and golden wheels had he set beneath the base of each that of themselves they might enter the gathering of the gods at his wish and again return to his house, a wonder to behold.” [Murray(1924)]

Contents

0.1 Acknowledgements . . . .
0.2 Abstract i

1 Introduction 1
1.1 The third era of modern computing 2
1.1.1 Ambient Intelligence and Ambient Assisted Living 3
1.2 The Focus of this Thesis 3
1.3 Structure of the Thesis 6

2 Approximate Ensemble Methods 7
2.1 Embedding Random Projections in Boosting Ensemble 8
2.1.1 Related works on Random Projections in Machine Learning . . 8
2.1.2 Gradient Boosting Machines 9
2.1.3 The Random Projections Method 10
2.1.4 Embedding Random Projections in Boosting Machines 11
2.1.5 Experimental Results 13
2.1.6 Discussion on Random Projections 22
2.2 Approximate Polytope Decision Ensemble for One-Class Classification 23
2.2.1 Convex Approximate Polytope Ensemble 25
2.2.2 Non-Convex Approximate Polytope Ensemble 29
2.2.3 Experimental Results 31
2.2.4 Discussions 34
2.2.5 Conclusions 37

3 Wearable Sensors and Features Selection for Physical Activity Recognition 39
3.1 BeaStreamer,
a new platform for wearable computing applications 39
3.1.1 Background on Wearable Computing 40
3.1.2 Overview on BeaStreamer 41
3.1.3 Description of the system 43
3.1.4 The Operating System Level:
OpenEmbedded + Ångstrom 46
3.1.5 The Application Software Level:
GStreamer and OpenCV 47

3.1.6 Showing the potentiality of BeaStreamer 47
3.1.7 Discussions 50
3.2 Features Extraction and Selection for Physical Activity Recognition . 51
3.2.1 Significant Related Works on Physical Activity Recognition . . 51
3.2.2 Features Extraction 53
3.2.3 Features Selection 57
3.2.4 Classification Results 60
3.2.5 Conclusions 62

4 Personalization and User Verification in Wearable Systems using Biometric Walking Patterns 65
4.1 Related Works on Biometric-based Verification Systems 66
4.2 User verification system 67
4.3 Personalized Activity Subsystem 68
4.4 User Verification subsystem 69
4.4.1 Layer 1: Approximate Polytope Ensemble 71
4.4.2 Layer 2: Robust convex hull 71
4.4.3 Layer 3: Mixture of convex hulls 72
4.4.4 Layer 4: Temporally coherent convex hulls 73
4.5 Experimental Results 73
4.5.1 Comparative Results 73
4.5.2 Validation protocol 73
4.5.3 Results of Personalized Activity Recognition Subsystem 75
4.5.4 Results of User Verification subsystem 78
4.6 Discussion 82
4.6.1 Concerning the discriminability of the walking activity 82
4.6.2 With respect to personalization in the general activity recognition subsystem 84
4.6.3 Concerning users walking variability and the effect of the layer-3 convex hull algorithm 85
4.6.4 Temporal coherence and the effect of the layer-4 convex hull algorithm 86
4.6.5 Concerning the state-of-the-art 88
4.7 Conclusions 89

5 Conclusions 91

Appendix 1 95

Appendix 2: Numerical Results obtained on Approximate Polytope Ensemble 99

List of Pubblications 107
.1 Book Chapters 107
.2 Journal Papers 107
.3 Lecture Notes in Computer Science 108
.4 International Conference and Workshops 108

.5 Local Conference and Workshops 109

List of Technician/BsC/MsC Thesis Supervised **111**

.6 Technician Thesis in Informatic Engineering 111

.7 BsC Thesis in Informatic Engineering 111

.8 MsC Thesis in Electronics Engineering 112

Bibliography **113**

List of Tables

2.1 Test Patterns 14
2.2 Numerical Values of Accuracy obtained on Test Pattern with RpBoost working on $N_{0,1}$ projections 16
2.3 List of UCI problems used to evaluate RpBoost 17
2.4 UCI: Comparative with projections $N_{0,1}$ 19
2.5 Resume of Results Obtained on Test Patterns 20
2.6 Resume of Results Obtained on UCI Datasets 20
2.7 List of One Class Problems 32
2.8 Comparative Results obtained on Artificial Datasets 33
2.9 Counts of Wins, Ties and Losses obtained on D^* 34
2.10 Counts of Wins, Ties and Losses obtained on D 34
2.11 Mean Number of Projections 35
2.12 Ranking of Best Methods 37

3.1 Features Extracted on Images 53
3.2 Features Extracted on Audio 54
3.3 Features Extracted on Accelerometer Data 56
3.4 Features Selected on Images (right) and Audio (left) 58
3.5 Features Selected on Accelerometer Data 58
3.6 Normalized Confusion Matrix for D_{avm} (up) and D_m (down) 61
3.7 Comparison of Results for Physical Activity Recognition using a Single Accelerometer 63

4.1 Classification Performances for General Activity Classifier. 78

1 Confusion Matrix 95
2 Classification Performances Measures 96

3 Comparative Results: AUC obtained on Artificial Datasets with 500 datapoints. 99
4 Comparative Results: AUC obtained on Artificial Datasets with 750 datapoints. 99
5 Comparative Results: AUC obtained on Artificial Datasets with 1000 datapoints. 100

6 Comparative Results: AUC obtained on D^* on Gauss, MoG, PDE, kNN, kM and kC. 100
7 Comparative Results: AUC obtained on D^* on MST, SVDD, APE-1, APE-2 and N-APE 101
8 Comparative Results: AUC obtained on D on Gauss, MoG, PDE, kNN, kM and kC. Problems from 1 to 41. 102
9 Comparative Results: AUC obtained on D on Gauss, MoG, PDE, kNN, kM and kC. Problems from 42 to 82. 103
10 Comparative Results: AUC obtained on D^* on MST, APE-1, APE-2 and N-APE. Problems from 1 to 41. 104
11 Comparative Results: AUC obtained on D^* on MST, APE-1, APE-2 and N-APE. Problems from 42 to 82. 105
12 Comparative Results: User Verification from Walkin Patterns on Gauss, MoG, PDE, kNN, kM and kC. 105
13 Comparative Results: User Verification from Walkin Patterns on MST, SVDD, APE-1, APE-2 and N-APE. 106
14 Comparative Results: Training Time computed in seconds 106
15 Comparative Results: Testing Time computed in seconds 106

List of Figures

1.1 A shift in people-computing power ratio 2
1.2 AALIANCE Systems Composition Reference Architecture 3
1.3 Main Topics of this Thesis 4
1.4 Structure of this Thesis 5

2.1 Test Patterns: (a) *Annulus*; (b) *Checkerboard* 14
2.2 Test Patterns: Comparative with projections P 15
2.3 Test Patterns: Comparative with projections $U_{-1,1}$ 15
2.4 Test Patterns: Comparative with projections $N_{0,1}$ 16
2.5 Test Patterns : Comparative with projections P 18
2.6 UCI: Comparative with projections P 18
2.7 UCI: Comparative with projections $U_{-1,1}$ 19
2.8 Annulus Test Pattern: Comparative 21
2.9 Checkerboard Test Pattern: Comparative 21
2.10 Expanded convex polytope in 2D space 25
2.11 Illustration of the approximate convex polytope decision strategy . . . 26
2.12 Projection of an expanded convex polytope 28
2.13 Approximation of bi-dimensional Banana-shaped dataset using NAPE 30
2.14 Comparison of Training and Testing Time 36

3.1 Wearable Sensors 40
3.2 Commercial Wearable Sensors 41
3.3 BeaStreamer 42
3.4 User Wearing BeaStreamer 42
3.5 BeagleBoard 43
3.6 BeagleBoard Usage Scenario 44
3.7 Wearable Accelerometers 45
3.8 Comparison ARM/DSP 46
3.9 Example of Pictures taken wearing BeaStreamer 48
3.10 OpenCV Face Detector on BeaStreamer 49
3.11 Face Verification using BeaStreamer 49
3.12 The Gumstix Board 50
3.13 User wearing BeaStreamer 54
3.14 Visualization of activity acceleration data 55

3.15 Classification Accuracy Using Different Classifiers 60
3.16 F-Measure obtained on D_m 62

4.1 Block Diagram of the User Verification System 68
4.2 Layer structure of the User Verification subsystem 70
4.3 User verification using a convex hull as one class classifier 71
4.4 User Verification AUC 74
4.5 Accuracy of Walking Classification 78
4.6 Verification Results on Wearable System 80
4.7 Verification Results on Commercial Device 81
4.8 Accelerometer Data of five different subjects 82
4.9 Accelerometer Data of the same subject 83
4.10 Acceleration Data and its Derivative 84
4.11 Localization of walking activity 84
4.12 Walking runs in two different paths 85
4.13 Average FAR and FRR 86
4.14 FAR and FRR for layer-3 87
4.15 FAR and FAR for temporal ensemble 90

5.1 Achievement of this Thesis 91

2 Classic Wavelets: Haar (left) and Daubechies-4 (right) 98

Chapter 1

Introduction

"I'm losing my mind.
[...]Each day that passes I forget more and remember less. I don't have Alzheimer's or even brain damage. I'm just aging."
With those words, Gordon Bell starts his book "Total Recall. How the E-Memory revolution will change everything" [Bell and Gemmel(2009)]. Bell, a luminary in the computer world, has been the first person in the world attempting to digitally record his life. He wore an automatic camera, wore an arm-strap that logged his biometrics and began recording telephone calls. The wonder of the technology created is the new capability to find the desired information in the large amount of data he stored. Bell is the center of a technological revolution that will accomplish a transformation in the way humans think about the meaning of their life.

Nevertheless, Bell's words also focus on an important social concern: aging. European population is having fewer children and is getting older [Hewitt(2002)]. Furthermore, the prevalence of health problems and chronic illnesses will affect how people live, their mobility and social relationships. In this context, the use of Information and Communication Technologies (ICT) must play the role of enhancing the quality of life of these people [European Commision(2001)]. In particular, ICT solutions based on new computational paradigms as Ubiquitous Computing, aiming to hide the technological complexity, will make older and disable people the most frequent users of selfcare and eHealth services via mobile phones and computers at home. With technology pervading everyday life, people can live longer and healthier lives, changing accordingly their habits and lifestyle.

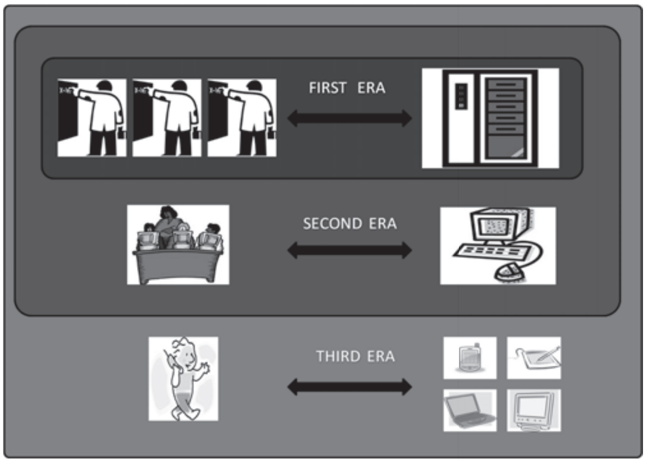


Figure 1.1: A shift in people-computing power ratio

1.1 The third era of modern computing

Ubiquitous computing is the term given to the third era of modern computing. The first era was dominated by mainframe computers, a single large time-shared computer owned by an organization and used by many people at the same time (see Figure 1.1). The second era was the era of the personal computer, a machine primarily owned and used by only one person. The third era, Ubiquitous Computing, representative of the present time, is characterized by the explosion of networked portable digital computer products in the form of smart phones, tablets, Personal Digital Assistants (PDAs) and embedded computers built into many of the devices we own. Each era has resulted in progressively larger numbers of computers becoming integrated into everyday life.

The original term Ubiquitous Computing, or UbiComp, was coined by [Weiser(1991)] at Xerox PARC. He envisioned a future in which computing technologies will be embedded in everyday artifacts, were used to support daily activities, and were equally applicable to our work and our homes. Weiser saw UbiComp as an opportunity to improve the style of computing that has been imposed on users since the early days of the mainframe, that is, sitting on a chair, looking at a screen, typing on a keyboard, and making selections with a mouse. Through this style of interaction, traditional computers consume much of our attention and divorce us from what is happening all around us, resulting in a somewhat solitary all-consuming experience. Personal computers try to virtualize our world with familiar PC desktops and nice icons representing documents, printers, and trash. UbiComp takes the opposite philosophy. UbiComp pushes the computerized versions of these technologies back into the physical world. Weiser believed that in a UbiComp world, computation could be integrated with common objects that you might already be using for everyday work practices, rather than forcing computation to be a separate activity. If the integration would be well done, you may not even notice that any computer was involved in your sorrounding. The confluence of Ubiquitous Computing and Artificial Intelligence brings to a new paradigm known as Ambient Intelligence [Augusto(2007)].

1.1.1 Ambient Intelligence and Ambient Assisted Living

The concept of Ambient Intelligence (AmI) provides a vision of the Information Society where the emphasis is on greater user-friendliness, more efficient services support, user-empowerment, and support for human interactions. People are surrounded by intelligent intuitive interfaces that are embedded in all kinds of objects and an environment that is capable of recognizing and responding to the presence of different individuals in a seamless, unobtrusive and often invisible way. The emerging demographic change towards an aging population has begun to introduce technological solutions developed to motivate and assist older people to stay active for longer in the labour market, to prevent social isolation and to help people stay independent for as long as possible. AmI plays a major role in helping to achieve these goals. AmI can help elderly individuals to improve their quality of life, stay healthier, live independently for longer, and counteract reduced capabilities which are more prevalent with age. AmI can enable them to remain active at work, in their community and at home. However, the problems cannot be solved by means of AmI alone. An interdisciplinary work is required to identify the actual problems and to develop and evaluate solutions together with the involved fields. It is important to note that the underlying technology is supposed to work in the background, as many users simply will not be able to deal with the attached technical complexity. This is a major factor to ensure the acceptance of developed solutions. The research field dealing with the above presented problems is called **Ambient Assisted Living** (AAL).

1.2 The Focus of this Thesis

The European Ambient Assisted Living Innovation Alliance [AAliance(2010)] released a road-map, which presents a detailed overview into the prospected future of AAL topics, concepts and technologies until the year 2025. This road-map contains the AAL Systems Composition Reference Architecture, shown in Figure 1.2, which details the common requirements for AAL systems. The reference architecture describes a three-layer networking approach to enable communication and connectivity between devices and services. It defines an AAL system consisting of seven component types.

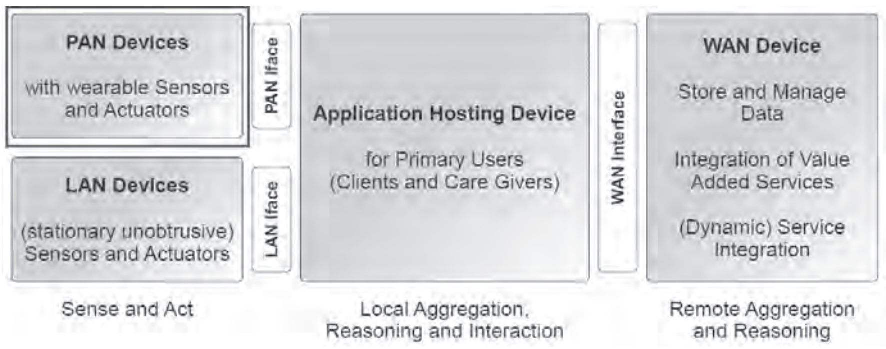

PAN Devices
with wearable Sensors and Actuators
PAN Iface
LAN Devices
(stationary unobtrusive) Sensors and Actuators
LAN Iface
Application Hosting Device
for Primary Users (Clients and Care Givers)
WAN Interface
WAN Device
Store and Manage Data
Integration of Value Added Services
(Dynamic) Service Integration
Sense and Act
Local Aggregation, Reasoning and Interaction
Remote Aggregation and Reasoning

Figure 1.2: AALIANCE Systems Composition Reference Architecture

The first block of the architecture, the Personal Area Network (PAN) Device, is object of discussion in this thesis[1]. PAN Devices represent sensors and actuators that are either in-body, on-body or wearable. PAN devices principally perceives the status of the user using a variety of sensors like obtrusive physiological ones supporting health monitoring or unobtrusive sensors like cameras, microphones and accelerometers. Some devices have the only capability to store a small batch of raw data and sending it to a network gateway. Other devices are powerful enough to process the collected data locally and provide useful support and information to the user. Depending on its type, a device may be able to provide basic feedback to the user in case of emergencies or failure. These aspects highlight an important characteristic. PAN Devices learn from users, reason about the actions occuring in a specific moment and react in an intelligent way. Nevertheless, the majority of AI and machine learning techniques running on PAN devices are methodologies usually developed for typical desktop computers. Some of these techniques still remains useful in the case of PAN devices, others are pretty difficult to be adapted to the specific limited resources hardware.

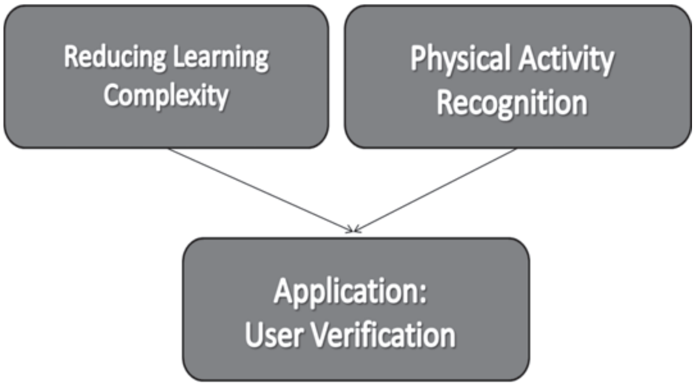

Reducing Learning Complexity
Physical Activity Recognition
Application: User Verification

Figure 1.3: Main Topics of this Thesis

The main topics of this thesis are presented in Figure 1.3. First, this work focuses on finding computational methodologies able to reduce the degree of complexity of learning algorithms in order to be adapted to computational devices with limited resources, as PAN Devices are. In particular, the focus has been put on ensembles learning, a widely accepted methodology to improve the performances of a predictive model [Kuncheva(2004)]. The technique of Random Projections is able to reduce the complexity of many geometrical and probabilistic algorithms [Vempala(2004)]. This technique is used here to reduce the computational complexity of the particular problem at hand. Beside the mere task of dimensionality reduction, Random Projections will be used to provide diversity as well, exploiting in this way the benefits that ensembles of classifiers provide in terms of performances and stability. The high dimensionality of PAN Devices data represents a of problems when machine learning algorithms are used. For this reason, a study on the optimal set of features capable to

[1]For a complete description of the other blocks please refere to [AAliance(2010)]

discriminate a subset of Activity of Daily Life (ADLs) will be performed. Monitoring ADLs should be accomplished using PAN Devices that look like common consumer devices rather devices with many wires [Lester et al.(2006)]. A wearable device is assembled in order to collect data, perform experiments on ADLs in different environmental situations and conduct the aforementioned study. Non intrusive sensors like a camera, a microphone and an accelerometer will be used instead of physiological sensors like ECG or pulse-oximetry those, despite its capabilities to provide direct information about the health of a subject, would provide more resistance at the moment to be worn by the subject.

The confluence of these two lines leads to the development of a machine learning architecture capable to verify people by means of their own walking motion patterns. Walking represents the baseline activity for healthy people. Furthermore, the capability of walking and its deviation from regular patterns is symptomatic of the health status of a person. The system is able to verify users in unobtrusive and reliable way, adapting to slight changes occurring when walking through different environments.

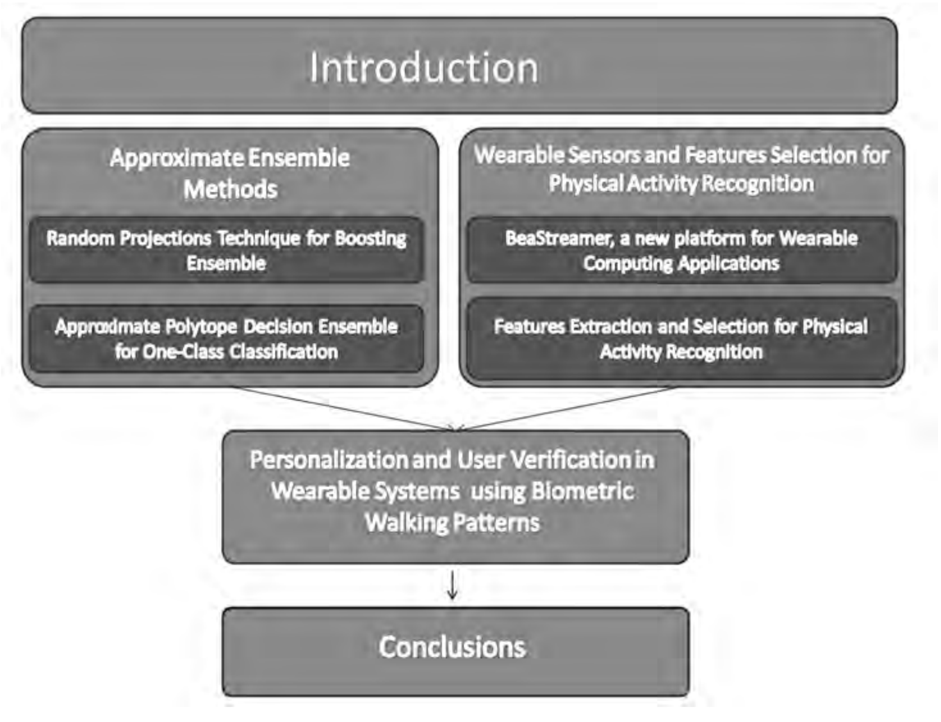

Introduction
Approximate Ensemble Methods
Random Projections Technique for Boosting Ensemble
Approximate Polytope Decision Ensemble for One-Class Classification
Wearable Sensors and Features Selection for Physical Activity Recognition
BeaStreamer, a new platform for Wearable Computing Applications
Features Extraction and Selection for Physical Activity Recognition
Personalization and User Verification in Wearable Systems using Biometric Walking Patterns
Conclusions

Figure 1.4: Structure of this Thesis

1.3 Structure of the Thesis

This thesis is structured as shown in Figure 5.1.

In **Chapter 2**, two algorithms using dimensionality reduction and approximation techniques are presented. In **Section 2.1**, the technique of Random Projections will be studied from the machine learning point of view, paying particular attention on both the dimensionality reduction aspect and the diversity that Random Projections are able to provide. These capabilities will be used in the construction of a boosted ensemble of classifiers. Different typologies of Random Projections will be taken into account and the effect of keeping or changing the original dimensionality of the learning problem will be also studied. The embedding of Random Projections in boosting ensembles will lead to a novel boosting technique called RpBoost. In **Section 2.2**, a new one-class classification ensemble strategy will be presented, the Approximate Polytope Ensemble. The main contribution of this section is two-fold. First, the geometrical concept of convex hull will be used to define the boundary of the target class defining the problem. Expansion or contraction of the structure will be introduced in order to avoid over-fitting and to provide the optimal behavior of the classifier. Secondly, the high computational complexity needed for computing the convex hull in high dimensional spaces will be reduced using the Random Projections technique.

Chapter 3 relates two aspects of physical activity recognition. In **Section 3.1**, a wearable computing device developed for acquiring data of physical activities will be presented. The device, designed for monitoring a variety of day-to-day activities, will be used for collecting different datasets in different environments. The capability of the system to acquire different datastreams will be shown and some highlights on the computational power of the system will be presented. In **Section 3.2**, a study on features for classifying physical activities will be conducted. The set of features obtained allows to achieve high results in classification problems and, when solely the features derived by motion data are taken into account, allows to use the only accelerometric sensor for classifying activity with good performances.

In **Chapter 4**, a novel technique for users authentication and verification using gait as a biometric unobtrusive pattern will be presented. The method, based on a two stages machine learning pipeline, uses an activity classifier and a verification system. The general activity classifier will be subsequently personalized for a specific user with data related to her walking pattern. As a result of this step, the system will be much more selective with respect to the new walking pattern. In the second stage, the system verifies whether the user is an authorized one or not. The core of this stage is the Approximate Polytope Ensemble and a four layers architecture is built around this one-class classifier. The architecture proposed will improve robustness to outliers, taking into account temporal coherence as well.

Finally, in **Chapter 5**, conclusions on this thesis will be reported.

Chapter 2

Approximate Ensemble Methods

Random Projections (RPs) are a well known and widely employed technique for dimensionality reduction. RPs are based on the theoretical results that high dimensional data can be projected into a lower dimensional space without significantly losing the structure of the original data. The capability to reduce effortlessly the dimensionality of the problem at hand without a remarkable loss in significance allows to create very simple and powerful learning algorithms. The principal interest towards RPs is due to this specific behavior. Using RPs, the computational complexity of reducing the dimensionality of a problem is based on a multiplication operation. This operation can be efficiently performed using Digital Signal Processors commonly available in many digital hardware systems.

On the other hand, the theoretical interest towards RPs is not limited to the only dimensionality reduction capabilities of this technique. In the specific context of machine learning, RPs may be considered a tool for generating diversity in the creation of an ensemble of classifiers. Using RPs, the different embeddings of the original feature space provide multiple view of the original features space. This diversity can be generated projecting data into subspaces, space having the same dimensionality of the features spaces or spaces with dimensionality higher than the original space.

2.1 Embedding Random Projections in Boosting Ensemble

AdaBoost [Freund and Schapire(1999)] represents the most popular meta-algorithm[1] to ensemble a set of classifiers. Boosting is based on the observation that finding many rough classification rules can be much easier than finding a single, highly accurate prediction rule. To apply the boosting approach, we start with a method or algorithm for finding those rough rules. The boosting algorithm calls this "weak" learning algorithm repeatedly. Each time it is called, the base learning algorithm generates a new weak prediction rule and, after many rounds, the boosting algorithm must combine these weak rules into a single prediction rule that, hopefully, will be much more accurate than any one of the weak rules.

From the point of view of incremental optimization, AdaBoost represents an additive model fitting procedure that approximates the optimization of an exponential loss function. Changing the exponential loss function with a least square loss function yields to a new model of boosting, known as LsBoost [Friedman(2000)]. Gradient Boosting Machines (GBMs) generalize this idea for any arbitrary loss function.

In this section, a methodology for embedding Random Projections into the incremental optimization process of GBM will be presented. The stepwise approximation will be obtained projecting data into random spaces at every step of the optimization process and looking for the classifier that best fits the data in the projected space.

2.1.1 Related works on Random Projections in Machine Learning

Random Projections have been widely used in pattern recognition applications as a tool for dimensionality reduction [Achlioptas(2001)],[Dasgupta(2000)], [Fradkin and Madigan(2003)],[Bingham and Mannila(2001)].

[Arriaga and Vempala(2006)] outline the basic learning algorithm based on Random Projections consisting of two simple steps: project the data in a random subspace and run the algorithm in that space, taking advantage of working faster with a lower dimensionality. This simple algorithm represents the baseline for embedding RPs into GBMs.

[Dasgupta(2000)] uses RPs with Gaussian mixture models for classification of both synthetic and real data. In his work, data are projected into a randomly chosen d-dimensional subspace and the learning algorithm works in this new smaller space achieving highly accurate classification results. [Fradkin and Madigan(2003)] compare the performances of C4.5, Nearest Neighbors and SVM using both Principal Component Analysis and RPs as dimensionality reduction technique. Their experimental results demonstrate the advantages of using PCA in almost all the considered

[1]A *meta-algorithm* is an algorithm that can be usefully considered to have other significant algorithms, not just elementary operations and simple control structures, as its constituents.

test cases.

Some works also suggest that RPs are a powerful tool even for non linearly separable classification problems. For example, [Rahimi and Recht(2008)] use RPs for building a weighted sum of linear separators. Authors show that using RPs is equivalent to use the kernel trick. At the same time, authors state that embedding RPs into the weighted sum of linear separators provides a faster decaying of the testing error rate with respect to standard AdaBoost.

Others works, like [Blum(2006)],[Balcan et al.(2006)], suggest that RPs can help in the features selection process and provide specific insights in the construction of large margin classifiers. [Blum(2006)] reports some basic algorithms showing that if the learning problem is separable with a large margin, then the problem still remains separable in the reduced random space. Moreover, even picking a random separator on data projected down to a line, provides a reasonable chance to get a weak hypothesis as well.

2.1.2 Gradient Boosting Machines

In regression and classification problems, given a set of training sample $\{y_i,\mathbf{x}_i\}_1^N$, we look for a function $F^*(\mathbf{x})$ that maps $\mathbf{x}$ to y such that, over the joint distribution of all $(y, \mathbf{x})$-values, the expected value of some specified loss function $\Psi(y, F(x))$ is minimized. Usually, the function $F(\mathbf{x})$ is member of parameterized class of functions $F(\mathbf{x}; \mathbf{P})$:

$$F(\mathbf{x}; \mathbf{P}) = \sum_{m=0}^{M} \beta_m h(\mathbf{x}; \mathbf{a}_m) \tag{2.1}$$

where $\mathbf{P} = \{\beta_m, \mathbf{a}_m\}\ _0^M$ is a set of parameters. Nevertheless, we can consider $F(\mathbf{x})$ evaluated at each point $\mathbf{x}$ to be a parameter and minimize:

$$\Phi(F(\mathbf{x})) = E_y[\Psi(y, F(\mathbf{x})|\mathbf{x})], \tag{2.2}$$

at each individual $\mathbf{x}$, directly with respect to $F(\mathbf{x})$. The solution is of the type:

$$F^*(\mathbf{x}) = \sum_{m=0}^{M} f_m(\mathbf{x}), \tag{2.3}$$

where $f_0(\mathbf{x})$ is an initial guess, and $\{f_m\}_1^M$ are incremental functions, known as "steps" or "boosts". Using steepest-descent, we get :

$$f_m(\mathbf{x}) = -\varrho_m g_m(\mathbf{x}), \tag{2.4}$$

where, assuming that differentiation and integration can be interchanged,

$$g_m(\mathbf{x}) = E_y \left[\frac{\partial \Psi(y, F(\mathbf{x}))}{\partial F(\mathbf{x})} |\mathbf{x} \right]_{F(\mathbf{x})=F_{m-1}(\mathbf{x})} \tag{2.5}$$

and

$$F_{m-1}(\mathbf{x}) = \sum_{i=0}^{m-1} f_i(\mathbf{x}). \tag{2.6}$$

When the joint distribution of $(y, \mathbf{x})$ is represented by a finite data sample, $E_y[\cdot|\mathbf{x}]$ cannot be evaluated accurately at each $\mathbf{x}_i$ and, if we could perform parameter optimization, the solution is difficult to obtain. In this case, given the current approximation $F_{m-1}(\mathbf{x})$ at the m-th iteration, the function $\beta_m h(\mathbf{x}; \mathbf{a})$ is the best greedy step towards the minimizing solution $F^*(\mathbf{x})$, under the constraint that the step direction $h(\mathbf{x}, \mathbf{a}_m)$ be a member of the parameterized class of functions $h(\mathbf{x}, \mathbf{a})$. One possibility is to choose the member of the parameterized class $h(\mathbf{x}; \mathbf{a})$ that is most parallel in the N-dimensional data space with the unconstrained negative gradient $\{-g_m(\mathbf{x}_i)\}_1^N$. In this case, it is possible to use $h(\mathbf{x}, \mathbf{a}_m)$ instead of the unconstrained negative gradient $-g_m(\mathbf{x})$. Weights ϱ_m are given by the following line search:

$$\varrho_m = argmin_{\varrho} \sum_{i=1}^{N} \Psi(y_i, F_{m-1}(\mathbf{x}_i) + \varrho h(\mathbf{x}_i; \mathbf{a}_m)) \tag{2.7}$$

and the approximation updated in the following way:

$$F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \varrho_m h(\mathbf{x}; \mathbf{a}_m). \tag{2.8}$$

When $y \in \{-1, 1\}$ and the loss function $\Psi(y, F)$ only depends on y and F only through their product $\Psi(y, F) = \Psi(yF)$, the algorithm reduces to the original boosting formulation. If the loss function is $\Psi(y, F) = \frac{(y-F)^2}{2}$, gradient boosting produces the stage-wise approach of iteratively fitting the current residuals. Algorithm 1, describing this situation, is called LsBoost. Generally, direct optimization of ill-posed problems usually yields to poor results. The influence of each base classifier in the ensemble is governed by the weighting vector ϱ and its optimization can be formulated as an L2 regularization problem modifying line 4 in Algorithm 1 as follows

$$(\varrho_m, \mathbf{a}_m) = argmin_{\mathbf{a},\varrho} \sum_{i=1}^{N} [\tilde{y}_i^m - \varrho h(\mathbf{x}_i; \mathbf{a})]^2 + \lambda^2 \|\varrho\|_2^2 \quad s.t. \quad \varrho > 0 \tag{2.9}$$

In the next section, an analytical solution for the L2 regularization problem will be used jointly to RPs in the construction of the ensemble of classifiers.

2.1.3 The Random Projections Method

[Johnson and Lindenstauss(1984)] showed the theoretical foundation of RPs. This lemma states that, given m points in $\Re^n$, it is possible to project these points into a d-dimensional subspace, with $d = O(\frac{1}{\gamma^2} log(m))$. In this space, relative distances and angles between all pairs of points are approximately preserved up to $1 \pm \gamma$, with high probability.

```
    Input: A set of training sample $\{y_i, \mathbf{x}_i\}_1^N$
    Output: A classifier $F(\mathbf{x})$
1   $F_0(\mathbf{x}) = argmin_{\varrho} \sum_{i=1}^{N} \Psi(y_i, \varrho)$;
2   foreach $m \in \{1..M\}$ do
3       $\tilde{y}_i^m = y_i - F_{m-1}, i = 1, N$;
4       $(\varrho_m, \mathbf{a}_m) = argmin_{\mathbf{a},\varrho} \sum_{i=1}^{N} [\tilde{y}_i^m - \varrho h(\mathbf{x}_i; \mathbf{a})]^2$;
5       $F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \varrho_m h(\mathbf{x}_i; \mathbf{a}_m)$;
6   end
```

Algorithm 1: LsBoost Algorithm

Formally, given $0 < \gamma < 1$, a set X of m points in $\Re^N$, and a number $n > n_0 = O(\frac{1}{\gamma^2} log(m))$, there is a Lipschitz function $f : \Re^N \rightarrow \Re^n$ such that

$$(1-\gamma)\|u-v\|^2 \leq \|f(u)-f(v)\|^2 \leq (1+\gamma)\|u-v\|^2. \tag{2.10}$$

If m data points in a features space are considered row-vectors of length N, the projection can be performed by multiplying all the m points by a randomly generated $N \times m$ matrix. The random matrix should be one of the following types:

- P with columns to be random orthonormal vectors;
- $U_{-1,1}$ with each entry to be 1 or -1 drawn independently at random;
- $N_{0,1}$ with each entry drawn independently from a standard Normal Distribution $N(0,1)$.

While using these types of projections ensures that relative distances and angles are approximately preserved, there is no guarantee that using other types of matrices could preserve the structure of the data. Data can be projected down to lower dimensions, to space having the same dimension than the original problem or, even, spaces with higher dimension. The projection of the data to spaces of higher dimension does not rely on any theoretical results. Here, the possibility to project data to super-spaces is also taken into account.

2.1.4 Embedding Random Projections in Boosting Machines

The original algorithm of LsBoost has been modified to be adapted to embedding RPs into the boosting algorithm. In point 4 of Algorithm 1, it is possible first searching analytically for the optimal set of weighting values for each candidate classifier and after to select the classifier that best approximates the negative gradient with the correspondent precomputed weight [Pujol(2010)]. It is possible to find the optimal weighting value for each candidate classifier $h(x;a)$ by:

$$\frac{\partial[(\tilde{\mathbf{y}}^m - \varrho_a^T h(x;a))^T (\tilde{\mathbf{y}}^m - \varrho_a^T h(x;a))]}{\partial \varrho_a} = 0 \tag{2.11}$$

solved by

$$\tilde{\mathbf{y}^m}^T h(x;a) = \varrho_a^T h(x;a)^T h(x;a) = \varrho_a^T N. \tag{2.12}$$

Since $h(x;a) \in \{+1,-1\}$, the dot product $h(x;a)^T h(x;a)$ is just the number of training examples. In this way, the regularization parameter can be simply taken into account by addition. The optimal set of weights is given by Eq.(2.13)

$$\varrho_m = \frac{\tilde{\mathbf{y}}^m A}{N+\lambda}, \tag{2.13}$$

where $\tilde{\mathbf{y}}^m$ denotes the vector of residuals at step m, A is the matrix of training examples, N is the number of training examples and λ represents the L2 penalization term. If $\lambda = 0$, regularization is not taken into account. Once the optimal set of values is found, a simple selection of the classifier that best approximates the negative gradient can be performed. The described procedure can be performed on training data projected onto random spaces. This modified version, called **RpBoost** is defined as Regularized Gradient Boosting where data are projected using a transformation represented by a specific RPs technique. Algorithm 2 defines the steps of the proposed method. Finally, the following three classifiers are defined:

Definition 1. *RpBoost.sub is defined as RpBoost working on data projected down-to a random space with dimension lower than the original space;*

Definition 2. *RpBoost.same is defined as RpBoost working on data projected to a random space with the same dimension of the original space;*

Definition 3. *RpBoost.super is defined as RpBoost working on data projected to a random space with dimension upper than the original space.*

```
Input: The type of projection, the dimension of the random space
Output: A classifier $F(\mathbf{x})$
1 $F_0(\mathbf{x}) = argmin_\varrho \sum_{i=1}^{N} \Psi(y_i, \varrho)$;
2 foreach $m \in \{1..M\}$ do
3     $\tilde{y}_i^m = y_i - F_{m-1}, i = 1, N$;
4     Set a new $R_p$;
5     $A_r = A \cdot R_p$;
6     $\varrho_m = \frac{\tilde{y}_m A_r}{N+\lambda}$;
7     $\mathbf{a}_m = argmin_\mathbf{a} \sum_{i=1}^{N} [\tilde{y}_i^m - \varrho_m h(\mathbf{x}_i; \mathbf{a})]^2$;
8     $F_m(\mathbf{x}) = F_{m-1}(\mathbf{x}) + \varrho_m h(\mathbf{x}_i; \mathbf{a}_m)$;
9 end
```

Algorithm 2: RpBoost Algorithm

2.1.5 Experimental Results

The performances of RpBoost will be evaluated on synthetic and real problems. RpBoost will be compared with AdaBoost and LsBoost on both set of problems. Decision stumps are used as weak classifiers. The maximum dimension of the ensemble has been arbitrarily set to 500 classifiers. In order to have a straightforward comparison, the dimensions of both projection subspace and super-space will be equals to the half and the double of the dimension of the features space, respectively. The effect of using the three types of projections will be evaluated and compared. Classification accuracy is used as performance measure.

Synthetic problems: Test Patterns

Test patterns are synthetic bi-dimensional datasets proposed by [Fawcett(2010)] to be used for comparing classifiers. The patterns are randomly generated on a two dimensional grid of points, in the ranges [0:4] x [0:4] with a resolution of 0.05, yielding 6561 total points. The points are labeled according to their position in the pattern. In Table 2.1, the formal description of all the used patterns is reported. In Figure 2.1, examples of two patterns are shown.

Validation Procedure: For each test pattern, a stratified sample of 1000 points is used for training and the complete distribution for testing. The validation procedure has been performed five times and results averaged.

Results: Results obtained using projection of type P and projections of type $U_{-1,1}$ are reported in Figure 2.2 and Figure 2.3. RpBoost performs slightly better than AdaBoost and LsBoost on some particulars patterns. RpBoost.sub with P pro-

Table 2.1: Test Patterns

Test Pattern	**Description**
Sine	$Y = 0.84sin(1.78X)$
Linear	$Y = 1.87 * X \pm 1.74$
Parity	9 parity circles
Annulus	Annulus at (2.00, 2.00)
Parabolic	$Y = \frac{(X-2)^2}{4*0.25+1}$
Disjunctive	4 disjoint concave polygons
Polynomial	$Y = \frac{1}{2} * (x-2)^3 + \frac{1}{2} * (x-2.2)^2 + 2$
Checkerboard	9 squares alternating classes

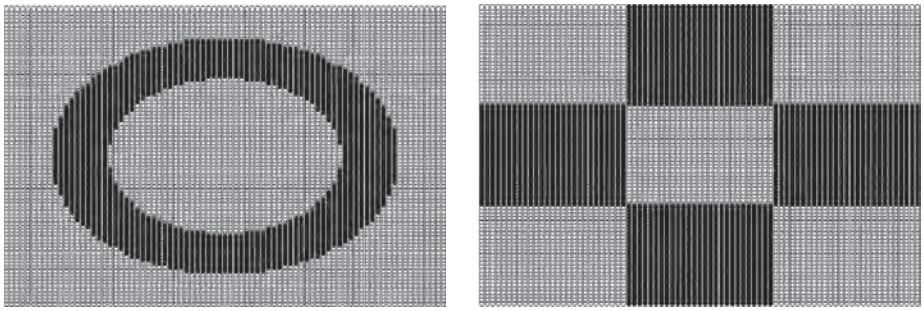


Figure 2.1: Test Patterns: (a) *Annulus*; (b) *Checkerboard*

jections always performs considerably worse than RpBoost.same and RpBoost.super. RpBoost.super with $U_{-1,1}$ projections always performs worst the RpBoost.same and RpBoost.sub. In the *Parity* and in the *Checkerboard* test patterns, RpBoost always performs better than the compared methods, independently from the type of projection. In Figure 2.4, results obtained with $N_{0,1}$ projections are shown. The correspondent numerical values are reported in Table 2.2. RpBoost.sub and RpBoost.same always have the best accuracy. In *Annulus* and *Checkerboard*, the performances are considerably improved.

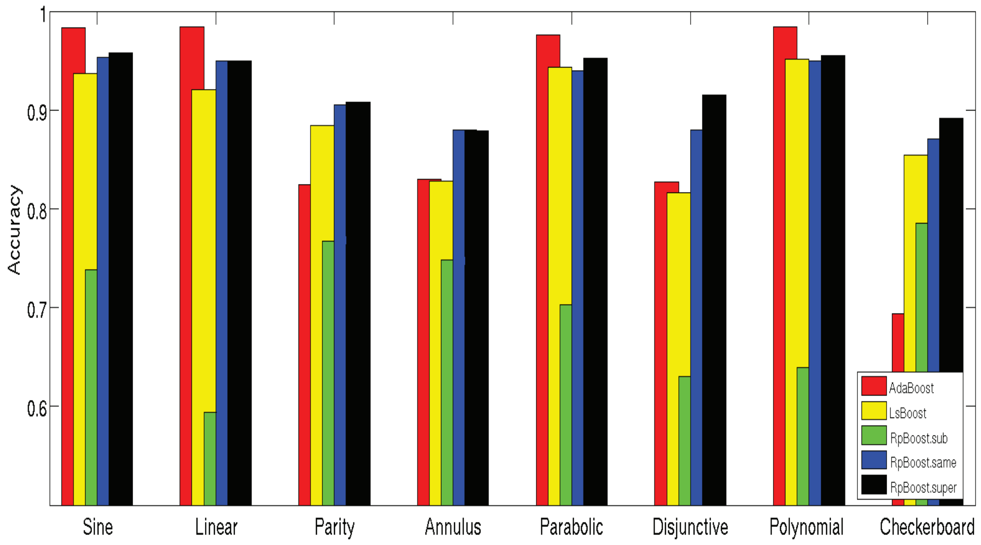


Figure 2.2: Test Patterns : Comparative Results with RpBoost using projections of type P

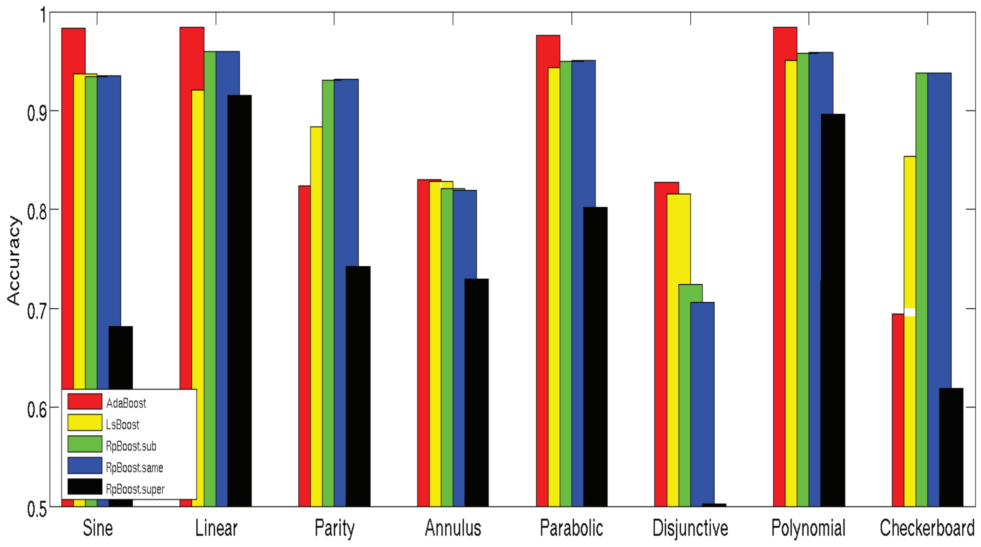


Figure 2.3: Test Patterns : Comparative Results with RpBoost using projections of type $U_{-1,1}$

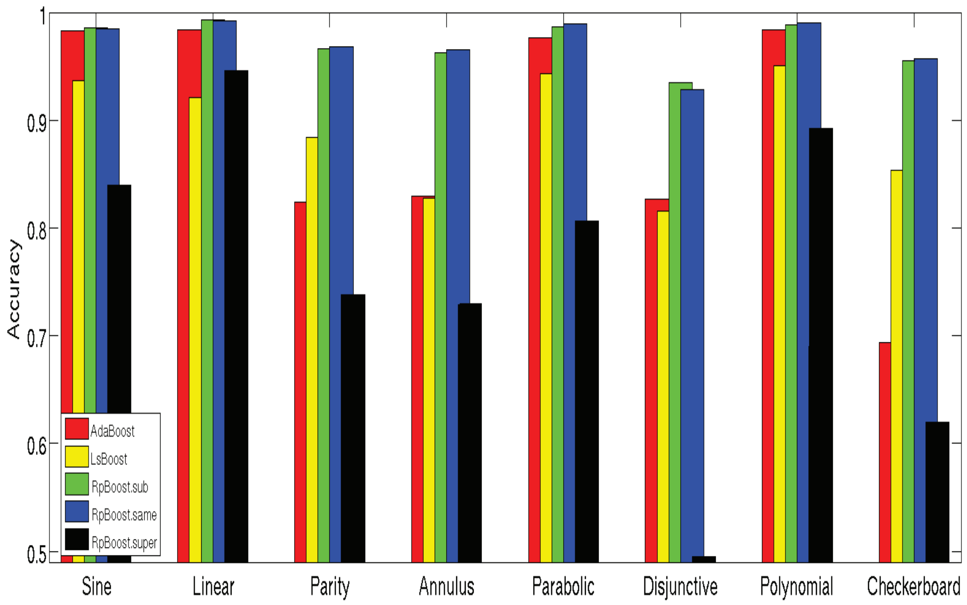


Figure 2.4: Test Patterns : Comparative Results with RpBoost using projections of type $N_{0,1}$

Table 2.2: Numerical Values of Accuracy obtained on Test Pattern with RpBoost working on $N_{0,1}$ projections

	AdaBoost	LsBoost	RpBoost.sub	RpBoost.same	RpBoost.super
Sine	0.983	0.937	**0.986**	0.985	0.840
Linear	0.984	0.921	**0.993**	0.992	0.946
Parity	0.824	0.884	0.966	**0.968**	0.738
Annulus	0.83	0.828	0.963	**0.965**	0.730
Parabolic	0.976	0.943	0.987	**0.989**	0.806
Disjunctive	0.827	0.816	**0.935**	0.928	0.495
Polynomial	0.984	0.951	0.988	**0.990**	0.892
Checkerboard	0.694	0.854	0.955	**0.957**	0.620

Table 2.3: List of UCI problems used to evaluate RpBoost

Dataset	**Elements**	**Dataset**	**Elements**
Monks-1	272,284	Liver	100,245
Monks-2	300,301	Tic-Tac-Toe	626,332
Monks-3	275, 279	Ionosphere	126,225
Breast	239,485	Sonar	97,111

UCI Datasets

Eight datasets from UCI Repository [Frank and Asuncion(2010)], reported in Table 2.7, have been used to evaluate the performances of RpBoost. The cardinality of each class of the dataset is also reported. All the datasets selected are binary classification problems.

Validation Procedure Results have been obtained using two rounds of 10-folds cross-validation and averaging the results. The value of the regularization parameter λ has been selected using generalized 5-fold cross validation on the training set for $\lambda \in \{1, 5, 10, 50, 100, 500, 1000, 5000, 10000\}$.

Results Comparative results for RpBoost are reported in Figure 2.5 and in Figure 2.6 for projections of type P and projections of type $U_{-1,1}$, respectively. As for synthetic data, there exist problems where RpBoost outperforms AdaBoost and LsBoost. RpBoost.super provides better classification accuracies only when using projections of type P. In Figure 2.7, comparative results of RpBoost using projections of type $N_{0,1}$ are shown. Numerical values are reported in Table 2.4. In Monks-1 and Monks-2, RpBoost outperforms AdaBoost and LsBoost. A slight improvement can be noted in Breast as well. The slight improvements on Sonar and Ionosphere, datasets with the highest dimensions, do not seem to completely benefit of the dimensionality reduction that RPs provide.

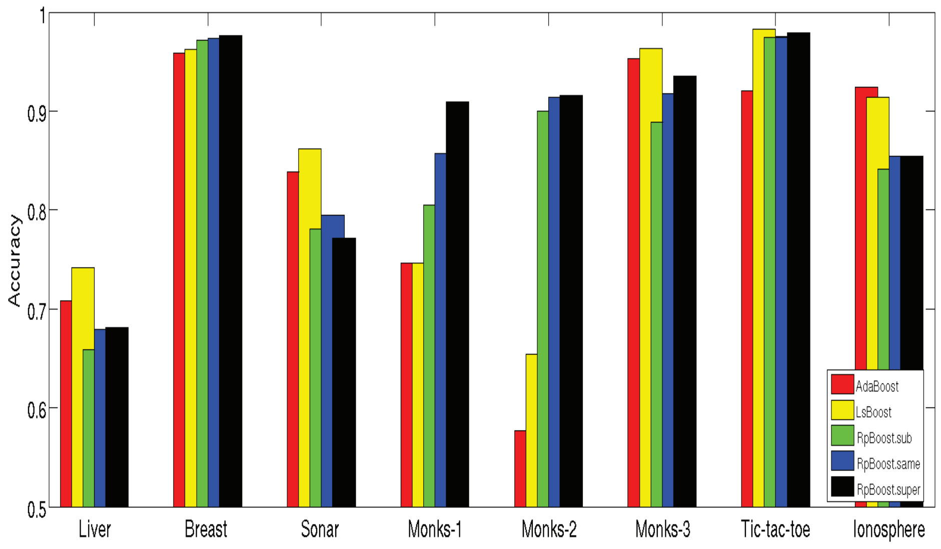


Figure 2.5: UCI Datasets: Comparative Results with RpBoost using projections of type P

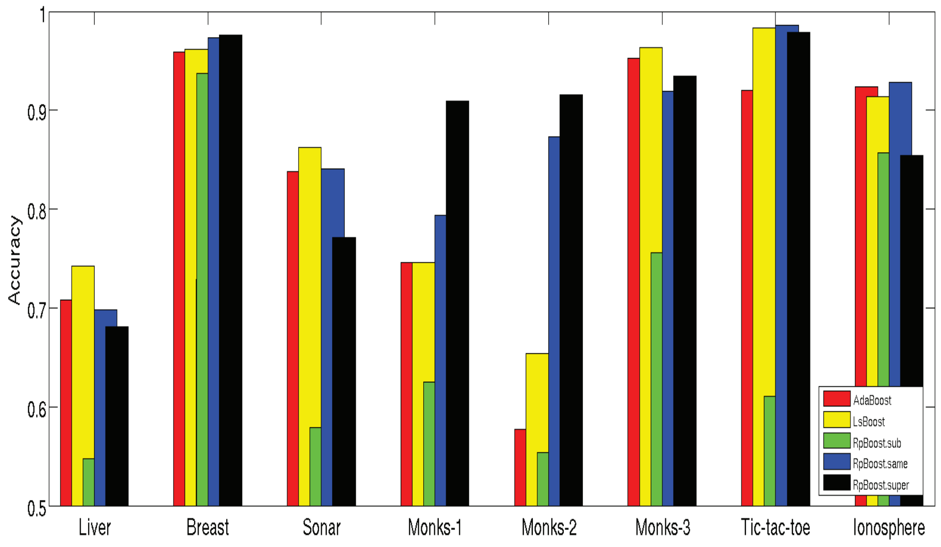


Figure 2.6: UCI Datasets: Comparative Results with RpBoost using projections of type $U_{-1,1}$

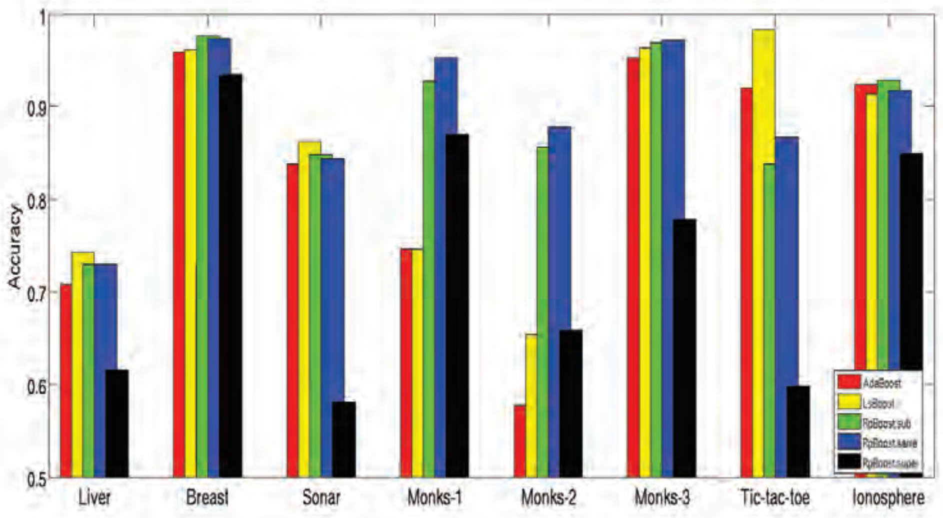


Figure 2.7: UCI Datasets: Comparative Results with RpBoost using projections of type $N_{0,1}$

Table 2.4: Numerical Values of Accuracy obtained on UCI datasets with RpBoost working on $N_{0,1}$ projections

	AdaBoost	LsBoost	RpBoost.sub	RpBoost.same	RpBoost.super
Liver	0.708	**0.742**	0.692	0.698	0.547
Breast	0.959	0.962	**0.974**	0.973	0.937
Sonar	0.839	**0.862**	0.829	0.841	0.579
Monks-1	0.746	0.746	**0.796**	0.794	0.625
Monks-2	0.577	0.654	0.872	**0.873**	0.554
Monks-3	0.953	**0.963**	0.913	0.919	0.756
Tic-tac-toe	0.920	0.983	0.983	**0.986**	0.611
Ionosphere	0.924	0.914	0.921	**0.928**	0.857

Table 2.5: Resume of Results Obtained on Test Patterns

Test Pattern	Accuracy	Classifier	R_p
Sine	98.6%	RpBoost.sub	N
Linear	99.3%	RpBoost.sub	N
Parity	96.8%	RpBoost.same	N
Annulus	96.5%	RpBoost.same	N
Parabolic	98.9%	RpBoost.same	N
Disjunctive	93.5%	RpBoost.sub	N
Polynomial	99.0%	RpBoost.same	N
Checkerboard	95.7%	RpBoost.same	N

Table 2.6: Resume of Results Obtained on UCI Datasets

Dataset	Accuracy	Classifier	R_p
Liver	74.2%	LsBoost	-
Breast	97.6%	RpBoost.super/sub	P N
Sonar	86.2%	LsBoost	-
Monks-1	95.3%	RpBoost.same	N
Monks-2	91.6%	RpBoost.super	P
Monks-3	97.2%	RpBoost.same	N
Tic-tac-toe	98.6%	RpBoost.same	U
Ionosphere	92.8%	RpBoost.same/sub	U N

Discussion on Results Obtained

Table 2.5 and Table 2.6 report the highest accuracies obtained on the set of problems considered. For each one, the value of accuracy, the classifier providing such a value and the type of projection R_p, if used, are reported. RpBoost always provides the best accuracy on synthetic data using projections drawn from a normal distribution. Significant improvements are experienced in *Annulus* but also in *Checkerboard*, *Parity* and *Disjunctive* where performance are incremented of more than 10%. In Figure 2.8, the classification of *Annulus* is shown. Although classification is not highly accurate, the effect of using RPs is evident. RPs allow to follow the non linear boundary even when weak linear classifiers as decision stumps are used. Figure 2.9 shows the classification of *Checkerboard*. Here, differently from other classifiers, RpBoost is capable to grasp the different class in the central part of the pattern. *Checkerboard* represents a typical XOR-type problem that linear classifiers are not able to solve. Similarly, *Parity* and *Disjunctive* represent XOR-type problems. This fact is confirmed in the Monks-1 and Monks-2 datasets, both representing XOR-type problems [Thrun et al.(1991)]. In all these cases, the classification accuracy is considerably improved compared to the performance obtained from AdaBoost and LsBoost.

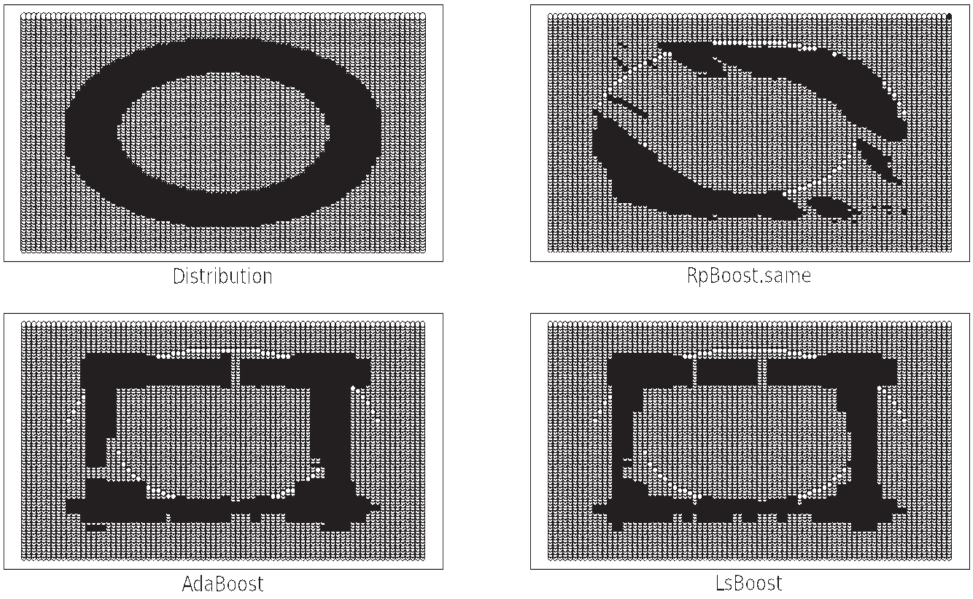


Figure 2.8: Classification of the *annulus* test pattern using RpBoost, AdaBoost and LsBoost

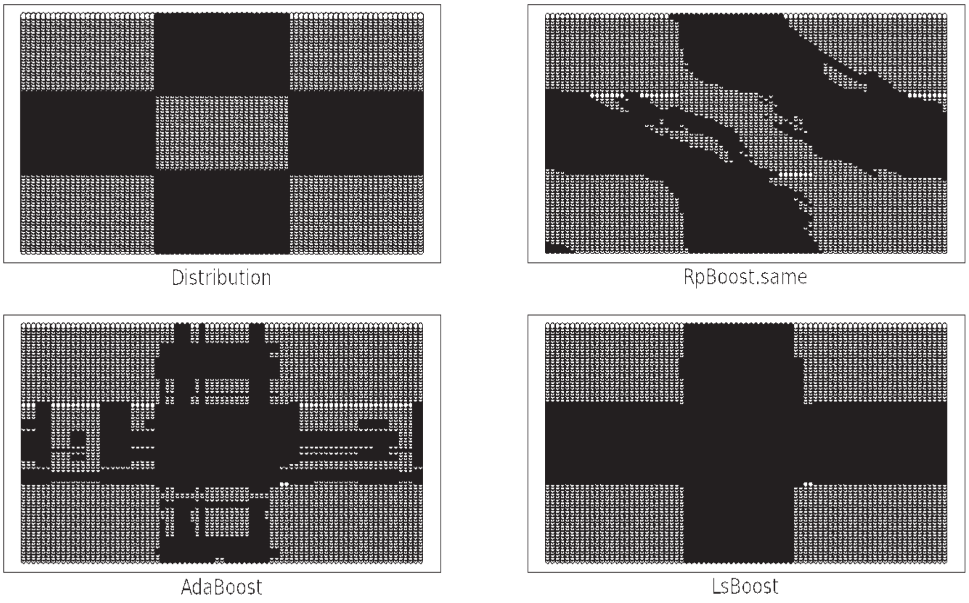


Figure 2.9: Classification of the *checkerboard* test pattern using RpBoost, AdaBoost and LsBoost

2.1.6 Discussion on Random Projections

A preliminary study on Random Projections has been conducted in this first section. Random Projections have been used for dimensionality reduction with the aim, at the same time, to generate diversity in the construction of Boosting Machines. This embedding has generated the definition of a new algorithm called RpBoost. At each step of the boosting optimization process, data are projected in a random space and, in this new space, the classifier that best fits the data is selected and added to the ensemble. Projections can be performed to random spaces having the same dimension of the original features space, but also to lower and higher dimensional spaces.

RpBoost always has the best performances on synthetic data and, in the majority of the cases, on real problems too. Performances are good especially when projections to subspaces or space of the same dimensionality than the original spaces are used. With these spaces, RpBoost performs well with all the types of projections on most of the problems. The use of super-spaces yields to better classification accuracy only when the projection is drawn completely at random. In this case, the performance appears to be slightly better than other types of projections. Nevertheless, results redoubtably highlights the fact that the type of projection providing the best results is $N_{0,1}$ that is, projections drawn from a standard Normal Distribution $N(0, 1)$. This type of projections provide always the best results in the artificial datasets and in the majority of the cases of the real problems.

When methodologies are not corroborated by solid theoretical foundations, as in the case of Random Projections, these empirical results may be useful and significant guidelines. In the next section, a new ensemble algorithm for one-class classification is presented where Random Projections represent the fundamental methodology aiming to reduce the complexity of the classification problem at hand. Thanks to the experience acquired, we naturally decide to use projection of type $N_{0,1}$ when using Random Projections.

2.2 Approximate Polytope Decision Ensemble for One-Class Classification

In pattern recognition, a particular class of problems is defined when only data related to a target are available. This typology of problems is known as One-Class classification. These classification tasks naturally arise when target data can be effortlessly collected while counterexamples are scarce or difficult to obtain
[Kulikowski et al.(1999)]. Typical one-class problems are the prediction of mean time before failure of a machinery [Kulikowski et al.(1999)], [Tax(2001)] where examples of non-regular operations can only be found in presence of cracks and malfunctions or the problem of authorship verification [Koppel and Schler(2004)] where, while it is possible to easily provide all the examples necessary to model the author taken into account, it is pretty hard to define a proper sampling of all the possible similar authors. Effective One-Class classification strategies use density estimation methods or boundary methods to model the target class. Gaussian Model [Bishop(1995)], Mixture of Gaussian Model [Duda and Hart(1973)] and Parzen Density Estimation [Parzen(1962)] are density estimation methods widely used. Density estimation methods work well when there exists *a-priori* knowledge of the problem at hand or a big load of data is available. Boundary methods only intend to model the boundary of the problem disregarding the underlying distribution. Well known approaches to boundary methods are k-Centers [Ypma et al.(1999)] and Nearest Neighbors Method [Duda and Hart(1973)], [Tax(2001)]. Support Vectors Data Description (SVDD) [Tax(2001)] represents the state of the art in One-Class classification. SVDD computes the minimum hypersphere containing all the data in a multidimensional space, providing an elegant and intuitive understanding about the solution of the classification problem. Indeed, many classification problems can be solved efficiently when looked at from the geometrical point of view. In this geometrical framework, the smallest polytope containing the full set of points, i.e. *convex hull*, represents an even more general structure than hypersphere.

The convex hull has always been considered a powerful tool in geometrical pattern recognition [Bhattacharya(1982)],[Toussaint(1978)].
Recent researches [Bennett and Bredensteiner(2000)], [Bi and Bennett(2001)] show that there exists a geometrical interpretation of the Support Vector Machine (SVM) [Vapnik(1995)] related to the convex hull. Finding the maximum margin between two classes is equivalent to find the nearest neighbors in the convex hull of each class when classes do not overlap. This intuitive explanation provides an immediate visualization of the main concepts of SVM from a geometrical point of view. Nevertheless, using the convex hull in real applications is limited by the fact that its computation in a high dimensional space has an extremely high cost. Many approximations have been proposed in order to circumvent this problem. For instance, [Takahashi and Kudo(2010)] use the convex hull as a maximum margin classifier. They approximate the facets of the convex hull where the support planes are represented by a set of reflexive support functions separating one class from the other ones. [Pal and Bhattacharya(2007)] propose a self-evolving two-layers neural network model for computing the approximate convex hull of a set of points in 3-D and spheres. The vertices of the convex hull

are mapped with the neurons of the top layer. [Mavroforakis and Theodoridis(2006)] introduce the notion for Reduced Convex Hull based on the geometric interpretation of SVM, formalizing the case when classes are not separable.

In this section, a new algorithm is introduced for using shrunk/enlarged versions of the convex hull of the training data in order to model the boundary of one-class classification problems. The geometrical entities that define the boundary are called *Extended Convex Polytopes* and their growth is governed by a parameter α. Using this model, a point is said to belong to the class if it lies inside the extended convex polytope. However, the creation of an exact extended convex polytope is computationally unfeasible in high dimensional spaces. Random projections represent a suitable tool to approximate the original multidimensional convex hull and to reduce its computational complexity. Therefore, the computational limitation derived from computing the convex hull in high dimensional spaces is circumvented by approximating the d-dimensional expanded convex polytope decision by an ensemble of decisions in low-dimensional ($<< d$) spaces, i.e. *Approximate Convex Polytope Decision Ensemble.* In those low-dimensional spaces, computing the convex hull and establishing whether points belong to the geometric structure are both well known problems having very efficient solutions [Preparata and Shamos(1985)].

2.2.1 Convex Approximate Polytope Ensemble

One-class classification can be performed by modeling the boundary of the set of points defining the problem. If the boundary encloses a convex area, then the convex hull, defined as the minimal convex set containing all the training points, provides a good general tool for modeling the target class. The **convex hull** of a set $C \subseteq \Re^n$, denoted as $\mathbf{conv}\, C$, is the smallest convex set that contains C and is defined as the set of all convex combinations of points in C:

$$\mathbf{conv}\, C = \{\theta_1 x_1 + \cdots + \theta_m x_m \,|\, x_i \in C, \theta_i \geq 0, \forall i; \sum_i \theta_i = 1\}$$

In this scenario, the one-class classification task is reduced to the problem of knowing whether test data lie inside or outside the hull. Although the convex hull provides a compact representation of the data, a small amount of outliers may lead to very different shapes of the convex polytope. Thus, a decision using these structures is prone to over-fitting. In this sense, it is useful to define a parameterized set of convex polytopes associated with the original convex hull of the training data. These polytopes are shrunk/enlarged versions of the original convex hull governed by a parameter α. The goal of this family of polytopes is to define the degree of robustness to outliers. This allows to set the operating point in the receiver-operating characteristic i.e. ROC curve. Given the set $C \subseteq \Re^n$, we define the **extended convex polytope** with respect to the center c and with expansion parameter α as

$$v_\alpha : \{v + \alpha \frac{(v-c)}{\|v-c\|} \,|v \in \mathbf{conv}\, C\} \tag{2.14}$$

Observe that the parameter α defines a constant shrinking $(-\|v-c\| \leq \alpha \leq 0)$ or enlargement $(\alpha \geq 0)$ of the convex structure with respect to the center c. If $\alpha = 0$ then $v_0 = \mathbf{conv}\, C$. Figure 2.10 illustrates a shrunk and enlarged expanded convex polytope. The light gray convex polytope represents the original convex hull with vertices $\{v_i, i = 1 \ldots 5\}$. The dark gray polytope corresponds to the enlargement of the original convex hull using $\alpha > 0$. The white polytope corresponds to a shrunk version of the convex hull using $\alpha < 0$. Two fundamental limitations exist in the suggested

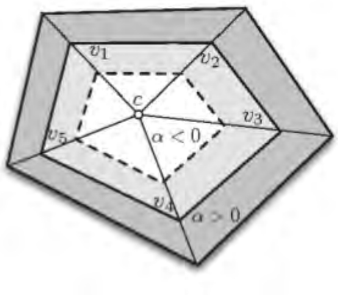


Figure 2.10: Illustration of the expanded convex polytope in the 2D space

approach. In high dimensional spaces, the task of computing the *extended convex polytope* and testing if a point belongs to its interior is computationally unfeasible. In the following section, an approximate solution to this problem is introduced.

Approximate convex polytope decision ensemble

The creation of high-dimensional convex hulls is computationally intensive. In general, the cost for computing a d-dimensional convex hull on N data examples is $\mathcal{O}(N^{\lfloor d/2 \rfloor + 1})$. This cost is prohibitive in time and memory and, for the classification task, only checking if a point lies inside the multidimensional structure is needed. The **Approximate convex Polytope decision Ensemble (APE)** consists in approximating the decision made using the **extended convex polytope** in the original d-dimensional space by aggregating a set of t decisions made on low-dimensional random projections[2] of the data. The approximation is based on the observation that the vertices defining a convex polytope in a low-dimensional projection of the data set correspond to a subset of the projected vertices belonging to the convex polytope in the original d-dimensional space. With this observation in mind, we define the **decision rule** as follows: given a set of T randomly projected replicas of the training set, a point does belong to the modeled class if and only if the point lies inside the corresponding projected convex polytope – the polytope corresponding to the projected data set – in all projections.

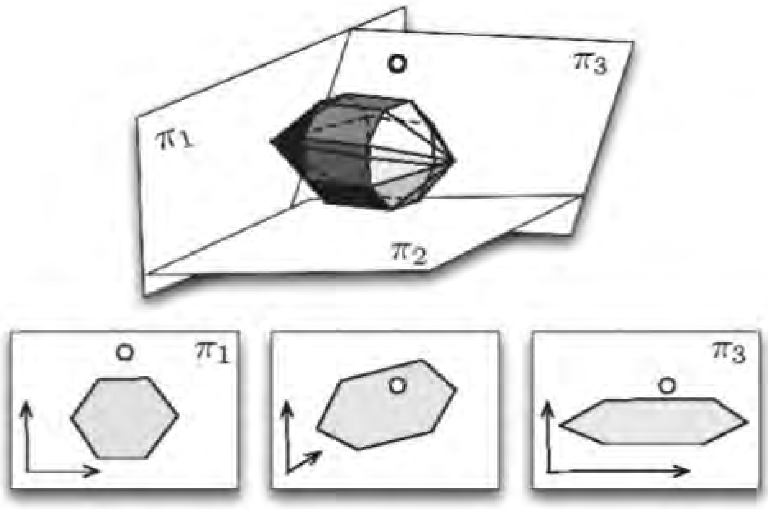


Figure 2.11: Illustration of the approximate convex polytope decision strategy

[2]As previously stated, random projections drawn by a normal distribution are those providing the best results. In the rest of this section, projections will be drawn from a normal distribution with zero mean and standard deviation 0.3, ensuring in this way that the elements of the projection matrix are in the range $[-1, 1]$ with high probability.

In Figure 2.11, the method is shown graphically. Here, a three-dimensional convex polytope and a test point laying outside of the hull is presented. At the bottom, three random projection planes are displayed. The point does not belong to the modeled class if there exists at least one projection in which the point is outside of the projected convex polytope. Note that a testing point outside the original structure might appear inside of some projections.

The expansion factor in random spaces

In order to decide if a point belongs to the target class, the *extended convex polytope* is used. It is worth to remind that this structure is a shrunk or expanded version of the convex hull governed by the parameter α. The *approximate convex polytope decision strategy* relies in creating the expanded polytope in a low-dimensional space. Since the projection matrix is created at random, the resulting space does not preserve the norm of the original space. Hence, a constant value of the parameter α in the original space corresponds to a set of values γ_i in the projected one. As a result, the low-dimensional approximation of the expanded polytope is defined by the set of vertices as follows

$$\bar{v}^{\alpha} : \left\{ \bar{v_i} + \gamma_i \frac{(\bar{v_i} - \bar{c})}{||\bar{v_i} - \bar{c}||} \right\} \tag{2.15}$$

where $\bar{c} = Pc$ represents the projected center, $\bar{v_i}$ is the set of vertices belonging to the convex hull of the projected data and γ_i is defined as follows,

$$\gamma_i = \frac{(v_i - c)^T P^T P(v_i - c)}{||v_i - c||_2} \alpha \tag{2.16}$$

where P is the random projection matrix, c is the center and v_i is the ith vertex of the convex hull in the original space. Note that there exists a different expansion factor for each vertex v_i belonging to the projected convex hull. Figure 2.12 illustrates the difference between the constant expansion of the original polytope and the different gamma parameters in the projected counterpart. The original polytope lies in a 2-dimensional space and the corresponding projected convex polytope in a 1-dimensional space. The figure clearly shows that though the convex polytope is expanded by a constant value α in the original space, different expansion parameters γ_i are needed in the projected space.

Approximate convex polytope decision ensemble learning algorithms

Algorithms 3 and 4 describe the steps needed for learning and testing the APE, respectively. Both algorithms require defining the number of projections T for approximating the original convex polytope. In Algorithm 3, at each iteration, we first create a random matrix. Then, the training set is projected into the space spanned by the random projection matrix. Finally, the vertices of the convex hull of the projected data set are found. The low-dimensional destination spaces is $\Re^2$. Algorithm 4 describes the test procedure. At each iteration t of the algorithm, the

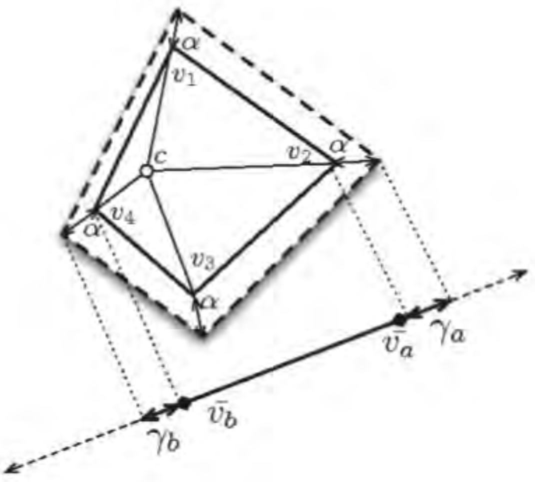


Figure 2.12: Projection of an expanded convex polytope

test point is projected into the space spanned by the tth random projection matrix. Then, each vertex of the tth convex hull computed at the training step is expanded by its corresponding gamma value (Eq. 2.16). Given the set of vertices of the expanded convex polytope in the low-dimensional space (Eq. 2.15), we check if the test point lies inside the projected polytope. A point is said to belong to the model if it lies inside all the T projected polytopes. Many algorithms may be used to check if a data point is inside a 2-D convex polytope like, for instance, the ray casting algorithm [Preparata and Shamos(1985)]. In the 1-D case, the convex polytope is reduced to a single line segment, and checking if a point lies in the segment just requires two comparison.

```
Input: Training set $C \in \mathcal{R}^d$, with $d$ the number of features;
       Number of Projections $T$
Output: The model $\mathcal{M}$ composed of T projection matrices and their respective convex hulls
        vertices. The center $c$ of the data.
1 $\mathcal{M} = \emptyset$
2 $c = \frac{1}{N} \sum x, \forall x \in C$
3 foreach $t = 1..T$ do
4     $P_t \sim \mathcal{N}(0,1)$ % Create a normal random projection matrix
5     $C_t : \{P_t x \mid x \in C\}$ % Project data onto the low dimensional random space
6     $\{v\}_t =$ conv $C_t$ % Find the convex hull and return the set of vertices
7     $\mathcal{M} = \mathcal{M} \cup (P_t, \{v\}_t)$ % Store the set of vertices associated to the convex hull in the projected
      space and the projection matrix
8 end
```

Algorithm 3: Approximate convex polytope decision ensemble training algorithm.

```
  Input: A test point $x \in \mathcal{R}^d$;
         The model $\mathcal{M}$ and center $c$;
         Parameter $\alpha$
  Output: Results
1 Results = INSIDE
2 foreach $t = 1..T$ do
3     $\bar{x_t} = P_t x$ % Project data.
4     $v_t^\alpha : \{v_i + \gamma_i \frac{(v_i - c)}{\|v_i - c\|} \, | v_i \in \{v\}_t\}$ % Find the expanded convex polytope in the low dimensional space
5     if $\bar{x_t} \notin$ conv $v_t^\alpha$ then
6         Results = OUTSIDE
          Break
7     end
8 end
```

Algorithm 4: Approximate convex polytope decision ensemble testing algorithm.

2.2.2 Non-Convex Approximate Polytope Ensemble

Many real world problems are not well modeled using a convex polytope. In this case, an extension of APE is proposed that approximates the non-convex boundary by an ensemble of convex polytopes. The algorithm is based on a tiling strategy where each convex polytope is approximated by the APE methodology formerly described.

The underlying idea of this extension is to divide the non-convex boundary into a set of convex problems. The result of this process is a new ensemble algorithm called **non-convex APE (NAPE)**. Algorithm 5 describes the pseudo-code for creating this decomposition. Starting with a random point c, the set of points inside a d-ball centered at c with radius r are considered at first. Figure 2.13(a) illustrates this step. The black point corresponds to the first center c. Around c, a d-ball of radius r is laid and the convex hull of the set of points inside the ball is computed. This first d-dimensional convex hull is approximated using the technique described in Section 2.2.1. For each projection the set of vertices conforming the convex hull in the reduced space are back projected[3] into the original space – see the gray points in Figure 2.13(a). These points are added to a list of new candidate centers. At the next iteration, the algorithm removes one of the possible candidate centers of the list and repeat the former process as long as there are still training data points not covered by any of the created convex hulls. Figure 2.13(b) shows the second iteration of the process.

[3]Note that this step only needs to keep track of the points that are selected in the low-dimensional space and correspond to the vertices of the projected convex hull.

```
Input: Training set $C \in \mathcal{R}^d$, with $d$ the number of features;
       Number of Projections $T$
Output: Model $\mathcal{M}$ composed of several convex models defined by Algorithm 3
1  $L = \emptyset$
2  Pick a random training point $p_{\text{start}}$
3  $L = L \cup \{p_{\text{start}}\}$ % Initialize the list of possible centers with the first random element
4  Set all data points $x \in C$ to the value not visited
5  while $\exists$ x with value not visited do
6      if $L = \emptyset$ then
7          Pick a random a training point with attribute not visited, $p \in C$
           $L = L \cup \{p\}$
8      end
9      $p = \text{first}(L)$ % Remove the first element of the list
10     $C_i : \{x \in C | \, \|x - p\|_2 \leq r\}$ % Find the set of points to be modeled with a convex polytope in
       this iteration
11     $\mathcal{M}_i = \text{TrainAch}(C_i, T)$ % Find the approximate model associated to the selected set using
       Algorithm 3
12     $\mathcal{M} = \mathcal{M} \cup \mathcal{M}_i$ % Add the new convex model to the final model set.
13     $L = L \cup \{v_i \in C | \, \bar{v_i} \in \mathcal{M}_i\}$ % Add the points of $C$ corresponding to vertices of the projected
       convex hulls of the current model $\mathcal{M}_i$
14 end
```

Algorithm 5: Non-convex approximate decomposition algorithm.

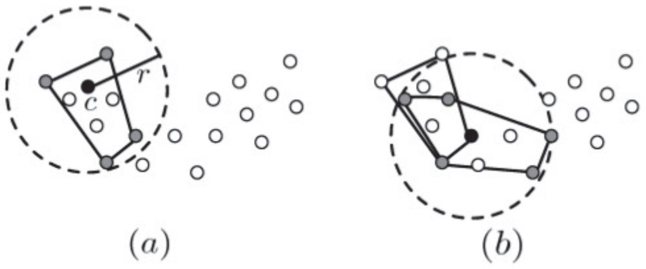


Figure 2.13: Approximation of bi-dimensional Banana-shaped dataset using NAPE

In the same way the extended convex polytope is defined, an extended non-convex polytope may also be defined. Observe that, due to the definition of the extended convex polytope in Eq.2.14 and its approximation in Eq.2.15, taking a global α value for all the convex polytopes suffices to expand the whole non-convex shape by a constant value α. In order to test if a point lies inside the non-convex model, it must be checked if it lies **inside of at least one** of the conforming extended convex polytopes. Otherwise, the point is considered external.

2.2.3 Experimental Results

In this section, an exhaustive comparison of APE and NAPE with many commonly known one-class classifiers is presented. Here, the effect of projecting multi-dimensional data down to one or bi-dimensional spaces is taken into account. For that reason, APE-1 and APE-2 will mark the APE strategy when data are projected respectively down to a line and down to a plane. The validation methodology, has been expanded and presented in the subsequent subsection in order to include artificial data and UCI datasets. In the next subsections, results obtained are reported.

Validation Methodology

APE and NAPE algorithms are compared with state-of-the-art one-class classifiers. APE is created by projecting data down to both 1-dimensional (APE-1) and 2-dimensional (APE-2) spaces. The methods are validated on two different typologies of problems using standard performance evaluation metrics for one-class classifiers.

Comparison Methods: The proposed approaches are compared to Gaussian model (Gauss), Mixture of Gaussians (MoG), Parzen Density Estimation (PDE), k-Centers (kC), k-Nearest Neighbor (k-NN), k-Means (kM) [Bishop(1995)], Minimum Spanning Trees (MST) [Juszczak et al.(2009)] and SVDD.

Evaluation Metrics: The Area Under the ROC Curve (AUC) is used as the performance measure. In the proposed approaches, the ROC curve is computed using the expansion factor α as a varying parameter.

Parameters Setting: For all the parametric one-class classifiers, the optimal values of the parameter have been found using a 2-folds cross validation procedure on the training set varying the parameter in the range 1 to 10. For PDE and MST, no optimization parameters are needed. The fraction of no-target data rejected by the compared classifiers is set to 0.1. For APE-1, APE-2 and NAPE, the number of projections represents a parameter of the ensemble and it has been arbitrarily set to 1000. Proper discussion on the number of projections is found in the following sections. For NAPE, the value of the optimal radius r has been chosen by 2-fold cross validation on the training set.

Datasets APE and NAPE are validated on two different typologies of problems:

- **Artificial Datasets** These datasets have been taken into account in order to evaluate the behavior of the methods with respect to non-convexity. Normal distribution, banana-shaped, S-shaped, toroidal and 3-shaped distributions have been used. No-target points are generated using the procedure described in [Tax and Duin(2001)]. The radius of the generating hypersphere is set to 2.

For each problem, 5-fold cross validation is performed on ten randomly created sets and the results averaged.

- **UCI datasets** A number of 82 real one-class problems derived from UCI machine learning repositories datasets is considered. One-class problems are obtained using one class as target and considering data from other classes as no-target points. The list D of datasets is shown in Table 2.7. Each problem is evaluated using 5-fold cross-validation on 10 different permutations of the data for a total of 50 experiments per problem. The final result is obtained averaging all the results. Due to its high computational complexity in training phase, a reduced set of problems are used for comparison with SVDD. This subset D^* is composed by problem in Table 2.7 marked with star. The choose of these datasets is based on the fact that all these datasets, except Tic-tac-toe, have cardinality less than 500 elements.

Table 2.7: List of One Class Problems

Id	**Dataset**	**Targets**
1,2,3	Balance-scale	3
4,5	Breast-Cancer	2
6,..,11	*Breast Tissue	6
12,13	*Bupa	2
14,..,17	Car	4
18,..,27	Cardiotogography	10
28,29	*CB-Sonar	2
30,31	*CB-Vowel	2
32,..,34	Contraceptive	3
35,..,37	*Glass	3
38,39	*Haberman	2
40,..,42	*Hayes-Roth	3
43,44	*Ionosphere	2
45,..,47	*Iris	3
48,49	*Monks 1	2
50,51	*Monks 2	2
52,53	*Monks 3	2
54,55	*Statlog Heart	2
56,..,62	Statlog Seg	7
63,..,65	*Teaching	3
66,..,68	*Tic-tac-toe	3
69,..79	Vowel	11
80,..,82	*Wine	3

Table 2.8: Comparative Results obtained on Artificial Datasets

	Normal	**Banana**	**S-shaped**	**3-shaped**	**Toroidal**
Gauss	0.963 ± 0.006	0.941 ± 0.010	0.951 ± 0.010	0.954 ± 0.008	0.912 ± 0.010
MoG	0.963 ± 0.006	0.967 ± 0.005	0.973 ± 0.006	$\mathbf{0.971} \pm 0.007$	0.941 ± 0.009
PDE	0.963 ± 0.006	0.967 ± 0.005	0.973 ± 0.006	0.970 ± 0.006	0.941 ± 0.008
kNN	0.949 ± 0.009	0.956 ± 0.008	0.966 ± 0.007	0.965 ± 0.006	0.930 ± 0.009
kM	0.958 ± 0.008	0.961 ± 0.006	0.969 ± 0.006	0.969 ± 0.007	0.934 ± 0.009
kC	0.961 ± 0.007	0.955 ± 0.008	0.961 ± 0.007	0.963 ± 0.008	0.924 ± 0.012
MST	0.942 ± 0.012	0.953 ± 0.009	0.964 ± 0.007	0.964 ± 0.006	0.927 ± 0.010
SVDD	0.959 ± 0.006	0.955 ± 0.010	0.954 ± 0.007	0.952 ± 0.007	0.911 ± 0.013
APE-1	0.921 ± 0.029	0.872 ± 0.055	0.931 ± 0.027	0.881 ± 0.056	0.911 ± 0.023
APE-2	0.960 ± 0.013	0.880 ± 0.064	0.962 ± 0.010	0.955 ± 0.015	0.831 ± 0.050
NAPE	$\mathbf{0.966} \pm 0.008$	$\mathbf{0.975} \pm 0.005$	$\mathbf{0.977} \pm 0.006$	0.960 ± 0.004	$\mathbf{0.957} \pm 0.012$

Results on Artificial Datasets

Experiments on artificial datasets have been performed on datasets with 500, 750 and 1000 data-points[4]. In Table 2.8, all results obtained are averaged and reported. All the numerical results are reported in Appendix 2. APE-2 provides good results on the normal distribution dataset and on the S-shaped datasets. APE-1 performs generally better than APE-2 on the Banana-shaped and Toroidal datasets. NAPE always performs better than APE with significant improvement in the performances when highly non-convex datasets are provided, like Banana and Toroidal. Finally, NAPE always performs better than all the other methods on all the artificial problems taken into account.

Results on UCI Datasets

Results obtained on D^* and D dataset are reported in Table 2.9 and Table 2.10. Each element of the table represents the number of times the methods reported on the rows wins, ties and loses with respect to the method reported on the column. In order to evaluate the statistical significance of the results, a z-test [Demšar(2006)] is applied[5]. NAPE is statistically significant with respect to almost all the methods on D^*. Using a larger number of problems, the improvement provided by NAPE still is quite significant. This fact is shown in Table 2.10. NAPE increases the number of problems where the approximate strategies perform better and it is able to approach more close the statistical difference at 95%, when not reached. Numerical values of the AUC obtained in the experiments are reported in Appendix 2. Finally, it should be noted that, there exists a wide range of problems where APE-2 and NAPE tie. This fact derives directly from the definition of the NAPE methodology that must behave like APE-2 when the problem is convex. APE-1 does not provide significant results compared with the other methods.

[4]Due to its high computational complexity in training, SVDD has been evaluated only on a dataset with 500 data-points.

[5]The z-test ensures that, given N problems, if a method behaves better on at least $N/2 + \sqrt{N}$ problems, then there exists a probability $p \geq 0.95$ that this behavior it is not due to randomness.

Table 2.9: Counts of Wins, Ties and Losses obtained on D^*

	Gauss	**MoG**	**PDE**	**kNN**	**kM**	**kC**	**MST**	**SVDD**	**APE-1**	**APE-2**	**NAPE**
Gauss	0/0/0	21/4/17	22/3/17	19/3/20	23/2/17	23/0/19	19/1/22	19/3/20	25/0/17	17/0/25	17/0/25
MoG	17/4/21	0/0/0	22/3/17	14/3/25	27/2/13	23/0/19	18/1/23	19/3/20	22/0/20	13/0/29	12/0/30
PDE	17/3/22	17/3/22	0/0/0	11/3/28	27/2/13	25/0/17	13/1/28	14/3/25	22/0/20	13/0/29	12/0/30
kNN	20/3/19	25/3/14	28/3/11	0/0/0	29/2/11	29/0/13	12/1/29	16/4/22	23/0/19	16/0/26	14/0/28
kM	17/2/23	13/2/27	13/2/27	11/2/29	0/0/0	24/0/18	9/0/33	14/2/26	24/0/18	11/0/31	11/0/31
kC	19/0/23	19/0/23	17/0/25	13/0/29	18/0/24	0/0/0	15/0/27	15/0/27	30/0/12	16/0/26	13/0/29
MST	22/1/19	23/1/18	28/1/13	29/1/12	33/0/9	27/0/15	0/0/0	17/1/24	22/0/20	14/0/28	13/0/29
SVDD	20/3/19	20/3/19	25/3/14	22/4/16	26/2/14	27/0/15	24/1/17	0/0/0	25/0/17	16/0/26	15/0/27
APE-1	17/0/25	20/0/22	20/0/22	19/0/23	18/0/24	12/0/30	20/0/22	17/0/25	0/0/0	12/2/28	10/1/31
APE-2	25/0/17	29/0/13	29/0/13	26/0/16	31/0/11	26/0/16	28/0/14	26/0/16	28/2/12	0/0/0	0/30/12
NAPE	25/0/17	30/0/12	30/0/12	28/0/14	31/0/11	29/0/13	29/0/13	27/0/15	31/1/10	12/30/0	0/0/0

Table 2.10: Counts of Wins, Ties and Losses obtained on D

	Gauss	**MoG**	**PDE**	**kNN**	**kM**	**kC**	**MST**	**APE-1**	**APE-2**	**NAPE**
Gauss	0/0/0	25/21/36	28/18/36	26/18/38	49/13/20	50/4/28	28/9/45	50/0/32	31/0/51	30/0/52
MoG	36/21/25	0/0/0	34/18/30	27/18/37	54/13/15	49/4/29	34/9/39	51/0/31	39/0/43	36/0/46
PDE	36/18/28	30/18/34	0/0/0	20/22/40	54/13/15	50/4/28	25/13/44	50/0/32	38/0/44	35/0/47
kNN	38/18/26	37/18/27	40/22/20	0/0/0	54/13/15	54/4/24	23/16/43	51/0/31	41/0/41	37/0/45
kM	20/13/49	15/13/54	15/13/54	15/13/54	0/0/0	43/3/36	16/4/62	39/0/43	24/0/58	20/0/62
kC	28/4/50	29/4/49	28/4/50	24/4/54	36/3/43	0/0/0	27/4/51	48/0/34	32/0/50	27/0/55
MST	45/9/28	39/9/34	44/13/25	43/16/23	62/4/16	51/4/27	0/0/0	47/0/35	36/0/46	32/0/50
APE-1	32/0/50	31/0/51	32/0/50	31/0/51	43/0/39	34/0/48	35/0/47	0/0/0	13/20/49	10/5/67
APE-2	51/0/31	43/0/39	44/0/38	41/0/41	58/0/24	50/0/32	46/0/36	49/20/13	0/0/0	0/51/31
NAPE	52/0/30	46/0/36	47/0/35	45/0/37	62/0/20	55/0/27	50/0/32	67/5/10	31/51/0	0/0/0

2.2.4 Discussions

In this section, the number of random projections needed to approximate the original multidimensional convex hull, the role of the expansion parameter α in the classification process and the computational complexity of the methods are discussed.

Number of Projections

The number of projections in APE-1, APE-2 and NAPE has been arbitrarily set to 1000. Experiments show that using 1000 projections, the AUC obtained generally converges to a maximum level of performances. However, using a lower number of projections, high level of performance can still be achieved. In Table 2.11, the number of projections needed to reach the 90%, 95% and 99% of the performances level, computed for D^*, is reported, averaged over all the problems. In order to reach the 90%, APE-2 and NAPE need much less projection than APE-1. Both of them need a very exiguous number of projections to achieve good performances. Nevertheless, the 99% of the performance is reached by using approximately the same number of projections. Results obtained also show that the slope of the performance curve for

APE-1 is smoother than APE-2 and NAPE.

Table 2.11: Mean Number of Projections

	90%	**95%**	**99%**
APE-1	24 ± 57	48 ± 93	156 ± 264
APE-2	8 ± 17	64 ± 132	184 ± 270
NAPE	6 ± 12	25 ± 70	169 ± 261

Expansion parameter

When non-target data are close to the boundary of the convex hull, the number of projections needed for checking if those points are inside the polytope might be very high. However, there is a synergistic effect that allows to mitigate this drawback. The number of projections needed for checking if a point lies inside of the polytope depends on the relative distance d to be checked and the size of the polytope. If we consider a ball of radius R inscribing the polytope, the number of projections is proportional to $\propto e^{-d/R}$. It should be observed that a negative value of the expansion factor α shrinks the polytope. This has two effects: it increases the distance of the point to the polytope by α and reduces the size of the polytope. As a result, the relative distance becomes $\frac{d+\alpha}{R-\alpha}$. Thus, reducing the value of the α parameter reduces drastically the number of projections needed without hindering the performance, i.e. for $\alpha = 0.1R$ the number of projections is approximately reduced by 70%.

Computational Complexity

The proposed approximation strategy demonstrates great advantages from both computational and memory storage points of view when compared to computing the multi-dimensional convex hull. Given a training set of N examples, the computational cost of building the convex hull in APE-2 is $\mathcal{O}(N \log N)$ [Preparata and Shamos(1985)] and $\mathcal{O}(N)$ in the case of APE-1. Let K be the number of points defining the convex hull. The memory needed for storing the convex hull is $K << N$. The cost of testing if a point lies inside or outside the structure is $\mathcal{O}(K)$. Thus, using t projections, the final computational cost for building the APE-2 is $\mathcal{O}(tN \log N)$ and the test cost is $\mathcal{O}(tK)$. In one dimension, the convex hull reduces to a segment. In this case, the modeling of the boundary simply consists in searching for the outermost points in the interval of the real line where the projected points lie.

An estimation of training and testing time for all the methods is reported in Figure 2.14[6]. Training time is represented by white bars and testing time with black ones. Bars are obtained as the logarithm of the training and testing time normalized

[6]Tests have been performed in Matlab R2009a on 4-core Intel i5-2300@2.80GHz desktop computer with 8 GiB RAM

with respect to APE-2. Hence, positive bars are representative of slower times, negative bars represent faster times. Measures are reported with respect to APE-2. For APE and NAPE, training and testing have been performed on 200 projections. APE-1 is the fastest algorithm in both training and testing, followed by APE-2. SVDD is the slowest algorithm in training, followed by NAPE although NAPE is 15 times faster than SVDD. NAPE is the slowest method in testing, followed by MST by a slight difference. NAPE is built using many APE-2 classifiers and the dimension of its ensemble depends by the radius parameter. If the problem is highly non convex, many polytopes are needed to approximate properly the original geometrical structure. Numerical results are reported in Appendix 2.

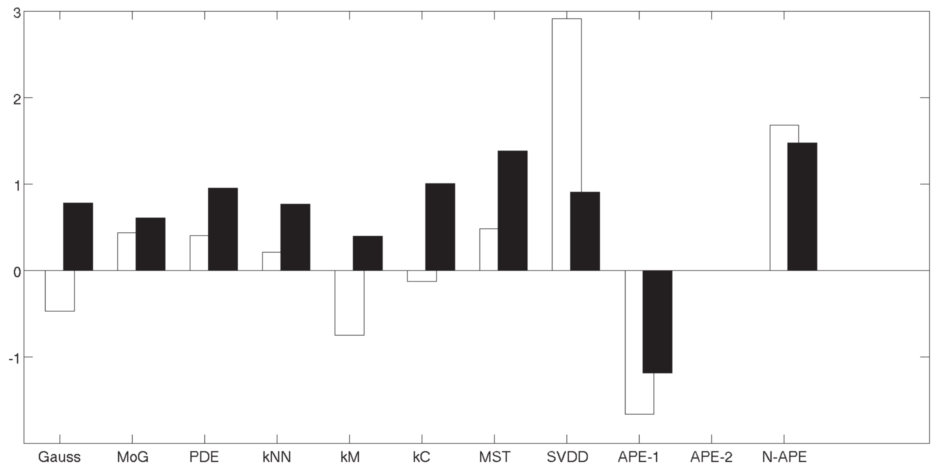


Figure 2.14: Comparison of Training and Testing Time

Table 2.12: Ranking of Best Methods

Position	D^*	D
1	NAPE (4.155)	NAPE (4.109)
2	APE-2 (4.715)	kNN (4.786)
3	SVDD (5.536)	APE-2 (4.908)
4	MST (5.655)	MoG (5.036)
5	kNN (5.702)	MST (5.067)
6	Gauss (5.881)	PDE (5.298)
7	MoG (6.309)	Gauss (5.561)
8	PDE (6.702)	kC (6.518)
9	kC (6.976)	APE-1 (6.665)
10	APE-1 (7.037)	kM (7.048)
11	kM (7.333)	-

2.2.5 Conclusions

In this section, the Approximate Polytope Ensemble and its extension, the Non-convex Approximate Polytope Ensemble have been presented.

APE is based on the Convex Hull and extends this geometric concept in order to model one-class classification problems. Expansion and contraction of the original polytope governed by a parameter α, allows to avoid over-fitting and looking for the optimal operating points in the ROC curve. The high computational complexity for building the convex hull in high dimensional spaces is handled by projecting data-points down to one or bi-dimensional spaces. In those low-dimensional spaces, building the convex hull and check if a point lies inside the polygon are well known problems with very efficient solutions. NAPE extends this approach using a tiling strategy of convex patches able to approximate the original non-convex structure.

APE and NAPE have been compared on two different typologies of problems with widely used one-class classifiers. A ranking of the results obtained on two sets of one-class problems has been done with results shown in Table 2.12. The ranking shows that NAPE and APE are highly competitive methods.

The fact that NAPE and APE are able to achieve similar results highlights the fact that many of the problem considered may have a convex structure. When strong non-convexities are present into the distribution, the differences in performance between NAPE and APE-2 is significant, as highlighted by results obtained on artificial datasets. On artificial datasets, MoG is competitive with NAPE, over-performing in some cases its performances.

In Chapter 4, the intuition that a mixture of convex structures can provide high performances will be used to model non convex shapes maintaining at the same time a good level of computational speed. The enhancement provided by the APE methodology regarding the computational performance has been shown from theoretical and practical point of view in comparison to the other classification methods taken into account. Nevertheless, while APE is one of the fastest methods in train and the fastest method in test, NAPE pays off its high performances at the expense of computational speed. NAPE manages the non-convexity by mean of a parameter that sets the dimension of the convex patches tiling the non-convex distribution. The optimal radius, able to provide the best approximation, provides also the largest ensemble that accurately models the non-convex shape. This drawback cannot be considered an advantage when the methodology is applied into devices with limited resources. As previously stated, in those cases the mixture of convex structure may represent a solution able to provide the best trade-off between performances and speed.

Chapter 3

Wearable Sensors and Features Selection for Physical Activity Recognition

3.1 BeaStreamer, a new platform for wearable computing applications

The original aim for a Wearable Computers (WCs) was to augment the human perception and increase human mental capabilities. At the early age of WC, fundamental characteristics for a WC were *persistence* and *consistency* [Starner(1999)]. The *persistent* WC was constantly available and used concurrently while the user is performing other tasks. At the same time, the WC was *consistent* because the same structured wearable interface and functionality would be used in every situation, though adapted to the current situation. This original trend, strength of the early prototypes of WC, has waned with time, making systems much more humble and adapted to specific applications. Modern WC systems have today the features already introduced for PAN Devices, able to perceive the status of their user.

In this section, a new platform for wearable computing application, developed during the work on this thesis, will be introduced and described in details. The system will be used principally for collecting data of physical activities in many environmental conditions and with many users. Moreover, the resulting system offers potentialities useful in many contexts related to wearable computing.

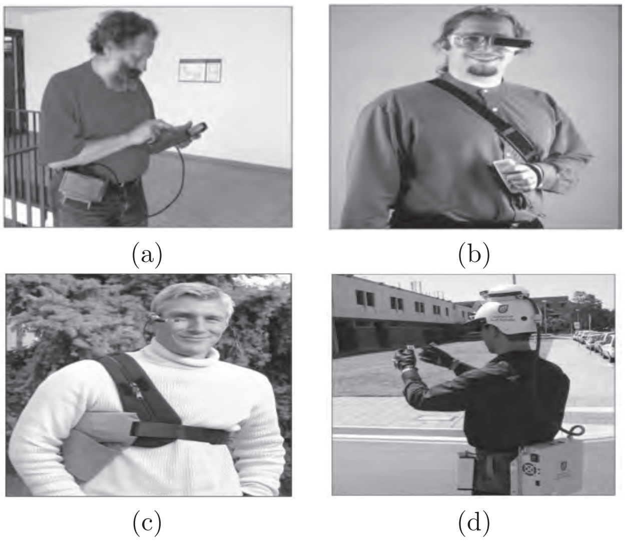


Figure 3.1: (a) The Xybernaut(1990); (b) The Twiddler, Massachusetts Institute of Technology (1993); (c) The WearARM, ETH Zürich (2001) ; (d) TimMith, University of South Australia (2003) (preprint from [Amft and Lukowicz(2009)])

3.1.1 Background on Wearable Computing

In the summer of 1961, Claude Shannon and Edward O. Thorp met in the Casino in Las Vegas for testing a special device. Claude Shannon wore a concealed computer and, using toe-operated switches, timed the ball and the rotor of the ball running in the roulette wheel. The computer would send the prediction about the outcome by radio to Ed Thorp, the bettor. The predictions were consistent with the laboratory expected gain of +44% but a minor hardware problem deferred sustaining serious betting. This curious history, published by Thorp himself at International Symposium of Wearable Computing [Thorp(1998)], told us how the first wearable computer was built. In the 80s, technological challenges related to size, power consumption, and weight constrained the development of wristwatch computers to simple calculators. In the early 90s, the increasing processing performance allowed to build WC able to perform classic desktop computing tasks. In that decade, the Xybernaut [Xybernaut(1990)] (Figure 3.1(a)) , one of the first commercial solutions, appears on the market. The standard Xybernaut system setup consisted of a belt-attached computing block and a carry-on display. Over all the decade, many universities and research centers started to build its own wearable system. The pioneer Thad Starner, in 1993, built the Twiddler [Starner(1993)](Figure 3.1(b)), with an head mounted display and a one-handed keyboard. At the early 2000s, researchers at ETH in Zürich investigated approaches to clothing attached electronics resulting in the WearARM computing core [Lukowicz et al.(2001)] (Figure 3.1(c)). As computational power increases, augmented reality has been integrated in WC. In 2003, Bruce Thomas and his colleagues at University of South Australia developed TinMith [Piekarski and Thomas(2009)](Figure 3.1(d)) to study information overlay outdoors, using a WC. Since the early 2000s, various mobile computers that could be used as WC have appeared on the market, such as the Sharp Zaurus [Sharp(2001)] and the Nokia N770 [Nokia(2005)]. Nevertheless real user-friendly devices started to appear only around the late 2000s. In 2007, Eurotech introduced Zypad WL1100 [Eurotech(2008)] (Figure 3.2(a)), a wrist-worn touchscreen computer, that includes GPS, motion and audio sensor. In 2008, a joint effort from Intel, the University of Washington and Standford, brings to the Mobile Sensing Platform [Choudhury et al.(2008)] (Figure 3.2(b)), system principally developed for physical activity recognition research.

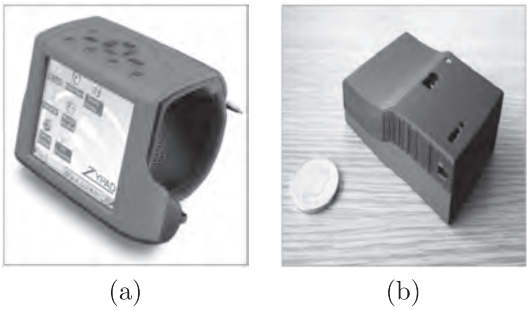


Figure 3.2: (a) ZyPad WL1110, Eurotech (2007); (b) The Mobile sensing Platform, Intel (2008)(preprint from [Amft and Lukowicz(2009)])

3.1.2 Overview on BeaStreamer

BeaStreamer is a custom wearable computing system designed for real-time multi-sensors data acquisition. The principal advantage offered by BeaStreamer is that, once purchased the hardware baseboard, all the Open-Source software components can be installed, assembled and used under GPL [GNU(1989)] or OSI [OSI(1998)] license. Moreover, the hardware baseboard respects the Open-Source hardware philosophy, recently defined in [OSHW(2010)], where the hardware design is made publicly available so that anyone can study, modify, distribute, make, and sell the design or hardware based on that design.

Any type of data-stream can be acquired from the system via hardwired or Bluetooth connection and stored in memory. In Figure 3.3(a), the system disassembled is shown with its components. In Figure 3.3(b), the system is shown packaged. The principal components of the system are the BeagleBoard [Beagleboard(2008)], a low-price board with high computational capabilities, the Bluetooth Inertial Measurement Unit (IMU) produced by Shimmer [Shimmer(2008)], a standard audio-video acquisition device, a Vuzix Wrap 310XL video eye-wear [Vuzix(2006)] and an USB battery. The system can be easily brought in one hand or in a little bag. Audio and video stream are acquired with standard low-cost webcam that can be hooked to the shirt just down the neck or at chest level. The Bluetooth accelerometer can be put in the

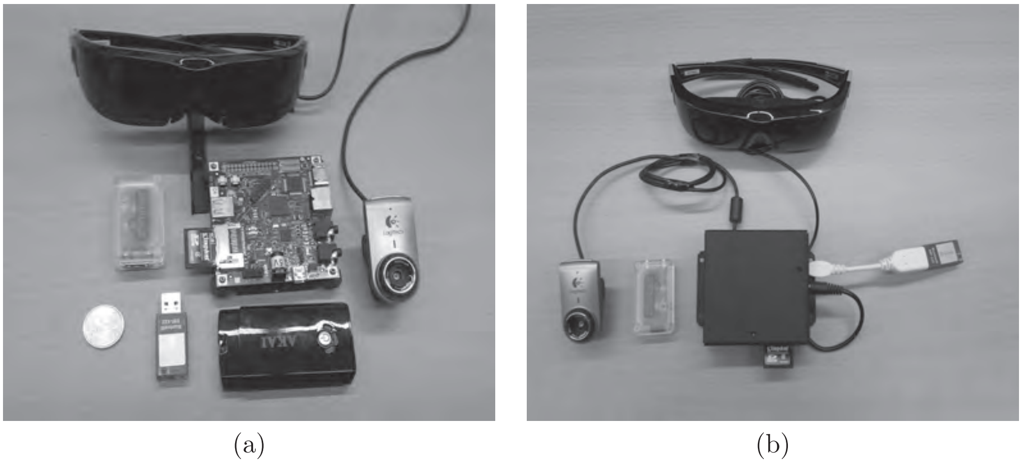


Figure 3.3: (a) Components of BeaStreamer and (b) BeaStreamer packaged

pant pocket or in the shirt pocket. Users can wear the system as in Figure 3.4(a). In Figure 3.4(b), an user proposing a different wearing setting is presented. In the former case, the system is used for acquiring data of physical activities. The accelerometer is inside the bag pocket, positioned on the chest, just behind the camera. The core package is inside the bag. In the latter case, the accelerometer is on the wrist, the core package is hooked at the waist level and the eye-wear glasses are also worn.

The system is able to perform intensive data analysis too. The core of the system is based on the BeagleBoard, an OMAP-based board with high computational capability. The system is equipped on-board with a 4 Gigabytes SD-Card where both operating system and data can be stored. In the next section, the system is described in details at all the levels.

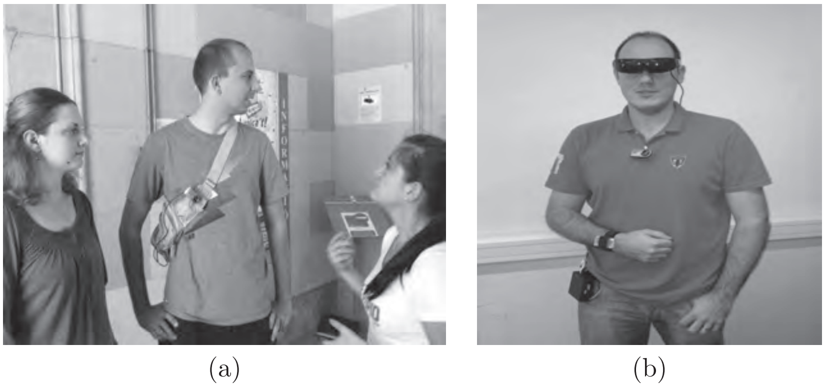


Figure 3.4: (a) User wearing BeaStreamer during data acquisition and (b) User wearing the full BeaStreamer system

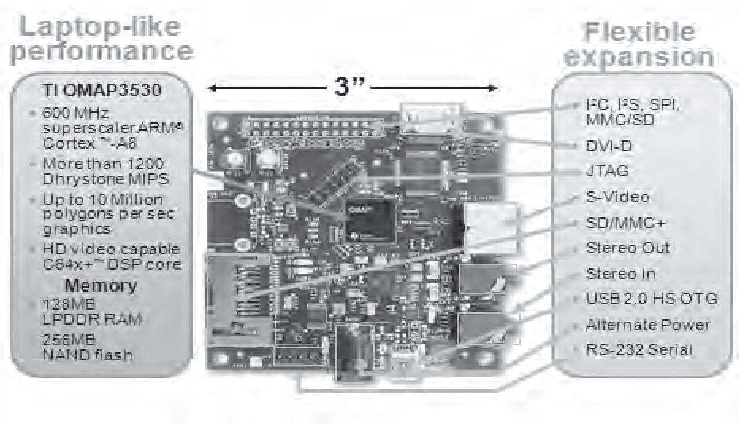


Figure 3.5: BeagleBoard front view with components highlights

3.1.3 Description of the system

In this section, BeaStreamer is described at all the architectural levels. The typical computer architecture approach [Tanenbaum and Goodman(1998)] is used during the description. Firstly the description will focus on the hardware level and, afterwards, on the operating system level and the application software level. In the following, we will refer to BeaStreamer as the conjunction of hardware and operating system constituting the wearable device. BeagleBoard will refer to the hardware board, that is, the board without the operating system.

The Core: BeagleBoard

The BeagleBoard (BB), shown in Figure 3.5, is a low-power, low-cost single-board computer produced by Texas Instruments (TI). With open source development in mind, BB has been developed to demonstrate the potential of the OMAP3530 system-on-chip, though not all OMAP functionalities are available on the board. The BB sizes approximately $80mm \times 80\ mm$ and it provides all the functionalities of a basic computer. The OMAP3530 system-on-chip includes an ARM Cortex-A8 CPU at 500 MHz which can run Windows CE or Linux, a TMS320C64x+ DSP for accelerated video and audio codecs, and an Imagination Technologies PowerVR SGX530 GPU to provide accelerated 2D and 3D rendering that supports OpenGL ES 2.0. Built-in storage and memory is provided through a Package on Package chip that includes 256MBytes of NAND flash memory and 256MBytes of RAM. The board carries a single SD/MMC connector, supporting a wide variety of device such as WiFi Cards, SD/MMC Memory Cards and SDIO Cards. One interesting feature of the OMAP3530 is the possibility of booting the processor from SD/MMC card. Video output is provided through separate S-Video and HDMI connections. A 4-pin DIN connector is provided to access the S-Video output of the BeagleBoard. This is a separate output from the OMAP processor and can contain different video output data from what is found on the DVI-D output. The S-Video output can be used to connect eyewear glasses like Vuzix Wrap 310XL. The BB is equipped with a DVI-D connector that uses an HDMI connector. It does not support the full HDMI interface and it is used to provide the DVI-D interface only. Two USB ports are present on the board. Both ports can be used as host ports, using High Speed USB devices conform to USB 2.0 protocol, using a maximum of 500 mA to power the host device. If additional power is needed or multiple devices as mouse, keyboard and USB mass storage devices have to be used, one USB port can be used as On-The-Go port to drive a self-powered USB hub. This port can be also used to power the board from a standard external USB port. If both USB ports need to be used, there exists an additional 5 mm power jack to power the board. DC supply must be a regulated and clean 5 Volts supply. The board uses up to 2 Watts of power. BB presents on board a populated RS-232 serial connection where a serial terminal is present. Using the terminal, it is possible to set the boot parameters and the size of the video buffer. Furthermore, a 14-pins JTAG connection is present on-board to facilitate the software development and debugging using various JTAG emulators. Two stereo 3.5mm jacks for audio input and output are provided. An option for a single 28 pin header is provided on the board to allow the connection of various expansion cards. Due to multiplexing, different signals can be provided on each pin providing more that 24 actual signal accesses. This header is not populated on the BB and, depending on the usage scenario, it can be populated as needed. Because of the efficient power consumption, the board requires no additional cooling. Typical usage scenario for the BB are shown in Figure 3.6. BB might be considered a substitute of a laptop PC.

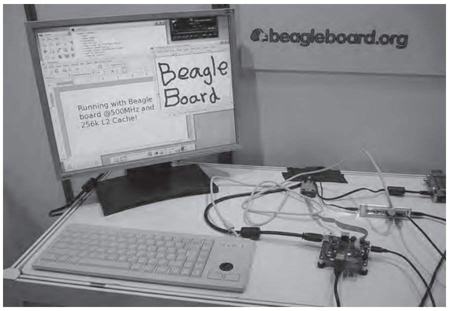


Figure 3.6: BeagleBoard usage scenario: BeagleBoard substitutes a laptop PC

The Motion Sensor: from Arduino to Shimmer

At the early stage of the development of BeaStreamer, no wireless motion sensor and IMU devices were commercially available at accessible cost. For that reason, a custom Bluetooth accelerometer was assembled to perform experiments. The custom Bluetooth accelerometer, shown in Figure 3.7(a), uses an Arduino board, a ADXL 345 accelerometer and a BlueSMiRF Gold Bluetooth modem. Arduino [Arduino(2008)] is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and other actuators. The micro-controller on the board is programmed using the Arduino programming language and the Arduino development environment. Although the custom accelerometer board has a small form factor, its usability is affected by its low physical robustness. Large scale experiments have been performed using a Shimmer Bluetooth-IMU devices, shown in Figure 3.7(b).

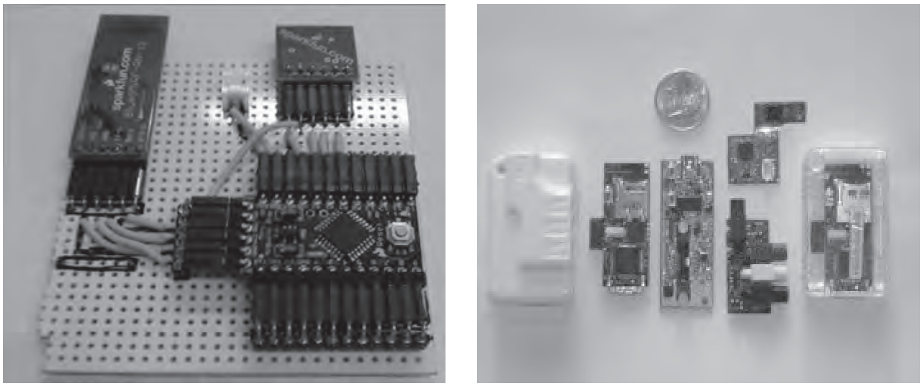

(a) (b)

Figure 3.7: (a) Arduino-based Bluetooth Accelerometer and (b) Shimmer Bluetooth Accelerometer

Power Management

There exists many projects using BB in many different applications , and lastly, a few projects start using BB for wearable computing applications [Beagleboard(2011)]. The major issue in using BB in wearable applications is the need of a suitable portable power supply source. In our typical setting, an AKAI external USB battery at 3400mAh has been used, allowing 4 hours of autonomy for the system in complete functionality. Nevertheless, it is possible to assemble custom batteries able to power the board (and the full BeaStreamer system) for the time needed. A custom four-cells Li-Ion battery has been built up for powering the system. Experiments about its duration have been performed receiving accelerometer data via Bluetooth. Wireless connections, and in particular Bluetooth, are the responsible of the biggest power consumption in electronic devices. Using the custom battery, up to ten hours of continuous Bluetooth data reception can be guaranteed on BeaStreamer.

Boosting computational performances using the DSP

The DSP embedded into the OMAP SoC can be used to enhance the performances of many algorithms, specially algorithms related to machine learning and computer vision. Although there is a JTAG port on the board, the cheapest way to enable the DSP is to make a software bridge from the ARM processor able to view the DSP as a shared-memory peripheral from the ARM-side. TI provides C6EZRun [TexasInstruments(2010)], a free, open-source development tool intended to easy develop DSP code on two-core heterogeneous SoC processor. C6EZRun is a tool which takes in DSP C files and generates either an ARM executable, or an ARM library which will leverage the DSP to execute the C code. Using C6EZRun, the performances of computing array and matricial operations have been evaluated. Results are shown in Figure 3.8 for bidimensional arrays. Depending on the dimension of the array, there may be no evident advantages in using the DSP. For very small array dimensions, using the DSP is much slower than using the ARM processor, due to the cost of inter-processor communication. Nevertheless, when the dimension of the array increases, the advantage of using the DSP are indisputable being able to perform operation of matrix up to three time faster than the normal ARM processing unit.

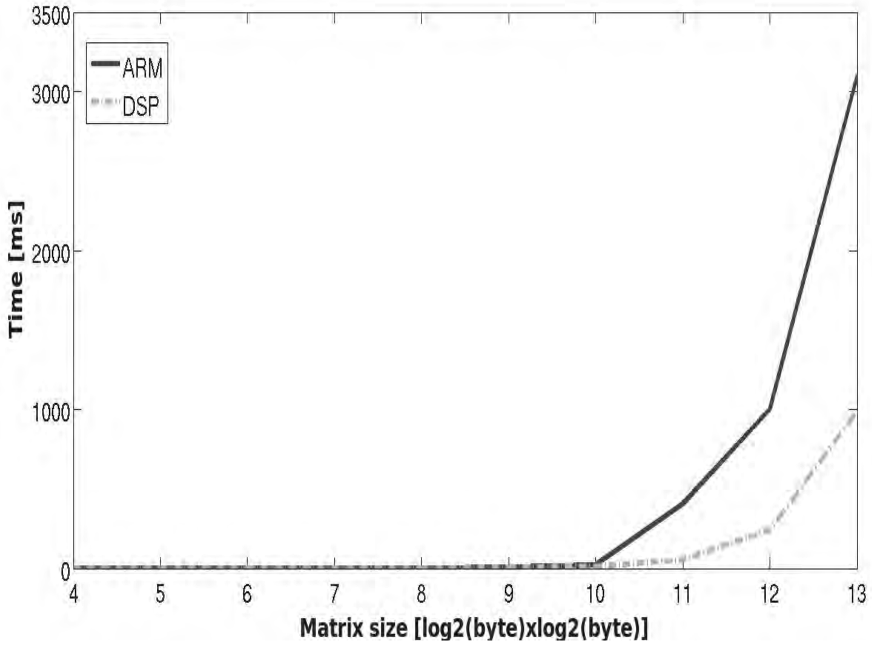


Figure 3.8: Comparison ARM/DSP speed on bi-dimensional arrays

3.1.4 The Operating System Level: OpenEmbedded + Ångstrom

Open Embedded (OE) [OpenEmbedded(2008)] is a complete cross-compiler suite allowing developers to create complete Linux Distributions for embedded devices. In particular, OE offers different kernels for the BeagleBoard. A stable Linux Kernel

2.6.28r12 runs on BeaStreamer. This kernel has Video for Linux drivers, allowing to plug in the system almost every Linux-compatible webcam. It also contains BlueZ, the official Bluetooth protocol stack. The Ångstrom Distribution, a specific Linux distribution for embedded systems, is running on the board. The Ångstrom Distribution can be easily built using OE with the possibility to install all the most common packages available for Linux. In the distribution built, a tool-chain for developing source codes on board has been included. Arm-gcc, arm-g++ and Python-Numpy development environment are installed on the board.

3.1.5 The Application Software Level: GStreamer and OpenCV

The main functionalities of BeaStreamer are acquiring and processing biometric signals. For that reason, a framework able to support the data acquisition process would be useful. GStreamer, although born as a framework oriented to streaming media applications, provide very useful functionality for data-stream acquisition process.

The GStreamer framework [GStreamer(1999)] is designed to easily write applications handling audio/video streaming allowing simple interfaces for streams synchronization. GStreamer is not restricted to audio and video. It can manage any kind of data flow. The main advantages of GStreamer are that the software components, called plug-ins, can be mixed and matched into arbitrary pipelines so that it is possible to write complete streaming data editing applications. Plug-ins can be linked and arranged in a pipeline. The main function of GStreamer is to provide a framework for connecting plug-ins, for data flow management and for media type handling and negotiation. Using GStreamer, performing complex media manipulations becomes very easy and it integrates an extensive debugging and tracing mechanism. In BeaStreamer a pipeline acquires audio and video from web-cam, with the possibility to encode the data-flow with the request quality and the resolution and the possibility to change the acquisition parameters at run time, and the motion data from the Bluetooth channel.

The Open Source Computer Vision Library (OpenCV) [OpenCV(2007)] is a library of programming functions for real time computer vision and machine learning applications, developed by Intel. OpenCV is released under a BSD license, it is free for both academic and commercial use. It has C++, C, Python and Java interfaces running on Windows, Linux, Android and MacOS. Many computer vision application are based on the OpenCV library. This library has been compiled for the Ångstrom distribution running on BeaStreamer. The optimization of many of its functionalities using the DSP are still under development from the BeagleBoard Internet community.

3.1.6 Showing the potentiality of BeaStreamer

In this section, experiments performed using BeaStreamer aiming to demonstrate its computational capabilities are shown. In Figure 3.1.6, a sequence of photos taken wearing BeaStreamer, walking in the street are shown. The frame-rate is one

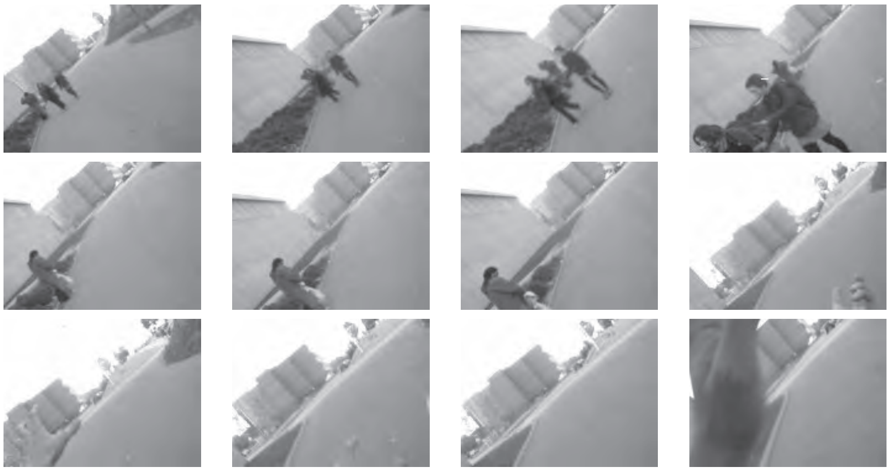
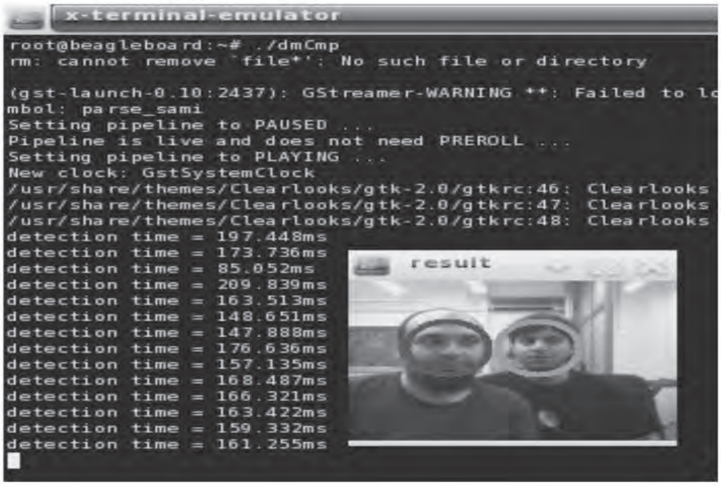


Figure 3.9: Example of Pictures taken wearing BeaStreamer

photo/second with a size of 320x240 pixels, compressed in jpeg format. At the same time, in a separate thread, a continuous audio flow is grabbed from the webcam microphone, sampled at 44100 samples/s and compressed in ogg format. GStreamer allows setting online the parameters of acquisition making simple to change the resolution of photo and the encoding audio quality at run-time, even under an intelligent adaptive setting.

In Figure 3.1.6, the functionality of an OpenCV-based face detector is shown. The face detector can compute detections at a frame-rate of 5-10 frames/second depending of the images resolution, without using DSP. The synergy of GStreamer and OpenCV, allows working with very good performances. Using images with size of 80x60 pixels, the face detector can scan the image in less than 100 ms and detect faces in 200 ms.

A system for face verification has been implemented using BeaStreamer. The system uses the eigen-faces technique [Turk and Pentland(1991)] based on Principal Components Analysis. The system runs with 2 frames/seconds, detects faces using the OpenCV face detector and performs the face verification on the face detected. In controlled environments, the systems perform with very high classification rates. Some examples of classification are shown in Figure 3.11. Big issues arise during the verification process due to the intrinsic problems of the technique used, especially concerned to lighting conditions and posing of the faces grabbed from the wearable camera.



```
root@beagleboard:~# ./dmCmp
rm: cannot remove `file*': No such file or directory

(gst-launch-0.10:2437): GStreamer-WARNING **: Failed to lo
mbol: parse_sami
Setting pipeline to PAUSED ...
Pipeline is live and does not need PREROLL ...
Setting pipeline to PLAYING ...
New clock: GstSystemClock
/usr/share/themes/Clearlooks/gtk-2.0/gtkrc:46: Clearlooks
/usr/share/themes/Clearlooks/gtk-2.0/gtkrc:47: Clearlooks
/usr/share/themes/Clearlooks/gtk-2.0/gtkrc:48: Clearlooks
detection time = 197.448ms
detection time = 173.736ms
detection time = 85.052ms
detection time = 209.839ms
detection time = 163.513ms
detection time = 148.651ms
detection time = 147.888ms
detection time = 176.636ms
detection time = 157.135ms
detection time = 168.487ms
detection time = 166.321ms
detection time = 163.422ms
detection time = 159.332ms
detection time = 161.255ms
```

Figure 3.10: OpenCV Face Detector running in real-time on BeaStreamer

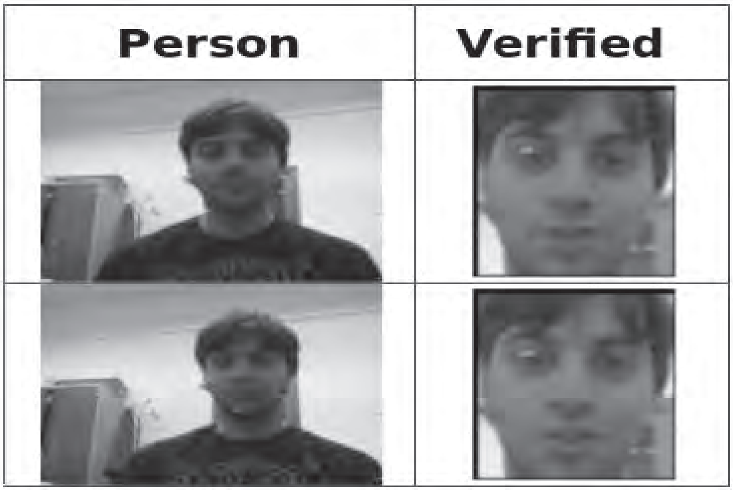


Figure 3.11: Face Verification using BeaStreamer

3.1.7 Discussions

In this section, BeaStreamer, a platform for wearable computing applications, has been presented. Although the system was initially designed for data acquisition tasks for wearable applications, its natural evolution has led to a very flexible system with a good physical strength, even if its development has been only limited to the prototypical phase. Its high computational capabilities allow to use the system not only for data acquisition but as a powerful and complete PAN Device.

BeaStreamer can sense the environment using audio and video and monitor user activity using an accelerometer. Intelligent feedback to the user can be provided using audio and visual information. Although the computational capabilities of the system have been shown at the early stage of development, its potentiality allows the use of BeaStreamer in a wide spread of Ambient Intelligence applications.

Nevertheless, the question of why developing a new wearable device when a great number of powerful computational devices like tablet-pc and smart-phones are nowadays available, naturally arises.

The degree of openness that an ad-hoc architecture can provide represents a great advantage at the time to develop software. Moreover, this development can be done with many different and commonly used tools. Available commercial systems, even those with open Application Program Interfaces(APIs), always enclose their hardware layer in order to avoid, in the right way, the full control of the system.

On the other hand, the main drawback of assemble an ad-hoc hardware architecture is provided by the Moore's law[1]. The small form factor of the BeagleBoard, in the early stage of development representing its strength jointly with the high computational power, has been outmoded by more powerful boards with even smaller form factor (see [Gumstix(2011)], shown in Figure 3.12). For a new development in a new board, the complete rebuilt of the whole software stack needs to be performed. Avoiding this rebuilt is what open API commercial devices provide to the final software developers.

Figure 3.12: The Gumstix Board

[1]In 1965, Gordon Moore, co-founder of Intel, predicted that the number of transistors integrated into a chip would double every two years. This forecast is commonly known as Moore's law.

3.2 Features Extraction and Selection for Physical Activity Recognition

The task of classifying and monitoring a basic set of Activities of Daily Living (ADLs) [Lin et al.(2011)] takes the name of Physical Activity Recognition. ADLs refers to daily self-care activities within an individual's place of residence, in outdoor environments, or both. ADLs are divided into Basic ADLs, consisting of self-care tasks like personal hygiene, dressing, self-feeding and deambulation, and Instrumental ADLs like taking medications, managing money, shopping or using telephone or technology. In this section, a sub-set of ADLs will be selected and successful classified, isolating at the same time the optimal set of features able to provide the best classification performances.

The *deambulation* ADL will be taken into account differentiating three aspects of that general definition. **Walking**, **walking up/down stairs** and **staying standing** will be separately included into our basic set of ADLs.[2]. These three activities may be considered the baseline activities of healthy people. "*Using technology*" is listed as Instrumental ADL. For that reason, **using a personal computer** will be included into our set of ADLs as well. Being involved in a social context represents an important aspect in everyday life and the definition of ADLs also contemplate a *speaking* activity. Hence, **social interaction** activities, i.e. speaking with a single or a group of persons, will be the fifth and last ADL taken into account. From our point of view, this basic set represents the minimum set of ADLs able to prove the healthy physical and psychological status of a subject.

3.2.1 Significant Related Works on Physical Activity Recognition

Traditionally, researchers used vision sensors for activity recognition.

[Clarkson and Pentland(1999)] use high dimensional and densely sampled audio and video streams classifying in an unsupervised way significant events in daily life activities like walking or visiting a supermarket. Since they work in an unsupervised way and general events need to be discovered, authors use coarse features for both audio and video streams.

[Spriggs et al.(2009)] use a wearable camera and Inertial Measurement Units (IMUs) for temporally segmenting human motion into actions and performing activity classification in the context of home environments. The system works with unsupervised temporal segmentation and classifies activities from multimodal data in a supervised way. For supervised experiments, results show that using a simple K-NN model for frame classification outperforms Hidden Markov Models (HMM) or Gaussian Mixture Models(GMM). Authors suggest that this is due to the high dimensionality of the data that cannot be properly handled by GMM and HMM. The gist

[2]The activity of staying standing is representative of situations as waiting for the elevator, waiting for the bus or waiting for in single line at bank, post office etc.

of each image has been used as features for visual data and Principal Components Analysis has been used on IMUs data.

[Lester et al.(2006)] resume their experience in developing an automatic physical activities recognition system. Authors answer some important questions about where to place the sensors, if variation across users helps to improve the accuracy in activity classification and which are the best modalities for recognizing activities. Results show that it does not matter where the users place the sensors but having sensors localized in the same part of the body enhances significantly the classification. Furthermore, variation across users do help improving accuracy classification and the best modalities for physical activities recognition are accelerometers and microphones. They use a machine learning pipeline composed by two classification stages with, at the first stage, a modified version of AdaBoost and, in the second stage, a Hidden Markov Model classifier. They achieve 87% of accuracy in classifying 8 activities. Features used are cepstral coefficients, log FFT frequency bands, spectral entropy, energy, mean, variance, linear FFT frequency bands, correlation coefficients and integration over the time series over a window.

As highlighted in the previous works, inertial sensor provide a low-cost, effective and privacy-aware alternative for activity recognition. Accelerometers are inertial sensors widely used in physical activity recognition. Accelerometers consist of a mass suspended by a spring and placed in a housing. The mass inside the accelerometer moves depending on the acceleration of the sensor and displacement of the mass is measured as the difference of accelerations. In many recent works, activity recognition is based on classifying sensory data using one or more accelerometers.

The seminal work of [Bao and Intille(2004)] represents the first work on activity recognition from user-annotated acceleration data. Acceleration data was collected from 20 subjects without researcher supervision or observation. Subjects were asked to perform a sequence of everyday tasks but not told specifically where or how to do them. Mean, energy, frequency-domain entropy and correlation of acceleration data were calculated and several classifiers using these features were tested. Decision tree classifiers showed the best performance recognizing everyday activities with an overall accuracy rate of 84%.

Using the dataset collected in the previous work, [Mannini and Sabatini(2010)] give a complete review about the state of the art of activity classification using data from one or more accelerometers, giving a complete list of features and classifiers used in the field. The best classification approaches are based on wavelet features using threshold classifiers. Furthermore, they classify 7 basic activities and transitions between activities, from 5 biaxial accelerometer placed in different parts of the body, using a 17th-dimensional feature vector and a Hidden Markov Model classifier, achieving 98.4% of accuracy. In their work, they separate high-frequency motion components from low-frequency components of the acceleration signal that are related to the dynamics of the activities the subject is performing. Features are extracted only from low frequencies components.

3.2.2 Features Extraction

High dimensionality does not necessarily mean richer information and, often, interesting patterns and underlying structures in the data can arise using a reduced number of dimensions. Learning algorithms suffer from the so-called “curse of dimensionality” which denotes the drastic raise of computational complexity and classification error with data having large dimensionality. In this case, data can be transformed into a reduced representation of features. This process is called *feature extraction*. Features extraction from raw data provides dimensionality reduction and robustness to noise and, when the features to compute are carefully chosen, the performances of the classification process can be significantly enhanced. In this section, the features extracted on images, audio and motion data are discussed.

Table 3.1: Features Extracted on Images

Horizontal Rectangular Haar Features(H-haar)
Vertical Rectangular Haar Features(V-haar)
4 Rectangular Haar Features(4-haar)
First 10 Singular Values(n-SV)

Features Extraction on Images

Wearing a wearable camera brings to a new paradigm in Computer Vision sometimes known as Egocentric Vision. This new paradigm addresses classical computer vision problems under a new perspective where objects do not appear well positioned in isolated photos but they are embedded into a real and dynamic environment and constantly interact with it. In this case, physical activities are not recognized and classified from a far and passive point of view but from an active and participant one. With the aim to classify ADLs, simple coarse features, computed on the difference between a frame and its subsequent, will be computed. Those features, listed in Table 3.1, roughly quantifies the degree of motion that is present between consecutive differential frames. The first three features are typical Haar-like features, commonly used in computer vision applications[3]. In addition, the first ten singular values obtained by the SVD decomposition[3] of the differential image will be used too. The use of singular values allows to represent the image with a smaller set of values able to preserve useful features of the original image.

Features Extraction on Audio

The process of separating conversational speech from silence is called Voice Activity Detection (VAD). The primary function of a VAD is to provide an indication of speech

[3]Cfr. Appendix 1

Table 3.2: Features Extracted on Audio

Mean Value (Mv)
Standard Deviation (Std)
Skewness(Sk)
Kurtosis (K)
Coefficients Energy 7-Wavelet Decomposition (n-Wdc)

presence and to hopefully provide delimiters for the beginning and end of a speech segment. VAD is a widely used technology for a variety of speech-based applications specially those concerning bandwidth issues in digital communications. VAD systems are based on standard classification schemes. The features shown in Table 3.2 will be used to characterize audio data frames containing a voice activity. Audio data frames have been obtained using windows of raw audio data with 44100 samples, representing one second of speeching, with 50% of overlapping between consecutive windows. Statistical measurements like mean value, standard deviation, skewness and kurtosis[3] will be used to characterize the speech activity. In addition, the sub-band energy of coefficients of 7 levels wavelet[3] decomposition with Haar-like mother wavelet will be used to detect the degree of activity present in the window.

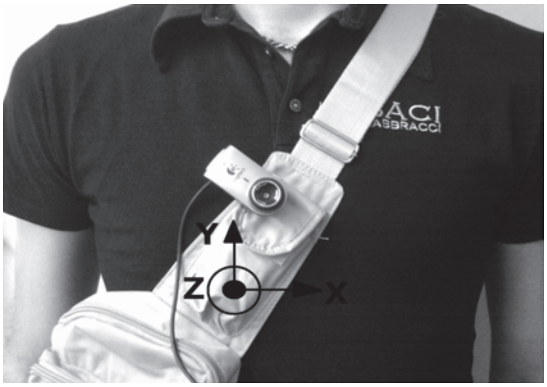


Figure 3.13: User wearing BeaStreamer

Features Extraction on Motion Data

Motion data are strongly related to physical activities. Accelerometers provide three separated values, each one related to a motion axis. The direction of acceleration axis as used in the experimental setting is shown in Figure 3.13. These values provide three time series A_x, A_y and A_z. An example of accelerometer data for different activities is shown in Figure 3.14. In the figure, it is possible to appreciate the regular pattern arising from a walking activity. In climbing stairs, activity similar to walking, the

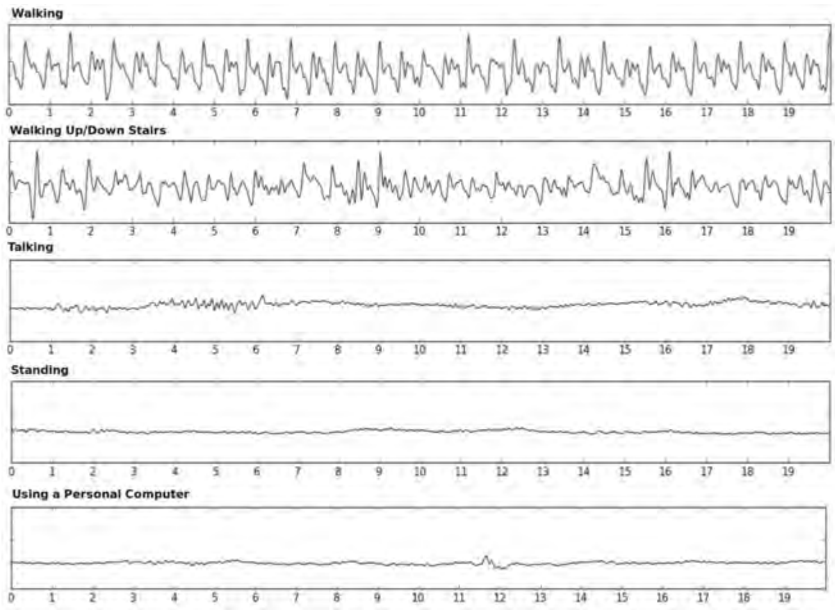


Figure 3.14: Visualization of activity acceleration data

same pattern seems not to be present although some common components between the two activities can be still noted. The rest of activities differs significantly from the previous ones specially in the time series trends and in the intensities of acceleration involved. Small differences in the variation of the acceleration can help to visually differentiate between the three activities.

From these separated time-series data, an additional time series, A_m can be obtained computing the magnitude of the acceleration as in

$$A_m = \sqrt{A_x^2 + A_y^2 + A_z^2} \tag{3.1}$$

Each time series A_i, with $i = \{x, y, z, m\}$ has been filtered with a digital filter in order to separate low frequencies components and high frequencies components. The cut-off frequency has been set to $1Hz$. In this way, for each time series, three more time series A_{ij} can be obtained, with $j = \{b, dc, ac\}$ representing respectively the time series without filtering, the time series resulting from a low pass filtering and the time series resulting from a high pass filtering. Features from each one of these time series have been extracted.

As previously outlined for audio data, a successful technique for extracting features from sequential data has been demonstrated to be windowing with overlapping [Dietterich(2002)]. Features will be extracted from windows of T samples, with $T = 52$, corresponding to 1 second of accelerometer data, with 50% of overlapping between windows. From each window, Root Mean Squared value of the integration of

Table 3.3: Features Extracted on Accelerometer Data

Mean Value (Mv)
Standard Deviation (Std)
Skewness(Sk)
Kurtosis (K)
Correlation between each pairwise of accelerometer axis ($Corr_{i,j}$)
Coefficients Energy 7-Wavelet Decomposition (n-Wdc)
RMS Velocity (Rms)
Mean Value of Absolute Difference of consecutive peaks (Mvad)

acceleration in one window, and the Mean Value of absolute difference of consecutive peaks have been extracted as features. The integration of acceleration corresponds to Velocity. For each window, the integral of the signal and the RMS value of this series will be computed as shown in

$$Rms = \sqrt{\frac{1}{2T} \int_{t}^{t+T} A_{ij}^2(t)} \tag{3.2}$$

Integration is approximated by running sums with step equals to 10 samples. The second typology of features is computed as the sum of all the differences of the ordered pairs of the peaks of the time series, as shown in

$$Mvad = \sum_{k=t\pm\varepsilon}^{t\pm\varepsilon+T} \max A_{ij}(k) - \min A_{ij}(k) \tag{3.3}$$

Finally, in order to complement the proposed set of features, further features already proved to be useful in physical activity recognition [Mannini and Sabatini(2010)] have been added to complement the set. Mean value, standard deviation, skewness and kurtosis[4] have been used to statistically characterize the acceleration in each window. The degree of coupling between acceleration axis has been measured by the correlation between each pair of accelerometer axis, without including magnitude. Finally, the sub-band energy of coefficients of 7 levels wavelet[4] decomposition are also used to measure the degree of activity present in the window. In this case, a Daubechies-4 has been used as mother wavelet. The complete list of features used is shown in Table 3.3.

[4]Cfr. Appendix 1

3.2.3 Features Selection

The dimensionality of the feature vector obtained can be further reduced to a selected subset of features chosen between those having mayor relevance into the classification process. The Random Forest learning function (RF) will be used to measure the relevance of the features extracted.

Here, a short explanation about how RF can be used for measuring features relevance, is given. For more details, please refer to [Breiman(2001)]. RF builds many classification trees. Each tree votes for a class and the forest chooses the classification having the most votes over all the trees. Each tree is built as follows:

- if the number of cases in the training set is N, N cases are sampled at random with replacement. This sample is the training set.

- if there are M input variables, a number $m \ll M$ of variables are selected at random and the best split on these m is used to split the node. The value of m is held constant during the construction of the forest.

- trees are not pruned.

When the training set for the current tree is drawn with replacement, about one-third of the cases are left out of the sample. This Out-Of-Bag (OOB) data is used to get an unbiased estimate of the classification error as trees are added to the forest. This can also be used to get estimates of features relevance. Measuring the importance of attributes in RF is based on the idea that randomly changing an important attribute between the m selected for building a tree affects classification, while changing an unimportant attribute slightly affect the results. Relevance of each feature for a single tree is computed as correctly classified OOB examples minus correctly classified OOB examples when an attributes is randomly shuffled. The final Relevance Measure (RM) is obtained dividing the accumulated attribute by the number of used trees and multiplying the result by 100 in order to get a percentage. The main advantage provided by this approach is that RF performs well on noisy data like those provided by accelerometers.

Selection of Image Features

In Table 3.4, on the right, ten features selected from images, presented in decreasing RM order, are shown. Many of the features selected are from the SV. This fact confirms the intuition that SVs can be representative features of an image. In this particular case where we work on the difference between subsequent images, SVs may be representative of images having low motion. The 4-Haar feature, having an important weight as well, is also representative of this fact.

Table 3.4: Features Selected on Images (right) and Audio (left)

Images		Audio	
Features	**RM**	**Features**	**RM**
1-SV	4.31	6-Wdc	15.05
4-haar	4.15	7-Wdc	10.87
2-SV	2.92	Std	9.94
3-SV	2.76	K	8.1
4-SV	1.68	4-Wdc	8
H-haar	1.43	5-Wdc	7.94
5-SV	1.33	1-Wdc	7.85
8-SV/10-SV	1.16	3-Wdc	5.93
6-SV	1.15	2-Wdc	5.3
7-SV	1.13	Sk	4.5

Selection of Audio Features

In Table 3.4, on the left, ten features selected from audio, presented in decreasing RM order, are shown. All the values of the sub-band energy of the coefficients of the 7-levels wavelet decomposition are selected as significant features. Standard deviation and kurtosis are also selected as significant features. High values in kurtosis mean that a big contribution in the deviation of the distribution is due to infrequent extreme deviations, fact that may be representative of the audio signal during conversation.

Table 3.5: Features Selected on Accelerometer Data

Feature	RM	Feature	RM
MV A_{zdc}	4.64	Mvad A_{zdc}	4.61
Rms A_{zdc}	4.23	Rms A_{mdc}	4.2
Rms A_{xac}	4.14	MVA_{mdc}	4.07
Mvad A_{ydc}	3.92	StdA_{xb}	3.9
Mvad A_{mdc}	3.89	StdJ_{xdc}	3.87
MV A_{ydc}	3.86	Rms A_{ydc}	3.67
MV A_{zb}	3.59	MV A_{xdc}	3.57
Mvad A_{xdc}	3.52	Mvad A_{zb}	3.51
MV A_{yb}	3.33	Rms A_{xdc}	3.22
Rms A_{zb}	3.2	Mvad A_{yb}	2.96

Selection of Motion Features

In Table 1, the best 20 features and their respective RM are reported. Really meaningful features are selected from those extracted on motion data for classifying activities. The most relevant features are related to the Z axis (refer to Figure 3.14(a) to visualize the direction of acceleration axis.) The Z axis is the axis concordant to the direction of the movements. The majority of the features are relative to the low-frequency (DC) components of movements. RMS velocity relative to the X axis has been selected from high-frequencies (AC) components. The information provided by this feature, relative to accelerations on an horizontal axis, helps to discriminate between activities like staying standing, talking and using a PC.

On the other side, features related to the acceleration on the Y axis, can help to discriminate between activities like walking and walking up/down stairs. Mean value, sum of consecutive peaks and RMS velocity are selected for all the DC components of all the time series.

All the features selected extracted from the time series without filtering are also selected from the DC time series. In all these cases, the features selected from the DC time series have an importance value bigger than their corresponding value from the series without filtering. This fact highlights that low-frequencies are more useful than high-frequencies in the classification process of physical activities.

Finally, features derived from higher level statistics (skewness and kurtosis) and features related to the correlation between axis are the features with the lowest importance and are not reported in Table 3.5.

The Physical Activity Dataset

A dataset containing audio, video and motion data related to the selected ADLs has been collected from fourteen volunteers. Testers, three women and eleven men with age between 27 and 35 were asked to perform all the ADLs of our selected basic set. More specifically, users should do :

- **walking up/down stairs** for at least 5 minutes without interruptions ;
- **walking** of a indoor/outdoor environments for at least 15 minutes, without interruptions ;
- **talking** with one or more persons for at least 15 minutes ;
- **staying standing** for at least 5 minutes ;
- **using a personal computer** for at least 15 minutes.

In order to label activities, testers had to annotate the sequential order of the activities performed just restarting the system every time a new activity is started. The system, booting in 2 minutes, automatically starts the acquisition process while the user is already performing the activity. In this way, no cutting has to be performed at the time to analyze data. The user can stop the acquisition in every moment just pressing a stop button. The dataset collected is composed by:

- around 70 minutes of walking up/down stairs (10 sequences);
- around 200 minutes of walking (14 sequences);
- around 120 minutes of talking (11 sequences);
- around 50 minutes of staying standing (9 sequences) ;
- around 190 minutes of using a personal computer (10 sequences).

3.2.4 Classification Results

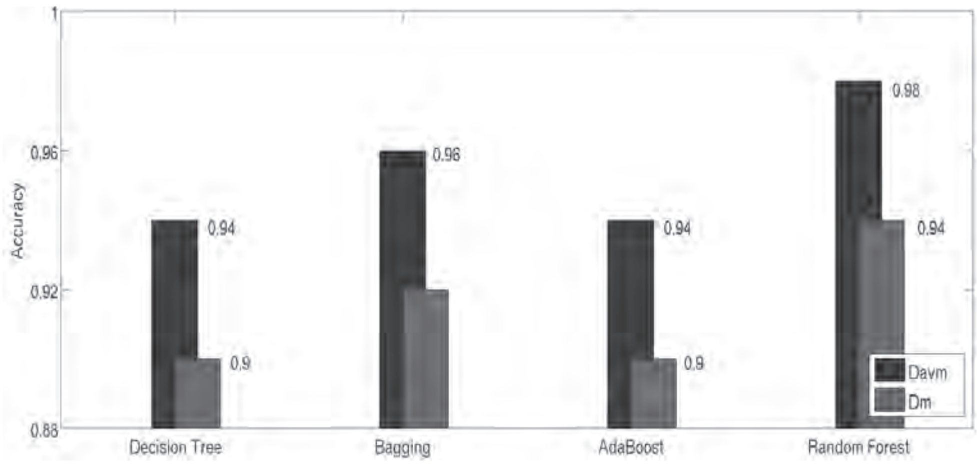


Figure 3.15: Classification Accuracy Using Different Classifiers

Classification performance has been evaluated in terms of accuracy on two different datasets. A first dataset, D_m has been created using the 20 features selected from all those extracted from acceleration data. A second dataset, D_{avm}, has been built joining D_m with the first three features selected from audio and the four best features selected from images. Using two separated datasets allows to evaluate how much the overall information provided by all the sensors may enhance the classification performances with respect to the case where the only motion sensor is used. Classification and Regression Trees [Brieman(1984)], a bagging ensemble [Breiman(1996)] of decision trees, AdaBoost [Freund and Schapire(1999)] and a Random Forest [Breiman(2001)] of 10 decision trees have been used as learning functions.

In Figure 3.15 classification accuracies obtained are shown for both datasets and all the classifiers. Results on D_{avm} are 4% better than results obtained on D_m. On D_{avm}, Random Forest ensures 98% of accuracy. This result is competitive with the state of the art in activities classification.

	Stairs	**Standing**	**Talking**	**Walking**	**usingPC**
Stairs	**0.970**	0.005	0.001	0.022	0.002
Standing	0.001	**0.972**	0.018	0.002	0.007
Talking	0.001	0.009	**0.984**	0.003	0.003
Walking	0.007	0.001	0.004	**0.987**	0.001
UsingPC	0.001	0.002	0.002	0.001	**0.994**

	Stairs	**Standing**	**Talking**	**Walking**	**usingPC**
Stairs	**0.899**	0.006	0.015	0.075	0.004
Standing	0.002	**0.889**	0.093	0.002	0.015
Talking	0.005	0.039	**0.929**	0.006	0.020
Walking	0.029	0.001	0.008	**0.959**	0.002
UsingPC	0.001	0.007	0.013	0.002	**0.977**

Table 3.6: Normalized Confusion Matrix for D_{avm} (up) and D_m (down)

The motivations of those differences can be easily found looking at the confusion matrices, shown in Table 3.6. Confusion matrices obtained with Random Forest on D_{avm} and on D_m are shown in the upper and bottom part of the table, respectively. The rows of the matrices, representing the total number of elements considered for each activity, have been normalized in order to get a percentage. The highest confusion obtained on D_{avm} is between *talking* and *standing*. Although this represents the biggest confusion, its value is not higher than 1.8%. This confusion becomes obviously bigger when only the accelerometer is used. Almost a 13% of confusion is obtained between *talking* and *standing*. For classifying these two activities, the use of the microphone provides an indubitable advantage. Other significant confusions are obtained between *walking* and *walking up/down stairs*. The total amount of confusion reaches in this case 10%. Nevertheless, the overall classification performance obtained on D_m are pretty good, taking into account that the only accelerometric sensor is used. In Figure 3.16, the F-Measure of each activity obtained on D_m are shown. The F-Measure is above 80% for each activity and for *walking*, *talking* and *using a PC* is above 90%.

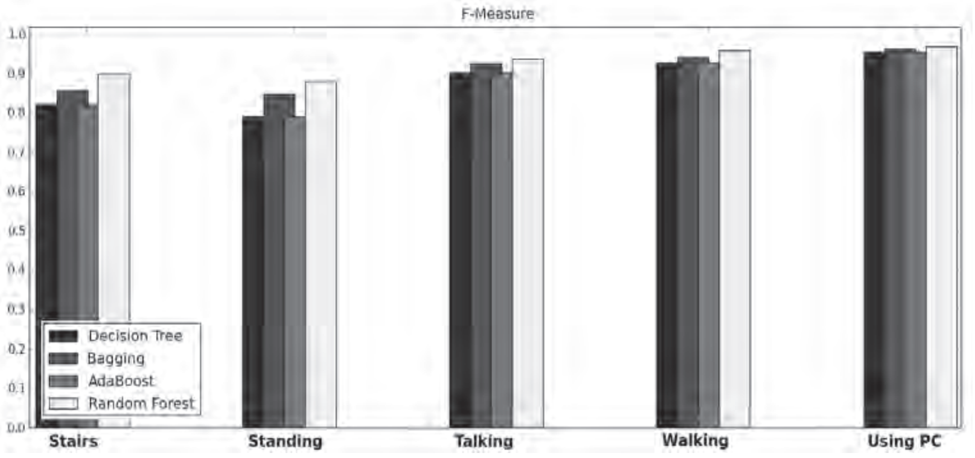


Figure 3.16: F-Measure obtained on D_m

3.2.5 Conclusions

In this chapter, the task of classifying physical activities has been carried out. Using simple and comfortable sensors, five common ADLs have been successful classified with high classification performances. These performances are comparable with the state of the art in physical activity recognition, specially taking into account that a single classifier is used instead of more complex classification architectures. These high performances are principally due to the features used in the classification process, selected from a bigger set using a selective procedure conducted by Random Forest. The selection process has been conducted on all the sensors available and, in particular for the accelerometer, provides a set of features able to provide physical meaning to the quantities involved into the classification process. Using the only set of motion features selected, good classification performances are obtained too.

In Table 3.7, comparison of works using a single accelerometer for physical activity recognition are reported. Results obtained in this chapter using a single accelerometer are completely aligned with results reported in the table, from both features and accuracy point of view. Some works are able to achieve very high classification performance, even higher than results obtained here. Putting aside the mere competition of reaching the highest performance value, the capability of a single accelerometer to provide a good classification represents an important fact. Accelerometers are going to be (or they already are) widely accepted due to their miniaturized form factor, their low-power requirement and for their capacity to provide data directly related to motion.

Last but not the least, accelerometric sensors are much more prone to be accepted by final users because they do not invade privacy. One of the major issues we faced on during the data collection process has been the reluctance of many potential testers to perform the experiment with a camera and a microphone, even once ensured that

Reference	Features	Classifiers	ADLs	Subjects	Accuracy
[Allen et al.(2006)]	Raw Data Delta Coefficients DC Coefficients	GMM	8	6	91.3%
[Song and Wang(2005)]	Wavelet Coefficients	k-NN	5	6	86.6%
[Ravi et al.(2005)]	Standard Deviation Energy Distribution DC Components Correlation Coefficients	Naive Bayes k-NN SVM Binary Decision	8	NA	46.3-99.3%
[Randell and Muller(2000)]	Standard Deviation	ANN	7	NA	95%
[Sekine et al.(2002)]	Wavelet Coefficients	Threshold-based	3	20	98.8%
[Lee et al.(2003)]	FFT	Threshold-based	9	12	95.1%
[Karantonis et al.(2006)]	Magnitude Area/vector Tilt Angle FFT	Binary Decision	10	6	90.8%

Table 3.7: Comparison of Results for Physical Activity Recognition using a Single Accelerometer

the semantics of their conversation would not be used and not even stored. This issue might be solved with the use of only accelerometer data at the expense of a small decrease (but with still good values) in the classification performances.

Finally, the possibility to use only an accelerometer to classifying ADLs is the fundamental concept next chapter states on. With only motion data, a system able to verify users wearing the accelerometer by mean of their walking patterns, will be presented. The system, besides its highly accurate performances, has the main advantage to be used as a unobtrusive and continuous authentication mechanism.

Chapter 4

Personalization and User Verification in Wearable Systems using Biometric Walking Patterns

Mobile devices, smart-phones and tablet-pc are nowadays inseparable objects for us. We bring a mobile phone with us to stay connected via voice, email and chat with our friends, workmates and family. These devices have become in the last couple of years multi-functional wearable computer systems. Their computational capabilities allow their use as GPS localization systems, portable video game systems or personal digital assistants. Moreover, their rapid acceptance in our society, jointly with the advent of new technology like NFC [ISO(2010)], is making those devices oriented towards on-line shopping and consumer applications. While up to now these devices manage information about our personal privacy, in the near future they will manage private and sensible information such as our bank account and credit cards numbers. Protecting the device from illicit and inappropriate use should be constantly ensured making, at the same time, the authentication task as transparent as possible to the user. In this sense, unobtrusive biometric measures become an appropriate tool for verifying devices user identity. Biometrics are currently available on mobile devices. Modern smart phones and PDA have integrated accelerometers able to detect changes in the orientation and acceleration of the device and, consequently, of the person wearing it. Consequently, physical activity recognition represents the starting point to make biometric measurement suitable to verify authorized users.

Persons can be distinguished on the base on their walking style and there seems to be a physiological justification of this fact. In [Bianchi et al.(1998)], authors state that there is inter-individual variability in walking styles between persons particularly at moderate and fast velocity. They show that this variability cannot be simply explained on the basis of the different bio-mechanical characteristics of the subjects, but that it depends on the different kinematic strategies. Subjects differ in their ability to minimize energy oscillations of their body segments and to transfer mechanical energy between the trunk and the limbs. Individual characteristics of the mechanical energy expenditure were correlated with the corresponding kinematic characteristics.

In this chapter, a personalized verification system based on a two stages machine learning pipeline is presented. The system aims to verify authorized users by means of their own walking patterns guaranteeing inappropriate intrusions from unauthorized users. Being truly general, the technique described here may be used in many different applications besides the mere authentication task. Many applications can be found in the fields of Ambient Intelligence and Pervasive Computing where a system, like the one proposed, can provide a continuous authentication mechanism that can possibly be complemented by other mechanisms able to provide very high and reliable authentication rate even in critical situations. Application domains for this kind of techniques may be principally intelligent and personalized settings of domestic and working environments. If a wearable sensor might be able to constantly authenticate yourself, the smart environment always would provide you personalized services and attention according to your needs. Analogously, at work place, the intelligent environment might provide a tracking of the to-do list of the day. Many others applications might be found in multiple application domains of physical activity recognition where a constant and pervasive authentication mechanism finds its natural application.

4.1 Related Works on Biometric-based Verification Systems

To the best of our knowledge, user verification by means of accelerometer data has been rarely addressed. Verification systems are usually measured in terms of the false acceptance rate (FAR) and false rejection rate (FRR). FAR measures the probability of an unauthorized user to be confused with a legit user. On the other hand, FRR measures the probability of the system to misclassify an authorized user as a non-authorized one. While the first measurement concerns the robustness of the system to intruders, the second measurement regards usability and inconspicuousness of the system. In [Vildjiounaite et al.(2007)], a walking-based authentication system has been integrated with fingerprints data and voice recognition, ensuring in this way an high degree of reliability. Walking based authentication provides 14% of Equal Error Rate (EER)[1] with small variations if the system is brought on the chest or on the waist. In that work, data are acquired from an ad-hoc accelerometer-based system assembled for the experiments. Although authors do not use a real mobile phone, that work represents the first work using walking biometrics for user verification. Some efforts have been done also in [Derawi et al.(2010)], where, using the accelerometer of an Android-based mobile phone, walking data have been collected from 51 testers walking for two runs in an interior corridor. Authors report results of 0.2 of EER. Results obtained are encouraging but not persuasive to be implemented in a mobile device for a reliable system. That work is the first one where user verification has been done using an everyday current mobile phone. A close notion to user verification is user

[1]When FRR and FAR are equal, the common value is usually named Equal Error Rate or EER.

identification – not necessarily needed for authentication. In [Gafurov et al.(2006)], a biometric user authentication system based on a person gait is proposed. Applying histogram similarity and statistics about walking cycle, authors ensure 0.16 of identification error rate. Accordingly to the physiological justification of inter-individual variability in walking styles, in [Terrier and Schutz(2003)], authors, using a GPS sensor, are able to capture basic gait parameters over a period of time of 5 seconds. They observed a specific gait pattern for slow walking. The walking patterns in free-living conditions exhibit low intra-individual variability, but there is a substantial variability between subjects.

4.2 User verification system

A machine learning architecture for user verification with personalization is presented in this section. Starting from a general physical activity classifier based on AdaBoost, trained on a baseline training set, the system is personalized adding data of walking activities of authorized users in order to boost the classification performances for those users. Since AdaBoost is an incremental classifier, this process is extremely efficient because just further weak classifiers need to be added to the original baseline classifier. On the base of this personalized system able to filter in its first stage many walking activities of unauthorized users, authorized users are verified too.

The modeling of the walking activity of authorized users can be considered a one-class classification problem where the boundary of a given dataset has to be found without the possibility of knowing any counter examples. This is specific of the case of user verification since examples of all the possible non-authorized users cannot be explicitly provided to the learning algorithm. In the second stage, the Approximate Polytope Ensemble (APE) technique has been used to model the one-class classification problem. The overall verification systems can be seen as a four layered architecture. The inner most layer is based on the APE algorithm. Since the convex hull structure is very sensible to outliers, the next layer concerns a robust way to ignore outliers and define the core distribution of the data. The third layer concerns the modeling of complex shapes using a mixture model. Finally, the last layer takes into account the temporal coherence of the accelerometer data-stream to improve the results.

The overall user verification system, shown in Figure 4.1, can be modeled with a two stage pipeline. The first stage consists of the personalized activity classifier. As a result of the first stage, only data belonging to the class “walking” are provided as input to the user verification stage. This second stage is tuned to verify the walking biometric parameters of the only authorized user.

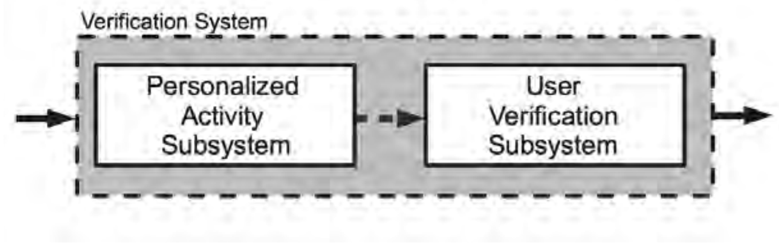


Figure 4.1: Block Diagram of the User Verification System

4.3 Personalized Activity Subsystem

The underlying idea behind the personalization in this subsystem is to bias a general physical activity recognition classifier towards the data of only authorized users. Thus, a general classifier is trained to distinguish among the five ADLs previously introduced. This classifier is trained using data from a general set of people performing these activities.

The general activity classifier is based on a multi-class extension of AdaBoost. It receives as input the features extracted from accelerometer data and it detects the occurrence of walking activities. AdaBoost, which has been briefly presented in Section 3.1, is an efficient incremental algorithm for supervised learning. AdaBoost boosts the classification performance of a weak learner by combining a collection of weak classification functions to form a strong classifier with high performance. The algorithm combines iteratively weak classifiers by taking into account a weight distribution on the training samples such that more weight is given to samples misclassified in previous iterations. The final strong classifier is a weighted combination of weak classifiers followed by a threshold.

Algorithm 6 shows the pseudo-code for AdaBoost. The algorithm takes as input a training set $(x_1, y_1), \dots, (x_m, y_m)$ where x_k is a N-dimensional feature vector, y_k are the class labels and $D_1(k)$ an uniform weights distribution over the training examples. At the training step t, a weak classification hypothesis h_t is selected with error $\epsilon_t \geq 0.5$. The weight α_t correspondent to the current hypothesis h_t is computed proportional to the error ϵ_t. Examples are weighted based on the updated distribution proportional to the current hypothesis. In this way, misclassified examples will have, in the next step, more weight than well-classified examples. After T rounds of training, the weak classifiers h_t and ensemble weights α_t are used to assemble the final strong classifier. The multi-class extension of AdaBoost is performed using an Error Correcting Output Codes One-Vs-All [Dietterich and Bakiri(1991)] technique.

For classifying activities, AdaBoost is previously trained using a large amount of data from many subjects as a general activity recognition classifier $H_{original}(x)$. However, when this general classifier is applied to a specific user, its performance may be poor since inter-variability between people may be significant. For this reason,

```
  Input: A training set $(x_1, y_1), \dots, (x_m, y_m)$, with $x_k \in R^N$ ,
         $y_k \in Y = \{-1, +1\}$
  Output: A classifier $H(\mathbf{x})$
1 Initialize weights $D_1(k) = 1/m, k = 1, \dots, m$;
2 foreach $t = 1, \dots, T$ do
3     Train weak learner using distribution $D_t$;
4     Get weak hypothesis $h_t : X \to \{-1, +1\}$
      with error $\epsilon_t = Pr_{k \sim D_t}[h_t(x_k) \neq y_k]$;
5     Choose $\alpha_t = \frac{1}{2}\ln(\frac{1-\epsilon_t}{\epsilon_t})$;
6     Update $D_{t+1}(k) = \frac{D_t(k)\exp(-\alpha_t y_k h_t(x_k))}{Z_t}$
      where $Z_t$ is a normalization factor chosen so that $D_{t+1}$ will be a
      distribution.;
7     Output the final hypothesis $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$ ;
8 end
```

Algorithm 6: AdaBoost Algorithm

when the system is given to an authorized user, a dataset X_{auth} with data only concerning the authorized user is recorded. The incremental nature of AdaBoost allows to perform T runs of training using a dataset and to follow the training process for T' runs using a new dataset. This feature becomes crucial in the personalization step of the pipeline. In order to bias the performance of the general classifier towards an authorized user, the recorded data from this user X_{auth} is used for T' runs. This adds to the previous strong classifier T' new weak classifiers which take into account the specific biometric features of the authorized user. Hence, the classifier specializes on them. It should be observed that, since our biometric verification system from inertial data is based on the walking activity, only data that the general classifier labels as "walking" are used for this specialization. What we expect as a result of this personalization, is that the performance when classifying authorized users walking activities will be enhanced for the specific user and, consequently, a big load of walking activity from other users will be considerably filtered. Therefore, the first stage of the pipeline serves two purposes. First, it filters walking activity from the rest of ADLs and second, due to the personalization process, it filters many walking activities from non-authorized users.

4.4 User Verification subsystem

Many times, verification tasks are confused with recognition. However, while in recognition it is necessary to chose among different possible choices or classes, in verification only a decision if data belong or not to a given class must be performed. It may be argued that the verification task can be reduced to that of recognizing the desired class versus the undesired one. Although there are some scenarios in which this is true, this reduction does not hold in general. This is because a recognition system needs to correctly model all the different choices in order to achieve good results. However, taking into account our specific problem, the class *non-verified user* can not be effectively modeled since it would require the knowledge of all possible unauthorized users with respect to the authorized one. For this reason, probabilistic approaches that model the distribution of just one class are generally used in these specific problems. When looked at from the machine learning discriminative point of view, the counterpart of the aforementioned problem is One-Class classification. As described in the previous chapter, one of the most successful strategies and state-of-the-art in one-class classification is Support Vector Data Descriptor(SVDD) [Tax(2001)]. However, training a SVDD is computationally expensive and it can not be done efficiently in embedded systems or mobile phones not only due to the computational complexity but also because it usually involves a delicate parameter tuning step.

In the user verification subsystem, the one-class classification paradigm will be followed and the Approximate Polytope Ensemble as baseline one-class classification strategy will be used. Appropriate strategies for outliers rejection and temporal consistence coherence will be adopted and a new ensemble, based on the mixture of APEs will be used to tackle the non-convexity of the problems at hand.

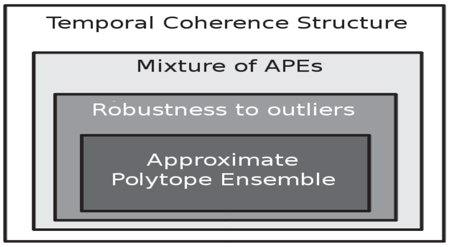


Figure 4.2: Layer structure of the User Verification subsystem

The verification system is structured in layers (see Figure 4.2). The inner most layer concerns the way of modeling the one class problem. The approach proposed is based on the Approximate Polytope Ensemble.

The different layers created around APE deal with the concept of robustness and appropriateness of the convex hull for modeling user data. The inner most layer builds the approximate polytope. The second layer concerns the robustness of the approximated convex hull to outliers by means of a bagging strategy. The third layer models the problem of the convex hull to approximate non-convex shapes. Finally, the last layer considers the temporal coherence of the users walking pattern in order to reduce the verification error rates. In the following subsections, the algorithms used in each layer are described in detail.

4.4.1 Layer 1: Approximate Polytope Ensemble

A One-Class classifier ensemble will be learned using the Approximate Polytope Ensemble strategy. The underlying idea is shown in Figure 4.3 where a scatter plot of a random plane is reported. In this bi-dimensional space, data related to each user are localized in a specific region of the features space. Learning the boundaries of the one-class classification problem means building the convex hull and defining the region of the space where user data lie. When a new point appears, if the point is inside the convex hull, then it represents a walking activity of the user. This approach, as already commented in Section 2.2.4 has important computational and storage advantages. On one hand, given a training set of N examples, the computational cost of building the convex hull in a bi-dimensional space is $\mathcal{O}(N \log N)$. Let K be the number of examples that define the convex hull – the set of examples of the dataset that conform the facets of the polygon – then $K << N$ and the cost of testing if a data point lies inside or outside the convex hull is $\mathcal{O}(K)$. Another worthwhile advantage of computing the convex hull is that it can be built or updated online – as new training data arrives – with cost $\mathcal{O}(\log N)$ [Preparata and Shamos(1985)]. Thus, if t random projections have been used, the final computational cost for building this approach is $\mathcal{O}(tN \log N)$ and the test cost $\mathcal{O}(tK)$.

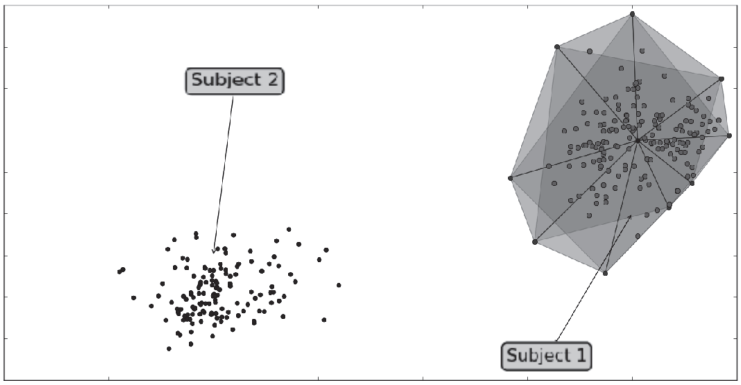


Figure 4.3: User verification using a convex hull as one class classifier

4.4.2 Layer 2: Robust convex hull

An issue that needs to be addressed when using a convex hull is about robustness with respect to outliers. Convex hulls are strongly sensitive to outliers. Outliers can heavily influence the performance of the verification process and, moreover, they are heavily affected by noise. Assuming that data from accelerometers are noisy and can contain outliers, the resulting convex hull might not represent user data accurately. In order to reduce the influence of outliers, when training data are projected down to a bi-dimensional plane, a bagged set of convex hulls will be used. Bagging [Breiman(1996)] is a machine learning ensemble meta-algorithm that improves classification models in terms of stability and classification accuracy. It reduces variance and helps to avoid over-fitting.

Given a training set $X \in \mathcal{R}^M$ of size N, once data are projected, the bagging strategy generates l new training sets X_i' of size N by sampling examples from X uniformly with replacement. Then, l convex hulls will be built using the examples of each sub-sampled training set X_i' and combined by majority voting. In the scatter plot shown in Figure 4.3, the result of building three convex hulls on different sub-sampled sets is shown. This process aims to reduce the influence of the elements on the boundary of each convex hull and better defines the core set of training examples.

4.4.3 Layer 3: Mixture of convex hulls

Data from different walking runs of one user can vary significantly and not necessarily be well represented by a convex shape. NAPE, studied in Section 2.2.2, could represent a possible solution to the problem. Nevertheless, two main drawbacks avoid the use of NAPE in this situation. First, computational complexity of learning and testing NAPE may be high enough to prevent it to be used in limited resources computational devices. On the other side, batches of walking data come on-line. Applying NAPE would mean learning again the whole structure. NAPE has not built to manage this type of problems. In order to overcome this problem, a mixture of APE will be built in this layer (see Algorithm 7). The mixture begins with one APE. As training data arrive, the performance of this convex hull will be checked. If data are well represented by the current model, there is no need to change anything but if a new stream of training data is not correctly represented by the current model, a new APE is learned on the new data and added to the user profile model.

```
Input: A training source that produces streams of data $X_i \in \mathcal{R}^M$
       An APE-2 learning function such that $CH = APE(x)$

Output: A classifier $\mathcal{M}$
1 $\mathcal{M} = \varnothing$;
2 while Not End of Training do
3     if $t_{\mathcal{M}}(X_i) < \tau$      %%$\tau$ : Performance measure then
4         $\mathcal{M} = \mathcal{M} \cup APE(X_i)$
5     end
6 end
```

Algorithm 7: Mixture of APEs algorithm

4.4.4 Layer 4: Temporally coherent convex hulls

The final step is to take into account the temporal coherence of each user data stream. The assumption in this layer is that in a given temporal window, the user of the device does not change, and thus, considering the full length of the window, she must be either an authorized or an unauthorized user. Thus, a simple majority voting on a temporal window of walking data will be used. That is, given a sequential training set $X = \{x_1, x_2, \dots, x_N\}$, a sliding window of size t will be used on the predictions of the layer-3 convex hull $\bar{y}$ of those examples. As a result, given a sequence of predictions, the final decision is achieved by $\hat{y}_i = majority\{\bar{y}_{i-t}, \bar{y}_{i-t+1}, \dots, \bar{y}_i\}$.

4.5 Experimental Results

The system developed has been validated over different setting and, for each sub-component of the system, a proper validation protocol has been defined and used. In the following sections, this validation protocols and experimental settings are results will be explained in detail.

4.5.1 Comparative Results

The APE approach has been compared with the set of one-class classifiers introduced in Section 2.2.3. Average AUC and standard deviations obtained are shown in Figure 4.4. Results are obtained performing a 10-fold cross validation using the LOUO protocol, introduced in the subsequent section. NAPE is the best method able to solve the verification problem with high performances, followed by APE-2. K-Means and K-NN are the methods that, after NAPE and APE-2, provide the best performances. This fact will be justified in Section 4.6.1 where it will be shown how the feature points derived from walking patterns have its own localization in the features space for each specific user. APE-1 does not provide significant good results when compared with the other methods.

4.5.2 Validation protocol

The general activity classifier, the personalized activity classifier and the user verification classifier will be validated using a Leave-One-Out based protocol. For each one of aspect of the system to be validated, the original protocol will need applied with the opportune modifications.

General Physical Activity Recognition The general activity classifier has been trained and validated using a Leave-One-User-Out (LOUO) cross-validation strategy, outlined in Algorithm 8. In LOUO, each subject is used once for testing a classifier trained on the rest of users. This process is repeated for each user and the performance measurements are computed.

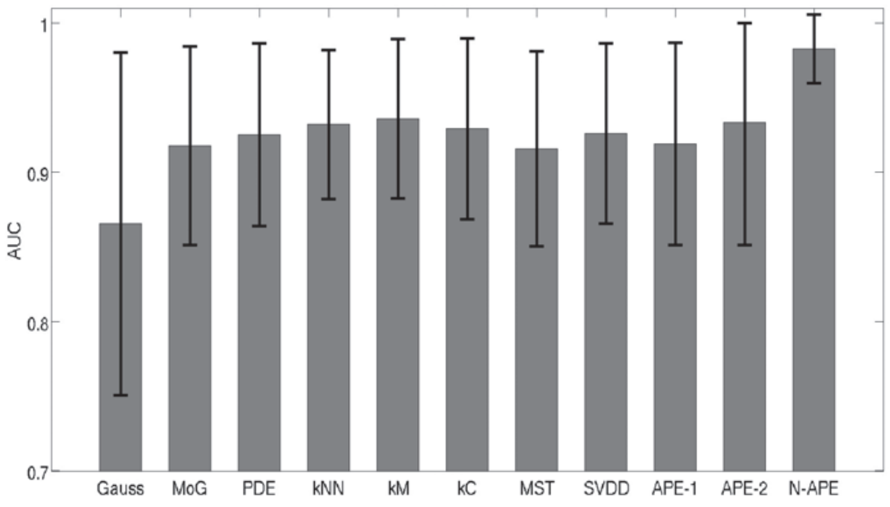


Figure 4.4: User verification Problem: AUC Mean value obtained on different one-class classifiers

```
Input: $M$ users;
a learning function $F$ ;
1 foreach $i = 1, \ldots, M$ do
2     Create a testing set TestSet with data from user $i$;
3     Create a training set TrainSet with data from all users except user $i$;
4     Train F on TrainSet;
5     Classify TestSet for classification performance estimation;
6 end
```

Algorithm 8: Leave-One-Person-Out Testing Protocol

Personalized Activity Classification In order to validate the personalized version of the activity classifier, a modification in the LOUO protocol has been done. The testing algorithm, called Folded-LOUO is shown in Algorithm 9. Once data of every subjects are randomly separate into N folds, Folded-LOPO iterates on both subjects and folds, using in this way all the examples of all the subjects for testing exactly once.

```
Input: M users; N users;
a incremental learning function F ;
1  foreach i=1,...,N do
2      foreach j=1,...,M do
3          Create a testing set TestUser with the iâth fold of user j;
4          Create a testing set TestRest with the i-th fold of all users except user j;
5          Create a training set TrainUser with all the folds of user j except the
           i-th fold;
6          Create a training set TrainRest with all the folds except the i-th fold of
           all users except for j-th user;
7          Train F on TrainRest;
8          Train F on TrainUser;
9          Classify TestUser for user-specific classification performance estimation;
10         Classify TestRest for no-user classification performance estimation;
11     end
12 end
```

Algorithm 9: Folded Leave-One-Person-Out Testing Protocol

User Verification User verification has been validated for both APE and Mixture of APEs algorithms. APE has been validated using a folded-LOPO validation strategy with the opportune modifications for being adapted to one-class classification. For each training step, there exist two testing sets, one related to the specific subject being tested and one related to all the other subjects. The testing protocol is shown in Algorithm 10. Mixture of APEs has been validated using a Leave-One-Walking-Out (LOWO) cross validation procedure. The testing algorithm is described in Table 11. For each user, a testing set with the walking data of all the other user is created. For every walking run of the user, a testing set is created using the current data and a training set is created for training a Mixture APE classifier. Results are averaged for every user.

4.5.3 Results of Personalized Activity Recognition Subsystem

Data acquisition setting A new dataset has been collected with BeaStreamer. The dataset contains data of the five ADLs from 15 different persons, with ages between 25-35, 5 women and 10 men. Differently to the previous dataset where people were completely free to perform the activities in the environment they selected without a predefined activity order, in this new dataset experimenters perform the

```
Input: M users; N users;
a One-Class learning function OC ;
1  foreach i=1,...,N do
2      foreach j=1,...,M do
3          Create a testing set TestUser with the iâth fold of user j;
4          Create a testing set TestRest with the i-th fold of all users except user j;
5          Create a training set TrainUser with all the folds of user j except the
           i-th fold;
6          Train OC on TrainUser;
7          Train F on TrainUser;
8          Classify TestUser for False Rejection Estimation;
9          Classify TestRest for False Acceptance Estimation;
10     end
11 end
```

Algorithm 10: Folded Leave-One-Person-Out Testing Protocol for One-Class

```
Input: $N$ users;
Given a Mixture $OC$ Learning Function ;
1  foreach $i = 1, \dots, N$ do
2      Create a testing set TestRest with the $i - th$ fold of all users except user $j$;
3      foreach $j = 1, \dots, numberOfRuns$ do
4          Create a testing set TestUser using walking run $j$;
5          Create a training set TrainUser using all walking runs except run $j$;
6          Train a Mixture $OC$ with TrainUser;
7          Classify TestUser for False Rejection Estimation;
8          Classify TestRest for False Acceptance Estimation;
9      end
10 end
```

Algorithm 11: Leave-One-Walking-Out Testing Algorithm

same sequence of activities in the same environment. Also in this case, ground-truth has been obtained by automatic labeling provided by subsequent start/stop phases of the device. Separation between activities is performed by means of one minute of "staying standing" activity. With this new dataset, 15 hours and 16 minutes of labeled activity have been obtained divided in:

- around 75 minutes of walking up/down stairs;
- around 300 minutes of walking;
- around 325 minutes of talking;
- around 60 minutes of staying standing;
- around 150 minutes of using a personal computer.

Performance Measurements: Accuracy, precision and recall have been chosen as classification performance measures. The quantities are defined in Equation 4.1, Equation 4.2 and Equation 4.3 in terms of the elements of the confusion matrix C [2].

$$\text{Accuracy}_i = \frac{C(i,i) + \sum_{l \neq i, m \neq i} C(l,m)}{\sum_{i,j} C(i,j)} \tag{4.1}$$

$$\text{Precision}_i = \frac{C(i,i)}{\sum_{j \neq i} C(j,i)} \tag{4.2}$$

$$\text{Recall}_i = \frac{C(i,i)}{\sum_{j \neq i} C(i,j)} \tag{4.3}$$

Experimental Results: The general activity recognition system performance are shown in Table 4.1. All the activities are classified with good performance. *Walking* is the activity with the best overall performance for all the metrics taken into account. This effect is expected since the features extracted, discussed in Section 4.6.1 have been specifically designed to capture subtleties in the walking pattern. The effect of personalization is shown in the Figure 4.5 with accuracy, precision and recall. Black bars represent performances obtained testing the baseline classifier on the subject, gray bars represent performances obtained testing the personalized classifier on the user and white bars represent performances obtained testing the personalized classifier on all the other subjects. Personalization ensures that the performances of the classifier are reduced for all the subjects except for the user.

[2] An element $C(i,j)$ of the confusion matrix accounts for the number of examples of class j classified as class i.

Table 4.1: Classification Performances for General Activity Classifier.

	Accuracy	**Precision**	**Recall**
Walking Up/Down Stairs	98.55 ± 0.7	98.01 ± 3.9	89.89 ± 8.32
Standing	98.01 ± 1.07	81.58 ± 33.3	55 ± 28.7
Talking	92.64 ± 3.7	87.41 ± 7.6	85.01 ± 15.3
Walking	99.52 ± 0.4	98.37 ± 1.5	99.43 ± 0.09
Using PC	92.7 ± 3.9	89.95 ± 6.5	86.8 ± 12.2

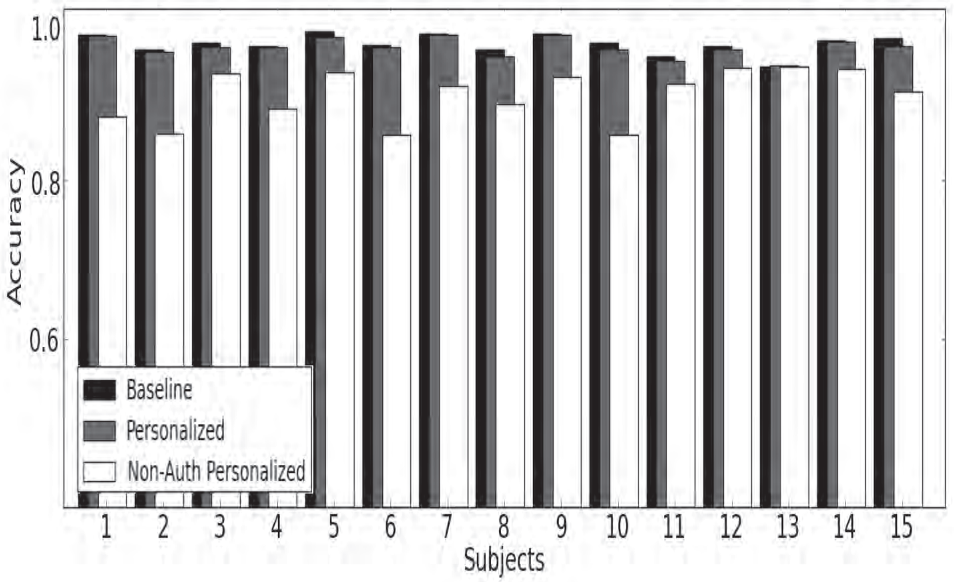


Figure 4.5: Accuracy of General Walking, Personalized and Non Personalized Classification

4.5.4 Results of User Verification subsystem

Data acquisition setting The validation of this subsystem has been performed using two different acquisition systems and conditions. The same acquisition system and data as in the validation of the personalized activity recognition subsystem are used. However, in order to have a more complete experimental section, an additional database has been collected using a different acquisition system with different experimental settings using an Android-based smart-phone.

Acceleration data from walking activities have been acquired using a Google Nexus One mobile phone with operating system Android 2.2 [Android(2007)]. Android-based mobile phones have an open Application Program Interface that allows programming the phone and accessing the sensors present in it. In this way, it is possible to read and save accelerometer data. Every sensor in the Android platform has an listener associated that delivers data when a change in its value happens. The delivery rate can be set to different frequency but this value is just an hint to the system. Events may be received faster or slower than the specified rate. In the setting for the experiments, accelerometer has been sampled with timestamps of approximatively 33 ms, using the mobile phone in normal mode, with all the network connections active.

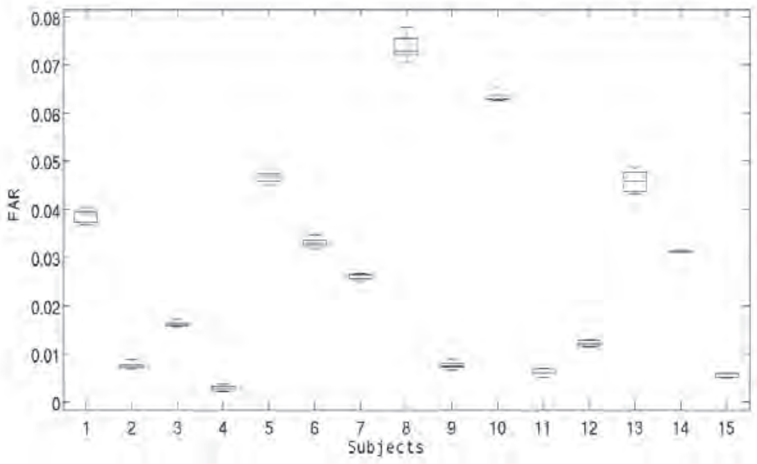
Data have been collected from 20 testers with ages between 25-35, 15 men and 5 women. Testers perform seven different runs of walking, for a total of 140 different walking runs. Testers were free to perform all the runs as they like. The mobile phone has been put in the jacket pocket on the chest. The acceleration axis are concordant to the specification of the Android platform and, in our setting, the Z axis refers to the direction concordant to the movement. The walking activity is performed in indoor, outdoor and urban environment. The walking scenarios are described in the following :

- 1 : Indoor corridor ;
- 2 : Outdoor street uphill and downhill ;
- 3 : Crowded flat urban street;
- 4 : Free flat urban street;
- 5 : Mixed scenario: Passing through doors from urban environment to indoor environment with people ;
- 6 : Mixed scenario: Semi-Indoor corridor with up and down ramps ;
- 7 : Walking in a garden with rough floor.

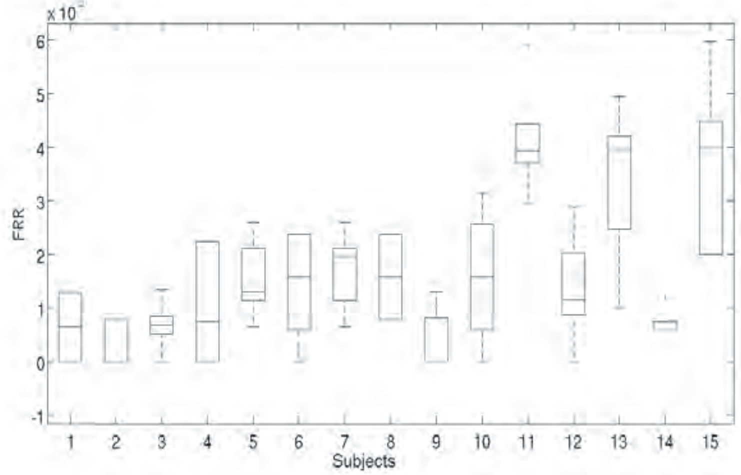
Performance measurements False Rejection Rate (FRR) and False Acceptance Rate (FAR) have been chosen as performance measures for verification. The FRR is defined as the percentage of identification instances in which false rejection occurs or the percentage of how many times an authorized user is not well verified. FAR is a measure of the likelihood that the system will wrongly accept an unauthorized user.

Experimental Results

Custom wearable system results The verification system has been evaluated on accelerometer data using a temporal frame of 35 seconds. In this way, results are achieved every 35 seconds of observing user data. Figure 4.6(a) and Figure 4.6(b) show the box plot representing the distribution of the cross-validation results for each user for both values FAR and FRR, respectively. For all the users, FAR values are below 0.08. There exist some variabilities between different users. Nevertheless, the wide majority of the users is under a 5% of FAR. The false rejection reaches even better results. All the users have FRR values below $6 \cdot 10^{-3}$. Using data acquired by the custom wearable device, the rejection of an authorized user is smaller than the acceptance of an intruder. This fact means that the system is able to verify an authorized user with very good performances but, more important, that the system provides very few false alarms. This feature is important for usability purposes because the aim of using biometrics is that the system would be as unobtrusive as possible.



(a)

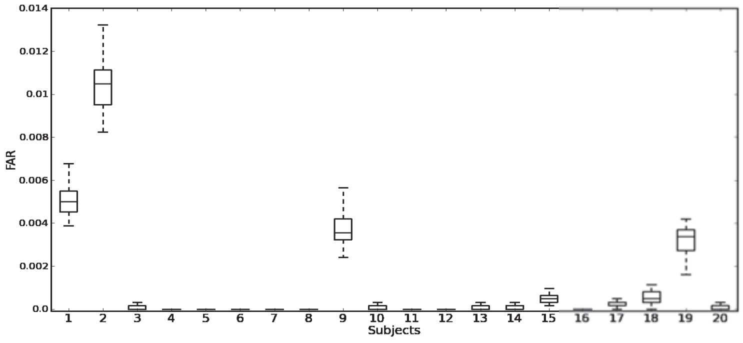


(b)

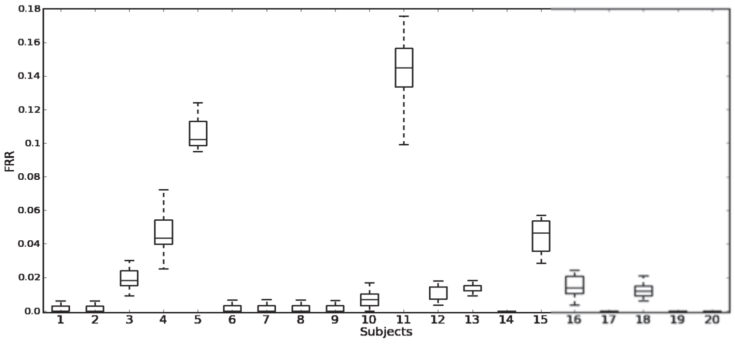
Figure 4.6: Verification Results using a custom wearable system: (a) False Acceptance Rate and (b) False Rejection Rate.

Android-based system results With the Android-based smart-phone, the system has been evaluated on the same temporal frame of 35 seconds. In Figure 4.7(a) and Figure 4.7(b) FAR and FRR distributions for each subject are shown in box plots.

The great majority of users FAR is below $6 \cdot 10^{-4}$. There is one pathological example in which its FAR is over 10^{-2}. The FRR is slightly worse but for the majority of users it is below $2 \cdot 10^{-2}$. Using the commercial device, each user is very well discriminated from the rest of users with FAR even smaller than those obtained in the case of the custom device. On the other side, FRR has values higher then in the previous case. Both these facts find their explanation in the data that the mobile phone provide. Accelerometer data collected from the smart-phones, being filtering before to delivery, have less noise than data provided by the custom device. This fact allows to verify the user with very good performances. Nevertheless, accelerometer data sampling does not follow a regular clock. This fact affects the quantity of data available in the temporal window taken into account and, consequently, the construction of an accurate boundary of the convex hull. For that reason, the probability that a sample falls outside the convex hull is higher and, hence, the system is prone to provide more false rejections.



(a)



(b)

Figure 4.7: Verification Results using a commercial device: (a) False Acceptance Rate and (b) False Rejection Rate.

4.6 Discussion

In this section, interesting questions are addressed regarding the verification process, such as the discriminability of the walking activity, the personalization step, the influence of the mixture model and the effect of the temporal ensemble.

4.6.1 Concerning the discriminability of the walking activity

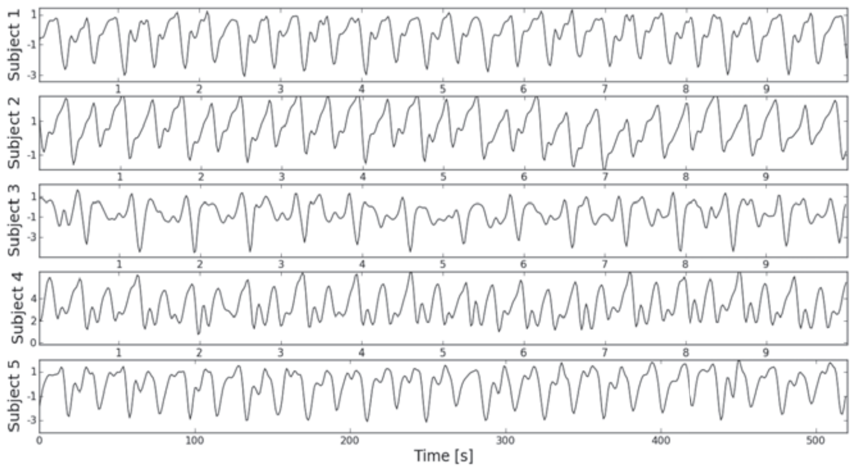


Figure 4.8: Accelerometer Data of Walking Activity for five different subjects

In Figure 4.8, walking activities for five different users are shown. The mean value of the time series depends on the position of the sensor and it represents the rotation of the accelerometer around the correspondent axis. The pattern related to the walking activity is clear but its shape depends on the subject. This pattern is representative of the walking cycle i.e. the step that a person performs after moving both legs. Systems like pedometers or step-counters are based on the information provided by this walking-cycle. The walking cycle duration depends on the velocity of walking. Inside the cycle, the swinging movement characterizes the difference between subjects. However, in some cases, it seems difficult to find a walking cycle, as for instance, in subject 4.

In every case, the shape of the time-series and its variations are characteristic of the subject. This fact is shown in Figure 4.9, where the acceleration time series related to the walking activity of a subject in different environmental conditions are shown.

In order to model walking activities from the users, no information related to the position of the sensor and the velocity of the specific activity performed have been

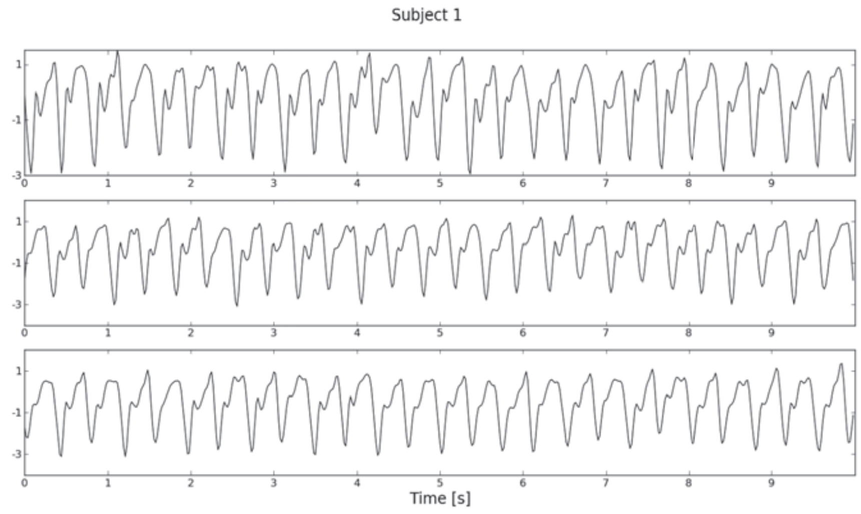


Figure 4.9: Accelerometer Data of Walking Activity for the same subject in different environmental conditions

used but only measures related to the variations of the oscillations in the acceleration time-series. The measures used are the following:

- difference between pairs of consecutive peaks ;
- difference between the value of consecutive upper-side peaks ;
- difference between the value of consecutive lower-side peaks .

The first feature can provide important information about the variation of the intensities in the acceleration during an activity. The second and third feature provide informations about the shape of the waveform. A further time series have been obtained computing the derivative of acceleration data. In physics, the derivative of acceleration is called Jerk [Sprott(2003)]. Jerk represents the rate of change of the force acting on a body. In Figure 4.10, an example of acceleration data and its correspondent derivative is shown, with peaks printed upon the data. The jerk also have been characterized using the measures previously described. Using those measures, the oscillatory movements typical of different activities and, at the same time, typical of the interpersonal differences are taken into account. The mean value of these measures computed for both acceleration and jerk has been used for activity classification. The standard deviation has been used for modeling the walking profile of authorized users. In Figure 4.11, the profiles of three different users modeled by their walking activity can be seen. The three users are perfectly separable and distinguishable. Six features have been computed for each acceleration axis, obtaining a 18-dimensional feature vector when the three acceleration axis are taken into account.

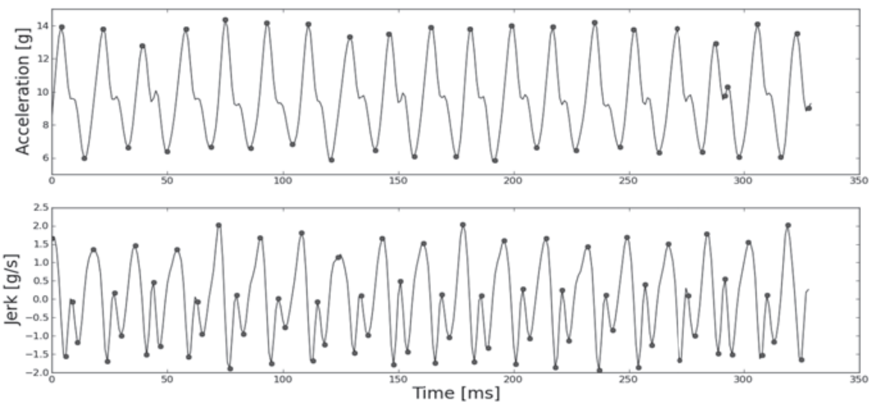


Figure 4.10: Acceleration Data and its Derivative

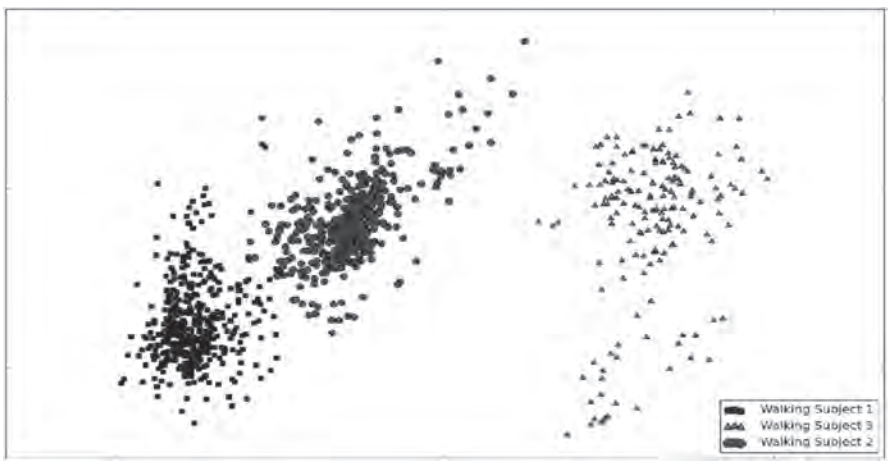


Figure 4.11: Localization of walking activity of three different subjects in a randomly projected plane.

4.6.2 With respect to personalization in the general activity recognition subsystem

After the personalization of the general activity recognition, the system is much more selective with respect to the new walking pattern. This effect reduces the performance of the activity classifier for the rest of the subjects and filters out part of the walking patterns from other users while keeping the discrimination performance for the desired user. At first glance, this effect could be surprising since one expects the performance for the verified user to increase. However, observe that the classification ratios are very high. Thus, even if the system further improves its verification rate the performance difference would still be small. By reducing the performance for all users except for the authorized one, this performance difference increases. This effect filters out non-walking patterns for all users. The reduction in recall makes the system accept more non-walking patterns as walking ones for the rest of users. Hopefully and effectively, the second verification stage, which is finely tuned to user walking pattern, rejects these last ones.

4.6.3 Concerning users walking variability and the effect of the layer-3 convex hull algorithm

Most of state-of-the-art methods use a very small number of runs from the users in the dataset when performing walking activity. Even worse, these runs are usually performed in very controlled conditions with little terrain variability. These data severely under-represents users walking variability and makes the task of user verification much more simple. For that reason, in this thesis we try to provide results using data acquired "in the wild" such as in rough terrains or in adversarial conditions, in crowded places or with obstacles. This feature is important because the variability of the walking pattern for one user can be very high. In Figure 2, a scatter plot of Feature 1 versus Feature 2 computed from the acceleration time series are shown. Features data points are relative to the same subject but they belong to two different walking paths. Although clusters are close, it is clear that using just one walking run is not enough for modeling users walking activity .

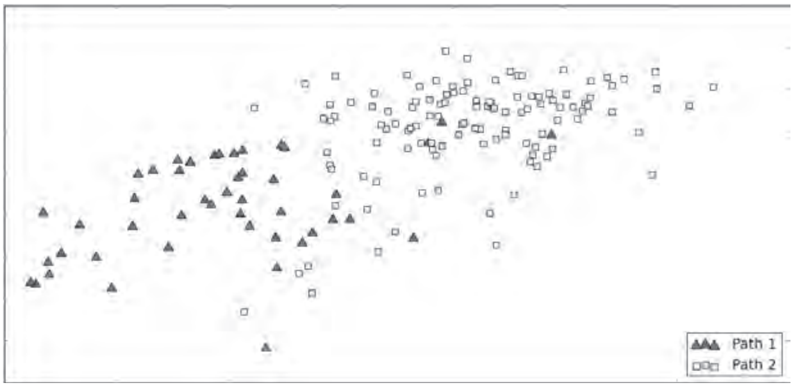


Figure 4.12: Walking runs in two different paths of the same subject plotted on a randomly projected plane

In the algorithm proposed, layer-3 uses a mixture of convex hulls in order to better represent user's walking pattern. Given a data set in a temporal frame of an authorized user, if this set is not well represented, a new model – layer-2 convex hull – is added to the user walking profile. Figure 4.13 shows the effect of adding up to six models to the user's profile on the false acceptance and false rejection ratios. These empirical results reinforce the intuition and claim stated in the former lines about the unsuitability of using one or two runs for verification purposes.

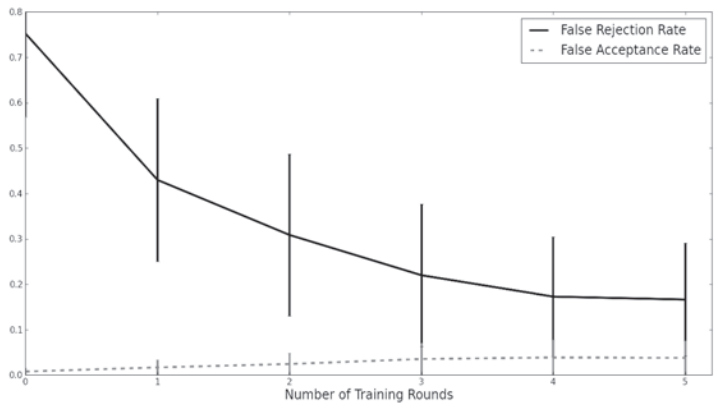


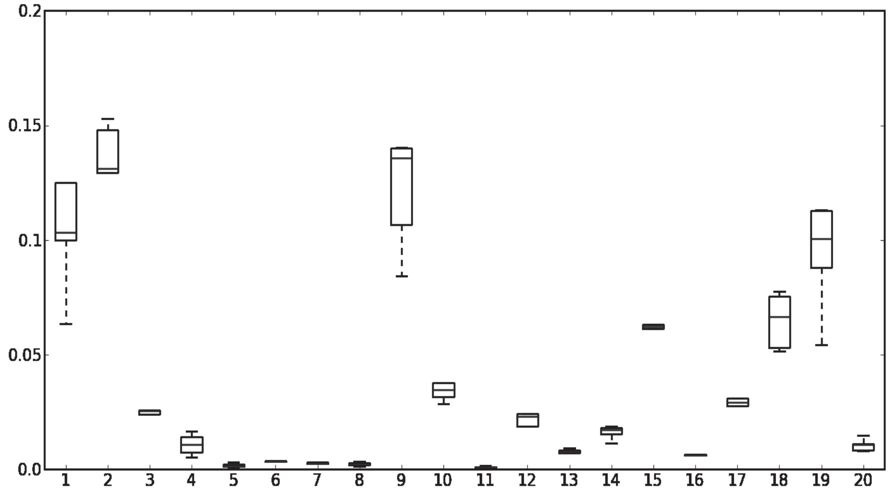
Figure 4.13: Average FAR and FRR over all users when building a layer-3 hull using a commercial device.

Observe that FRR is very high while the FAR is nearly zero at the first step. This confirms that the first model does not represent very well the diversity of the walking data of one user. As new data are observed and new models are added, the FRR drastically reduces and FAR is barely affected. The final results achieved after the construction of the layer-3 one-class verification system are: average FAR is 0.01604 and the average FRR is 0.3.

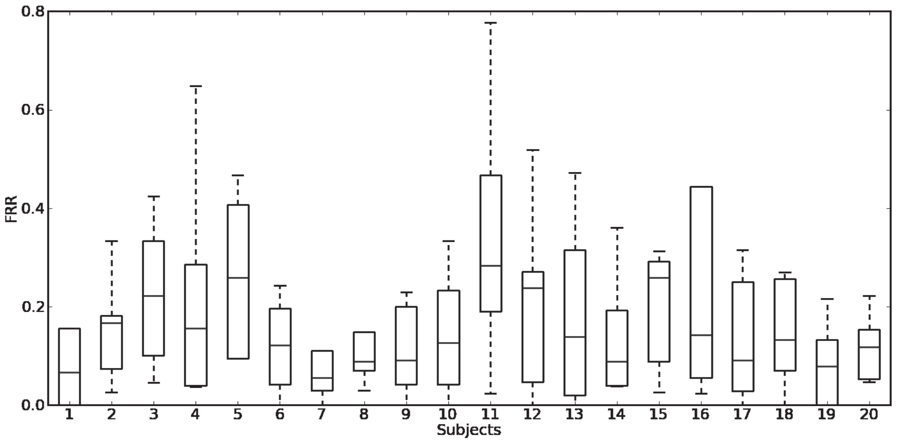
If we observe the distribution of FAR and FRR over all users in Figure 4.14, one observes that the worst case scenario for FAR is below 0.15 with six users over 0.05. If we consider FRR, the majority of the results are below 0.2. The former results were reported on an adversarial scenario and acquisition conditions. If we consider the case of the custom wearable device using a dedicated sensor working on high resolution with data adequately sampled, the same results change considerably. On the ideal scenario, the system is able to reject a legitimate user with value 0.006 ± 0.006 and accepts a non-authorized user with 0.058 ± 0.048.

4.6.4 Temporal coherence and the effect of the layer-4 convex hull algorithm

The final result of the verification system is greatly improved if one takes into account temporal coherence of the data sequence. Figure 4.15 shows FAR and FRR as time window increases. The value in the abscissa shows how many consecutive examples of the sequence are used in the majority voting ensemble. Using the Android-based system, 0.0001 of FAR is reached just after 55 seconds and 0.003 of FRR is reached after 150 seconds. This fact means that, in less than one minute, no intruders are allowed in the system and, in less than 2 minutes, the system has a very low probability to be wrong about the authorized user. Observe that both FAR and FRR are greatly



(a)



(b)

Figure 4.14: (a) False Acceptance Ratio (b) False Rejection Ratio for the layer-3 convex hull.

reduced as more examples are taken into account in both systems. If we consider the results reported in the former section before the time ensemble (Figure 4.14) and the final results in Figure 4.7, we observe a decrement of an order of magnitude in FAR and FRR in the results 'in the wild' and a reduction of half of those values in the custom scenario.

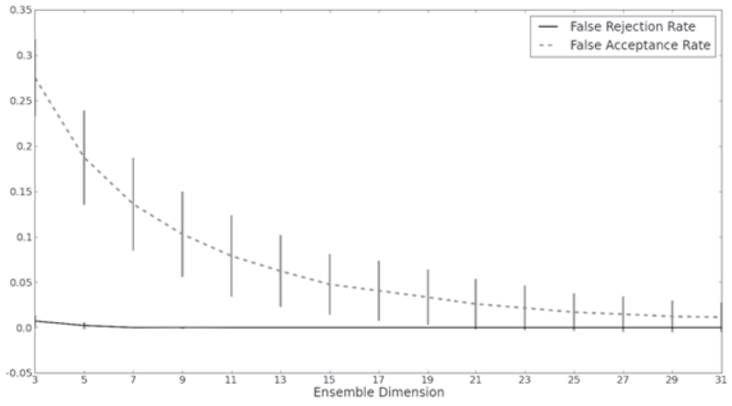
4.6.5 Concerning the state-of-the-art

In the following lines, we summarize the results and conditions for the most relevant state-of-the-art works. In Vildjiounaite et al. [Vildjiounaite et al.(2007)], experiments with voice, gait and fingerprint data have shown that in most cases FAR is around 0.01 and FRR is around 0.03. However, these results are the composite of three verification systems. If we focus on the accelerometer verification, they report an EER of 0.137. Note also that the data set is created walking along a 20 meters corridor. In Mäntyjärvi et al. [Mäntyjärvi et al.(2005)] 36 test subject for recognition walked with fast, normal and slow walking speeds in two sessions wearing the accelerometer device on their belt, at back. The best equal error rate $EER = 0.07$ was achieved with a signal correlation method on two runs of 20 meters per walking speed. Derawi et al. [Derawi et al.(2010)] used a commercially available mobile device. The system was evaluated having 51 volunteers and resulted in an equal error rate of 0.20 with the system attached to the belt with two runs for each user in in-vitro conditions (about 37 meters down the hall on flat carpet). Finally, Gafurov et al. [Gafurov et al.(2006)] attached the wearable system to the hip of 22 subjects performing six rounds of walking at normal speed on a flat floor. They report EER of 0.16.

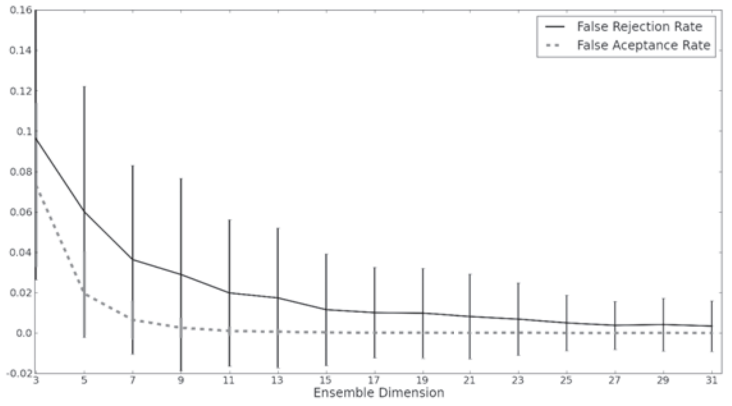
Observing the former works, the general state-of-the-art verification rate is around 0.1 of equal error rate, which means that a decrement on one of the two parameters, FAR or FRR, worsens the value for the other. Observe, that using our custom system the reported values for FRR and FAR are 0.007 and 0.03 which is far smaller than the best error rates reported in literature. Note that the EER is between those values. However, EER is simple to obtain if the system is parameterizable with just one parameter. In our case different parameters would result in many different decision error trade-off curves. Thus, we choose to report the most honest results from the cross-validation tuning of the parameters. Using the Android-based wearable device in the 'wild' scenario, we achieve values of FAR and FRR of 0.0001 and 0.003, respectively. Again, in this scenario the results are far better than the best reported, but in a hard and adversarial scenario with several obstacles.

4.7 Conclusions

In this final chapter, a novel personalized technique for user authentication and verification using gait as a biometric unobtrusive pattern has been proposed. The method is based on a two stages pipeline. In the first level, an activity recognition classifier is able to distinguish the walking pattern of any user with respect to other activities. This classifier is personalized for a specific user using a small sample of her walking pattern. As a result, the system is much more selective with respect to the new walking pattern. This effect reduces the performance of the activity classifier for the rest of the subjects. This phase of personalization effectively filters out part of the walking patterns from other users while keeping the discrimination performance for the desired user. A second stage verifies whether the user is an authorized user or not. This stage is defined as a one-class classification problem. In order to solve this problem, a four layer architecture is built around the APE methodology. Each layer covers different needs. The first layer concerns computational complexity and storage requirements. The second layer improves the performance by adding robustness to noise and outliers by means of a bagging ensemble procedure. The third layer allows the use of multiple APEs for modeling non-convex shapes. Finally, the fourth layer takes into account temporal coherence to boost the results of the verification system. Two different scenarios have been used for validation with two different wearable systems. A custom high-performance wearable system is built and used in a free environment. Using this system, a data set of ten users is gathered during several days. A second dataset is acquired from an Android based commercial device. In this last experiment, twenty subjects freely perform seven runs in a “wild” scenario with rough terrains, adversarial conditions, in crowded places and with obstacles. Results on both systems and datasets are FRR = 0.0072 and FAR = 0.03 for the custom wearable system, and FRR = 0.02 and FAR = 0.001 for the commercial system. These results are very encouraging and, to the best of our knowledge, improve the verification rates from gait patterns compared with state of the art techniques. All the results are obtained setting the accelerometer sensor in the upper torso of a person. Future works would plan on extending the system for handling data acquired from the sensor located at different parts of the body. Another interesting problem arises when the number of authorized users increases in the system. A general verification system considering just authorized users patterns is not feasible, since FAR will redoubtably increase due to the fact that there is more feature space considered as authorized. An effective way of handling this problem is to create and automatically select user profiles.



(a)



(b)

Figure 4.15: FAR and FRR evolution as the temporal ensemble size increase in (a) the custom wearable system and (b) in a commercial device

Chapter 5

Conclusions

This thesis has focused its attention into the main topics of reducing the learning complexity in ensemble of classifiers and recognizing physical activity. The main achievements of this work are shown in Figure 5.1.

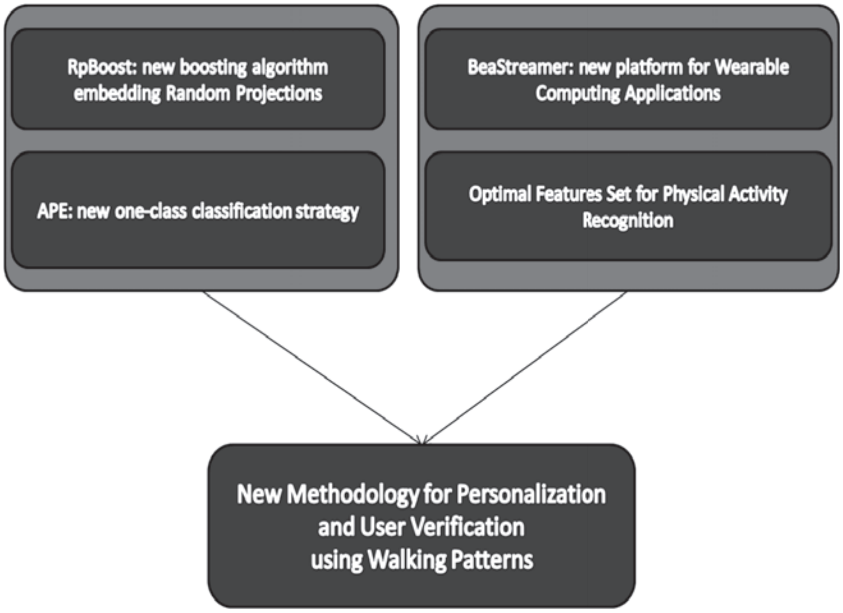

RpBoost: new boosting algorithm embedding Random Projections
APE: new one-class classification strategy
BeaStreamer: new platform for Wearable Computing Applications
Optimal Features Set for Physical Activity Recognition
New Methodology for Personalization and User Verification using Walking Patterns

Figure 5.1: Achievement of this Thesis

Starting from a study on Random Projections, a new algorithm called **RpBoost** has been developed. RpBoost embeds Random Projections with the aim of reducing the dimensionality of the classification problem at hand and generating diversity in the construction of ensemble. At each step of the optimization process, data projected in a random space are fitted with the classifier that best approximate data in that space. RpBoost performs better than usual boosting approach on synthetic data and in many real problems. Results show that projections drawn from a standard Normal Distribution $N(0, 1)$ provide the best results when compared to other type of projections. A new methodology for one-class classification problems, the **Approximate Polytope Ensemble (APE)** and its extension, the Non-convex Approximate Polytope Ensemble (NAPE) have been also presented. APE uses a convex hull to model the boundaries of one-class classification problems. Expansion and contraction of the original polytope governed by a parameter α, allows to avoid over-fitting. The high computational complexity needed for building the convex hull in high dimensional spaces is overcome using the Random Projection technique. The multi-dimensional structure modeling the boundaries of the problem is approximated projecting data down to bi-dimensional spaces. In those spaces, building the convex hull and checking if a point lies inside the polygon are well know problems with very efficient solutions. NAPE extends this approach using a tiling strategy of convex patches able to approximate the original non-convex structure. Experimental results have shown that APE and NAPE represent a good general methodology for one-class classification problems, being competitive and many times outperforming state-of-the-art methodologies in one-class classification. The low computational complexity of training and testing makes APE suitable to be used in system with limited computational resources.

BeaStreamer, a new platform for wearable computing applications, has been developed for carrying out experiments for physical activity recognition. Although the system was initially designed for acquisition tasks, its high computational capabilities allow to use the system as a powerful and complete wearable computer. BeaStreamer senses the environment using a standard low-cost web-cam and monitor user activity by mean of an accelerometer, being able to provide Intelligent feedback to the user using audio and visual information. Using BeaStreamer, data of different physical activities can be easily collected for large amount of time, ensuring user comfortableness. The task of **classifying physical activities** has also been carried out. Using BeaStreamer, five common Activities of Daily Living have been successful classified with high classification performances. These performances are comparable with the state-of-the-art in physical activity recognition, specially taking into account that a single classifier is used instead of more complex classification architectures. These high performances are principally due to the features used in the classification process. Particularly for motion data, features selected provide a set of meaningful features able to give physical meaning to the quantities involved into the classification process. Using the only set of motion features selected, good classification performances are obtained using the only accelerometer.

Finally, a novel **personalized technique for user authentication and verification** using gait as a biometric unobtrusive pattern has been presented. The method is based on a two stages machine learning pipeline. In the first level, a personalized activity recognition classifier is able to distinguish walking patterns in a much more selective selective way for the authorized user than for others. The second stage verifies the user. This stage is based on the APE methodology and is composed of three more layers, each one devoted to tackle a specific problem arising from the pure application of APE. The final verification system improves its performance by adding robustness to noise and outliers, it is capable to manage situations where different walking runs are performed in different environments and, finally, it improves itself on a temporal scale. Results obtained show that the system significantly improves the verification rates from gait patterns when compared with state of the art techniques. This encouraging results induce to implement the system for a large scale testing phase, in order to verify its effectiveness in a wider range of situations.

At the end of a research, two types of findings should always be outlined. The first ones are related to all the technical results, theoretical and methodological, which have been reached during the work. This type of conclusions, previously listed, are pretty simple and immediate to show. Nevertheless, there exist conclusions much more laborious to be drawn. These last findings, often omitted, represent the philosophical[1] contributions that the research has given to the researcher himself. In this work, a complete system has been developed starting from the hardware up to the highest intelligent layer. This process, conducted under a practical "problem-solving" approach, led to aspects that may have interesting implications from the theoretical points of view. At first, a theoretical study on the APE methodology presented in this thesis should be conducted from the Computational Learning Theory point of view. Experiments performed point out that the number of projections used for learning the concept at hand is a quantity that might be strongly related to the robustness of the concept to be learn. Moreover, the subtle intuition that random projections provide is that the particular problem at hand benefits of some particular (random) projections more than others but the randomness of the process prevents to see this relationship clearly. On the other side, the evidence that users can be modeled by its walking patterns opens a door towards biometrics authentications that allow constant verification mechanisms. This fact, consonant with the increasing human-computer symbiosis every day more knit, could bring to a new form of interaction. Under the Ubiquitous Computing guideline, the trend seems to go towards systems personalized on the users, able to constantly learn from them and augment their perception, as well as first wearable computers hoped.

[1]Here the term "philosophical" is intended with its original meaning of "love for wisdom".

Appendix 1

Common Performance Measures in Machine Learning

A commonly used technique for validating classification results is N-fold *cross-validation*. In N-fold cross-validation, data are partitioned into N approximative equally sized segments (or folds). Subsequently N iterations of training and validation are performed such that within each iteration a different fold of the data is held-out for validation while the remaining $N - 1$ folds are used for learning. The results of the validation process can be summarized in the *Confusion Matrix*(CM). The typical structure of a bi-dimensional CM is show in Table 5, where the columns represent the instances of the predicted class and the rows represent the instances of the true class. Using CM, confusions between predicted classes can be easily visualized. Many quantitative measures of classification performances can be derived directly from CM. *Accuracy* is the most common measure of classification performances. Nevertheless, when many classes are involved into the classification process, quantities like *Precision* and *Recall* might be also interesting to use. Precision and Recall provide a measure about how much the prediction for each class is consistent with respect false positives and false negatives, respectively. The *F-Measure* combines precision and recall. All these quantities are defined in Table 5.

ROC Curve and Area Under ROC

The Receiver Operating Characteristic (ROC) curve is the representation in a plot of the sensitivity and 1-specificity. In the context of binary classifier system, ROC is equivalent to plot the fraction of true positive computed over all the positive values (as known as the True Positive Rate) obtained versus the fraction of false positive computed over all the negative (also known as False Positive Rate). The variation of True Positive and False Positive is discriminated as the threshold varies. The area under the ROC curve represents a performance measure for binary classifiers.

Basic Statistics Measure

Let X be a random variable taking random value from a finite dataset of n elements,

	Ground-truth Positive Label	Ground-truth Negative Label
Predicted Positive Label	**True Positive (TP)**	**False Negative (FN)**
Predicted Negative Label	**False Positive (FP)**	**True Negative (TN)**

Table 1: Confusion Matrix

$$\begin{aligned} Accuracy &= \tfrac{TP+TN}{TP+TN+FP+FN} \\ Precision &= \tfrac{TP}{TP+FP} \\ Recall &= \tfrac{TP}{TP+FN} \\ F-Measure &= 2\cdot\tfrac{Precision\cdot Recall}{Precision+Recall} \end{aligned}$$

Table 2: Classification Performances Measures

$X = x_1, \cdots, x_n$, the **expcted value** or average of X, denoted by E, is provided using the formula in Equation 1.

$$\overline{x} = E[X]\frac{1}{n}\sum_{i=1}^{n} xi \tag{1}$$

The **standard deviation** of X is represented by the quantity provided by Equation 2. High standard deviation indicates that points are far and spread from the mean. On the other hand, small standard deviation indicates that points are clustered closely around the mean.

$$\sigma = \sqrt{E[(X-\overline{x})^2]} = \sqrt{\frac{1}{N}\sum_{i=1}^{n}(x_i-\overline{x})^2} \tag{2}$$

A measure about asymmetry of the probability distribution of a real-valued random variable is provided by the **skewness**, defined in Equation 3. The skewness for a normal distribution is zero, and any symmetric data should have a skewness near zero. Negative values for the skewness indicate data that are skewed left and positive values for the skewness indicate data that are skewed right. By skewed left, we mean that the left tail is long relative to the right tail. Similarly, skewed right means that the right tail is long relative to the left tail.

$$skewness = \sqrt{E\left[\left(\frac{X-\overline{x}}{\sigma}\right)^3\right]} = \frac{\sum_{i=1}^{n}(x_i-\overline{x})^3}{(n-1)\sigma^3} \tag{3}$$

Kurtosis is a measure of the peakedness of the probability distribution of the real-valued random variable X. Kurtosis characterizes the relative peakedness or flatness of a distribution compared to the normal distribution. Positive kurtosis indicates a relatively peaked distribution. Negative kurtosis indicates a relatively flat distribution.

$$kurtosis = \sqrt{E\left[\left(\frac{X-\overline{x}}{\sigma}\right)^3\right]} = \frac{\sum_{i=1}^{n}(x_i-\overline{x})^4}{(n-1)\sigma^4} \tag{4}$$

Singular Value Decomposition

Singular Value Decomposition is a way of factoring matrices into a series of linear approximations that expose the underlying structure of the matrix. Let A be an mxn matrix, where $m \geq n$. Then A can be decomposed as follows:

$$A = UWV^T \tag{5}$$

where U is a mxn orthonormal matrix: $UU^T = I_m$, W is a nxn diagonal matrix, and V is an nxn orthogonal matrix, $VV^T = I_n$.

$$W = \begin{bmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & 0 \\ 0 & 0 & \cdots & w_n \end{bmatrix} \tag{6}$$

The diagonal elements of W are called the *singular values.* The singular value decomposition exists always and is unique.

Wavelet Transform

The wavelet transform was developed as a need for further improvements of Fourier transforms. Wavelets transform signals in the time domain to a joint time-frequency domain. The main weakness that was found in Fourier transforms was their lack of localized support, which made them susceptible to Heisenberg's Uncertainty principle. It means that we could get information about the frequencies present in a signal, but not where and when the frequencies occurred. Wavelets, on the other hand, are not anywhere as subject to it. A wavelet is, as the name might suggest, a little piece of a wave. Where a sinusoidal wave as is used by Fourier transforms carries on repeating itself for infinity, a wavelet exists only within a finite domain, and is zero-valued elsewhere. A wavelet transform involves convolving the signal against particular instances of the wavelet at various time scales and positions. Since we can model changes in frequency by changing the time scale, and model time changes by shifting the position of the wavelet, we can model both frequency and location of frequency. This provides the resulting domain as a joint time-frequency domain. Performing these convolutions at every position and every characteristic scale is called the continuous wavelet transform. This represents a costly process. Fortunately, most signal data are stored discretely. Moreover, Nyquist's theorem tells us that the highest frequency we can model with discrete signal data is half that of the sampling frequency. So that means we, at worst, have to perform the transform at every other point. The continuous wavelet transform is generally expressed with the integral in Equation 7.

$$CWT_x^{\Psi}(\tau, s) = \frac{1}{\sqrt{|s|}} \int x(t) \Psi \left(\frac{t - \tau}{s} \right) dt \tag{7}$$

In the discrete wavelet transform, a low-pass(scaling function) and a high-pass (wavelet function) version of the signal are computed separating the high-pass and low-pass information. The discrete wavelet transform behaves like a bank of filters. Iterating downward along the low-pass sub-band, the low-pass data are treated as a new signal subdivided into its own low and high sub-bands. The Haar wavelet is probably the most basic wavelet. It is equivalent to a sum-difference transformation. The Daubechies-4 is probably the shape most associated with the word wavelet.

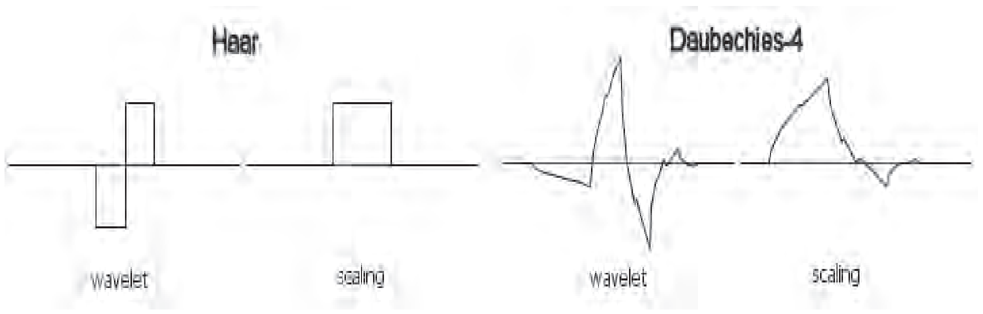


Figure 2: Classic Wavelets: Haar (left) and Daubechies-4 (right)

Appendix 2: Numerical Results obtained on Approximate Polytope Ensemble

In this Appendix, numerical results of the experiments performed in this thesis are reported.

Table 3: Comparative Results: AUC obtained on Artificial Datasets with 500 datapoints.

	Normal	**Banana**	**Esse**	**Tre**	**Toro**
Gauss	0.963 ± 0.008	0.940 ± 0.014	0.952 ± 0.009	0.953 ± 0.008	0.905 ± 0.010
MoG	0.962 ± 0.008	0.966 ± 0.007	0.973 ± 0.006	0.970 ± 0.009	0.936 ± 0.012
PDE	0.962 ± 0.008	0.966 ± 0.006	0.973 ± 0.005	0.970 ± 0.009	0.936 ± 0.011
kNN	0.947 ± 0.012	0.955 ± 0.009	0.966 ± 0.007	0.965 ± 0.008	0.926 ± 0.013
kM	0.956 ± 0.008	0.959 ± 0.007	0.969 ± 0.005	0.967 ± 0.010	0.930 ± 0.010
kC	0.959 ± 0.011	0.954 ± 0.009	0.962 ± 0.010	0.963 ± 0.008	0.924 ± 0.013
MST	0.938 ± 0.015	0.952 ± 0.011	0.965 ± 0.007	0.964 ± 0.007	0.925 ± 0.015
SVDD	0.959 ± 0.006	0.955 ± 0.010	0.954 ± 0.007	0.952 ± 0.007	0.911 ± 0.013
APE-ONE	0.921 ± 0.031	0.867 ± 0.048	0.926 ± 0.019	0.892 ± 0.045	0.883 ± 0.045
APE-TWO	0.959 ± 0.016	0.863 ± 0.078	0.962 ± 0.010	0.952 ± 0.015	0.824 ± 0.054
N-APE	0.968 ± 0.005	0.974 ± 0.008	0.974 ± 0.009	0.978 ± 0.007	0.940 ± 0.027

Table 4: Comparative Results: AUC obtained on Artificial Datasets with 750 datapoints.

	Normal	**Banana**	**Esse**	**Tre**	**Toro**
Gauss	0.963 ± 0.006	0.942 ± 0.008	0.951 ± 0.011	0.955 ± 0.008	0.915 ± 0.013
MoG	0.963 ± 0.006	0.967 ± 0.005	0.972 ± 0.007	0.972 ± 0.006	0.943 ± 0.009
PDE	0.963 ± 0.006	0.967 ± 0.005	0.972 ± 0.006	0.970 ± 0.006	0.944 ± 0.008
kNN	0.951 ± 0.009	0.957 ± 0.008	0.966 ± 0.007	0.965 ± 0.005	0.931 ± 0.009
kM	0.959 ± 0.011	0.963 ± 0.006	0.969 ± 0.007	0.969 ± 0.006	0.936 ± 0.009
kC	0.962 ± 0.008	0.956 ± 0.009	0.961 ± 0.004	0.963 ± 0.011	0.924 ± 0.012
MST	0.943 ± 0.011	0.954 ± 0.008	0.964 ± 0.006	0.963 ± 0.005	0.928 ± 0.010
APE-ONE	0.921 ± 0.038	0.874 ± 0.060	0.934 ± 0.033	0.875 ± 0.055	0.925 ± 0.011
APE-TWO	0.960 ± 0.007	0.888 ± 0.077	0.963 ± 0.013	0.957 ± 0.009	0.835 ± 0.039
N-APE	0.965 ± 0.008	0.975 ± 0.004	0.978 ± 0.005	0.951 ± 0.003	0.965 ± 0.008

Table 5: Comparative Results: AUC obtained on Artificial Datasets with 1000 datapoints.

	Normal	Banana	Esse	Tre	Toro
Gauss	0.963 ± 0.003	0.942 ± 0.009	0.951 ± 0.009	0.955 ± 0.007	0.915 ± 0.008
MoG	0.963 ± 0.003	0.967 ± 0.004	0.972 ± 0.006	0.972 ± 0.005	0.943 ± 0.005
PDE	0.963 ± 0.003	0.967 ± 0.004	0.972 ± 0.006	0.970 ± 0.004	0.944 ± 0.006
kNN	0.951 ± 0.007	0.957 ± 0.007	0.966 ± 0.006	0.965 ± 0.006	0.931 ± 0.006
kM	0.959 ± 0.006	0.963 ± 0.005	0.969 ± 0.007	0.969 ± 0.005	0.936 ± 0.006
kC	0.962 ± 0.003	0.956 ± 0.008	0.961 ± 0.006	0.963 ± 0.004	0.924 ± 0.010
MST	0.943 ± 0.009	0.954 ± 0.007	0.964 ± 0.007	0.963 ± 0.006	0.928 ± 0.007
APE-ONE	0.921 ± 0.019	0.874 ± 0.057	0.934 ± 0.028	0.875 ± 0.068	0.925 ± 0.011
APE-TWO	0.960 ± 0.016	0.888 ± 0.038	0.963 ± 0.006	0.957 ± 0.020	0.835 ± 0.058
N-APE	0.965 ± 0.013	0.975 ± 0.003	0.978 ± 0.004	0.951 ± 0.002	0.965 ± 0.002

Table 6: Comparative Results: AUC obtained on D^* on Gauss, MoG, PDE, kNN, kM and kC.

	Gauss	MoG	PDE	kNN	kM	kC
6	0.887 ± 0.086	0.900 ± 0.086	0.921 ± 0.081	0.924 ± 0.078	0.925 ± 0.077	0.922 ± 0.076
7	0.899 ± 0.110	0.875 ± 0.107	0.893 ± 0.094	0.894 ± 0.097	0.875 ± 0.109	0.894 ± 0.114
8	0.855 ± 0.102	0.841 ± 0.107	0.937 ± 0.077	0.937 ± 0.075	0.938 ± 0.074	0.929 ± 0.080
9	0.891 ± 0.076	0.793 ± 0.109	0.834 ± 0.085	0.813 ± 0.082	0.842 ± 0.086	0.912 ± 0.075
10	0.871 ± 0.069	0.851 ± 0.079	0.889 ± 0.065	0.885 ± 0.064	0.879 ± 0.078	0.885 ± 0.070
11	0.752 ± 0.068	0.780 ± 0.070	0.765 ± 0.076	0.780 ± 0.081	0.767 ± 0.092	0.794 ± 0.080
12	0.583 ± 0.047	0.608 ± 0.047	0.585 ± 0.049	0.606 ± 0.055	0.573 ± 0.052	0.562 ± 0.041
13	0.511 ± 0.051	0.522 ± 0.047	0.474 ± 0.045	0.511 ± 0.043	0.494 ± 0.051	0.518 ± 0.071
28	0.625 ± 0.063	0.630 ± 0.077	0.720 ± 0.065	0.720 ± 0.066	0.702 ± 0.059	0.655 ± 0.072
29	0.653 ± 0.084	0.643 ± 0.090	0.674 ± 0.082	0.676 ± 0.082	0.666 ± 0.080	0.608 ± 0.111
30	0.626 ± 0.081	0.641 ± 0.095	0.721 ± 0.083	0.723 ± 0.083	0.694 ± 0.080	0.650 ± 0.084
31	0.654 ± 0.078	0.639 ± 0.079	0.678 ± 0.073	0.681 ± 0.072	0.662 ± 0.088	0.613 ± 0.106
35	0.977 ± 0.022	0.964 ± 0.028	0.971 ± 0.020	0.971 ± 0.020	0.964 ± 0.025	0.963 ± 0.027
36	0.873 ± 0.072	0.841 ± 0.077	0.798 ± 0.066	0.842 ± 0.072	0.797 ± 0.079	0.770 ± 0.079
37	0.926 ± 0.051	0.808 ± 0.089	0.676 ± 0.085	0.713 ± 0.088	0.663 ± 0.096	0.679 ± 0.152
38	0.606 ± 0.060	0.675 ± 0.067	0.669 ± 0.068	0.674 ± 0.062	0.637 ± 0.082	0.589 ± 0.096
39	0.510 ± 0.066	0.439 ± 0.076	0.433 ± 0.067	0.426 ± 0.060	0.432 ± 0.082	0.645 ± 0.090
40	0.602 ± 0.074	0.781 ± 0.098	0.720 ± 0.085	0.761 ± 0.086	0.702 ± 0.077	0.650 ± 0.078
41	0.801 ± 0.085	0.763 ± 0.101	0.775 ± 0.087	0.774 ± 0.089	0.761 ± 0.101	0.683 ± 0.084
42	0.763 ± 0.091	0.809 ± 0.132	0.802 ± 0.083	0.821 ± 0.075	0.796 ± 0.084	0.722 ± 0.107
43	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.385 ± 0.082
44	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.957 ± 0.038
45	0.973 ± 0.022	0.934 ± 0.044	0.941 ± 0.036	0.951 ± 0.034	0.937 ± 0.041	0.938 ± 0.042
46	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.998 ± 0.004	0.993 ± 0.012
47	0.990 ± 0.013	0.979 ± 0.023	0.975 ± 0.019	0.975 ± 0.017	0.971 ± 0.024	0.973 ± 0.022
48	0.486 ± 0.057	0.426 ± 0.054	0.206 ± 0.043	0.239 ± 0.040	0.412 ± 0.057	0.476 ± 0.056
49	0.470 ± 0.056	0.419 ± 0.046	0.230 ± 0.037	0.261 ± 0.033	0.410 ± 0.047	0.482 ± 0.048
50	0.626 ± 0.045	0.689 ± 0.047	0.696 ± 0.044	0.698 ± 0.037	0.678 ± 0.062	0.560 ± 0.056
51	0.518 ± 0.044	0.619 ± 0.047	0.675 ± 0.035	0.667 ± 0.031	0.608 ± 0.055	0.564 ± 0.053
52	0.460 ± 0.051	0.408 ± 0.063	0.215 ± 0.031	0.241 ± 0.029	0.396 ± 0.051	0.480 ± 0.046
53	0.502 ± 0.053	0.442 ± 0.051	0.249 ± 0.035	0.267 ± 0.032	0.428 ± 0.048	0.492 ± 0.038
54	0.798 ± 0.045	0.767 ± 0.068	0.783 ± 0.048	0.770 ± 0.050	0.768 ± 0.055	0.768 ± 0.067
55	0.666 ± 0.065	0.671 ± 0.079	0.653 ± 0.057	0.628 ± 0.058	0.635 ± 0.069	0.830 ± 0.102
63	0.722 ± 0.071	0.770 ± 0.087	0.710 ± 0.083	0.782 ± 0.075	0.724 ± 0.085	0.692 ± 0.088
64	0.553 ± 0.103	0.612 ± 0.115	0.607 ± 0.105	0.678 ± 0.108	0.593 ± 0.103	0.541 ± 0.107
65	0.496 ± 0.097	0.668 ± 0.077	0.678 ± 0.095	0.720 ± 0.088	0.662 ± 0.094	0.651 ± 0.107
66	0.754 ± 0.038	0.651 ± 0.050	0.340 ± 0.036	0.742 ± 0.034	0.746 ± 0.055	0.571 ± 0.043
67	0.431 ± 0.026	0.543 ± 0.034	0.330 ± 0.022	0.558 ± 0.032	0.420 ± 0.034	0.530 ± 0.042
68	0.618 ± 0.037	0.650 ± 0.043	0.488 ± 0.038	0.669 ± 0.043	0.601 ± 0.056	0.577 ± 0.049
80	0.988 ± 0.013	0.981 ± 0.017	0.995 ± 0.006	0.995 ± 0.006	0.994 ± 0.009	0.992 ± 0.010
81	1.000 ± 0.000	1.000 ± 0.000	0.997 ± 0.006	0.997 ± 0.005	0.992 ± 0.011	0.990 ± 0.013
82	0.951 ± 0.033	0.927 ± 0.049	0.883 ± 0.059	0.870 ± 0.062	0.873 ± 0.052	0.872 ± 0.072

Table 7: Comparative Results: AUC obtained on D^* on MST, SVDD, APE-1, APE-2 and N-APE

	MST	**SVDD**	**APE-1**	**APE-2s**	**N-APE**
6	0.927 ± 0.080	0.895 ± 0.082	0.934 ± 0.013	0.867 ± 0.154	0.867 ± 0.188
7	0.899 ± 0.094	0.899 ± 0.095	0.916 ± 0.025	0.836 ± 0.202	0.836 ± 0.236
8	0.944 ± 0.076	0.946 ± 0.078	0.938 ± 0.012	0.808 ± 0.190	0.808 ± 0.241
9	0.833 ± 0.084	0.881 ± 0.081	0.868 ± 0.099	0.843 ± 0.174	0.843 ± 0.174
10	0.878 ± 0.072	0.836 ± 0.077	0.864 ± 0.141	0.768 ± 0.229	0.819 ± 0.229
11	0.769 ± 0.076	0.755 ± 0.080	0.590 ± 0.183	0.581 ± 0.183	0.699 ± 0.224
12	0.608 ± 0.055	0.618 ± 0.050	0.526 ± 0.068	0.619 ± 0.110	0.627 ± 0.116
13	0.514 ± 0.041	0.538 ± 0.052	0.551 ± 0.064	0.667 ± 0.062	0.670 ± 0.062
28	0.719 ± 0.061	0.680 ± 0.059	0.622 ± 0.148	0.723 ± 0.117	0.730 ± 0.117
29	0.690 ± 0.078	0.701 ± 0.065	0.746 ± 0.069	0.674 ± 0.091	0.674 ± 0.091
30	0.724 ± 0.081	0.679 ± 0.077	0.630 ± 0.119	0.743 ± 0.112	0.743 ± 0.116
31	0.696 ± 0.071	0.705 ± 0.064	0.741 ± 0.068	0.680 ± 0.090	0.682 ± 0.090
35	0.973 ± 0.019	0.976 ± 0.018	0.920 ± 0.022	0.968 ± 0.024	0.968 ± 0.030
36	0.854 ± 0.072	0.850 ± 0.071	0.927 ± 0.032	0.870 ± 0.168	0.870 ± 0.168
37	0.725 ± 0.086	0.720 ± 0.087	0.924 ± 0.026	0.922 ± 0.102	0.922 ± 0.226
38	0.667 ± 0.062	0.625 ± 0.070	0.574 ± 0.079	0.791 ± 0.040	0.791 ± 0.040
39	0.436 ± 0.064	0.573 ± 0.094	0.517 ± 0.068	0.516 ± 0.226	0.541 ± 0.226
40	0.768 ± 0.086	0.724 ± 0.086	0.446 ± 0.042	0.737 ± 0.107	0.800 ± 0.107
41	0.779 ± 0.088	0.792 ± 0.089	0.808 ± 0.132	0.896 ± 0.049	0.896 ± 0.058
42	0.827 ± 0.081	0.691 ± 0.159	0.667 ± 0.219	0.709 ± 0.187	0.730 ± 0.231
43	0.500 ± 0.000	0.000 ± 0.000	0.821 ± 0.002	0.821 ± 0.002	0.821 ± 0.044
44	0.500 ± 0.000	0.000 ± 0.000	0.679 ± 0.002	0.679 ± 0.002	0.960 ± 0.017
45	0.953 ± 0.033	0.973 ± 0.028	0.916 ± 0.031	0.965 ± 0.014	0.965 ± 0.025
46	1.000 ± 0.000	1.000 ± 0.000	0.925 ± 0.026	0.982 ± 0.005	0.982 ± 0.016
47	0.976 ± 0.017	0.978 ± 0.020	0.908 ± 0.020	0.963 ± 0.016	0.963 ± 0.020
48	0.239 ± 0.040	0.320 ± 0.052	0.440 ± 0.076	0.722 ± 0.021	0.722 ± 0.055
49	0.261 ± 0.034	0.314 ± 0.052	0.472 ± 0.069	0.738 ± 0.018	0.738 ± 0.059
50	0.696 ± 0.037	0.692 ± 0.052	0.605 ± 0.039	0.791 ± 0.080	0.791 ± 0.100
51	0.663 ± 0.032	0.664 ± 0.050	0.546 ± 0.059	0.769 ± 0.059	0.769 ± 0.104
52	0.241 ± 0.029	0.315 ± 0.044	0.441 ± 0.064	0.733 ± 0.012	0.733 ± 0.054
53	0.267 ± 0.032	0.338 ± 0.038	0.465 ± 0.070	0.734 ± 0.019	0.734 ± 0.059
54	0.768 ± 0.049	0.778 ± 0.049	0.736 ± 0.070	0.830 ± 0.046	0.830 ± 0.057
55	0.636 ± 0.055	0.638 ± 0.057	0.731 ± 0.102	0.705 ± 0.118	0.726 ± 0.118
63	0.783 ± 0.075	0.752 ± 0.098	0.676 ± 0.168	0.860 ± 0.051	0.860 ± 0.120
64	0.678 ± 0.109	0.579 ± 0.102	0.504 ± 0.124	0.697 ± 0.151	0.697 ± 0.158
65	0.721 ± 0.088	0.705 ± 0.090	0.508 ± 0.122	0.737 ± 0.160	0.737 ± 0.176
66	0.175 ± 0.028	0.677 ± 0.059	0.793 ± 0.092	0.826 ± 0.037	0.826 ± 0.077
67	0.227 ± 0.025	0.468 ± 0.036	0.402 ± 0.045	0.528 ± 0.055	0.528 ± 0.055
68	0.404 ± 0.033	0.544 ± 0.051	0.451 ± 0.022	0.488 ± 0.022	0.580 ± 0.161
80	0.996 ± 0.006	0.995 ± 0.006	0.860 ± 0.016	0.980 ± 0.008	0.980 ± 0.011
81	0.997 ± 0.007	0.997 ± 0.006	0.880 ± 0.012	0.984 ± 0.006	0.984 ± 0.011
82	0.876 ± 0.060	0.880 ± 0.060	0.853 ± 0.024	0.938 ± 0.035	0.938 ± 0.092

Table 8: Comparative Results: AUC obtained on D on Gauss, MoG, PDE, kNN, kM and kC. Problems from 1 to 41.

	Gauss	**MoG**	**PDE**	**kNN**	**kM**	**kC**
1	0.930 ± 0.018	0.960 ± 0.014	0.971 ± 0.011	0.968 ± 0.011	0.918 ± 0.020	0.880 ± 0.034
2	0.930 ± 0.027	0.953 ± 0.045	0.690 ± 0.057	0.723 ± 0.067	0.624 ± 0.086	0.653 ± 0.063
3	0.929 ± 0.018	0.963 ± 0.014	0.972 ± 0.011	0.970 ± 0.010	0.915 ± 0.023	0.883 ± 0.038
4	0.516 ± 0.042	0.595 ± 0.042	0.558 ± 0.041	0.562 ± 0.040	0.538 ± 0.053	0.582 ± 0.083
5	0.688 ± 0.046	0.642 ± 0.051	0.612 ± 0.050	0.558 ± 0.041	0.604 ± 0.054	0.663 ± 0.052
6	0.887 ± 0.086	0.900 ± 0.086	0.921 ± 0.081	0.924 ± 0.078	0.925 ± 0.077	0.922 ± 0.076
7	0.899 ± 0.110	0.875 ± 0.107	0.893 ± 0.094	0.894 ± 0.097	0.875 ± 0.109	0.894 ± 0.114
8	0.855 ± 0.102	0.841 ± 0.107	0.937 ± 0.077	0.937 ± 0.075	0.938 ± 0.074	0.929 ± 0.080
9	0.891 ± 0.076	0.793 ± 0.109	0.834 ± 0.085	0.813 ± 0.082	0.842 ± 0.086	0.912 ± 0.075
10	0.871 ± 0.069	0.851 ± 0.079	0.889 ± 0.065	0.885 ± 0.064	0.879 ± 0.078	0.885 ± 0.070
11	0.752 ± 0.068	0.780 ± 0.070	0.765 ± 0.076	0.780 ± 0.081	0.767 ± 0.092	0.794 ± 0.080
12	0.583 ± 0.047	0.608 ± 0.047	0.585 ± 0.049	0.606 ± 0.055	0.573 ± 0.052	0.562 ± 0.041
13	0.511 ± 0.051	0.522 ± 0.047	0.474 ± 0.045	0.511 ± 0.043	0.494 ± 0.051	0.518 ± 0.071
14	0.865 ± 0.017	0.893 ± 0.023	0.862 ± 0.015	0.854 ± 0.046	0.858 ± 0.022	0.738 ± 0.034
15	0.657 ± 0.033	0.873 ± 0.047	0.947 ± 0.008	0.950 ± 0.010	0.679 ± 0.033	0.545 ± 0.056
16	0.899 ± 0.014	0.955 ± 0.024	0.929 ± 0.016	0.929 ± 0.028	0.879 ± 0.033	0.807 ± 0.038
17	0.947 ± 0.012	0.965 ± 0.020	0.959 ± 0.011	0.957 ± 0.016	0.946 ± 0.022	0.884 ± 0.033
18	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.999 ± 0.002	1.000 ± 0.000
19	1.000 ± 0.000	1.000 ± 0.000	0.997 ± 0.004	0.997 ± 0.004	0.994 ± 0.006	0.996 ± 0.004
20	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
21	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
22	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
23	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.999 ± 0.002	1.000 ± 0.001
24	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.001	1.000 ± 0.001	0.991 ± 0.010	0.996 ± 0.006
25	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
26	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
27	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.999 ± 0.002	0.999 ± 0.002
28	0.625 ± 0.063	0.630 ± 0.077	0.720 ± 0.065	0.720 ± 0.066	0.702 ± 0.059	0.655 ± 0.072
29	0.653 ± 0.084	0.643 ± 0.090	0.674 ± 0.082	0.676 ± 0.082	0.666 ± 0.080	0.608 ± 0.111
30	0.626 ± 0.081	0.641 ± 0.095	0.721 ± 0.083	0.723 ± 0.083	0.694 ± 0.080	0.650 ± 0.084
31	0.654 ± 0.078	0.639 ± 0.079	0.678 ± 0.073	0.681 ± 0.072	0.662 ± 0.088	0.613 ± 0.106
32	0.407 ± 0.026	0.403 ± 0.033	0.421 ± 0.027	0.426 ± 0.027	0.442 ± 0.033	0.460 ± 0.045
33	0.661 ± 0.029	0.656 ± 0.038	0.661 ± 0.037	0.645 ± 0.037	0.644 ± 0.035	0.580 ± 0.042
34	0.635 ± 0.029	0.610 ± 0.032	0.592 ± 0.029	0.583 ± 0.029	0.611 ± 0.029	0.621 ± 0.037
35	0.977 ± 0.022	0.964 ± 0.028	0.971 ± 0.020	0.971 ± 0.020	0.964 ± 0.025	0.963 ± 0.027
36	0.873 ± 0.072	0.841 ± 0.077	0.798 ± 0.066	0.842 ± 0.072	0.797 ± 0.079	0.770 ± 0.079
37	0.926 ± 0.051	0.808 ± 0.089	0.676 ± 0.085	0.713 ± 0.088	0.663 ± 0.096	0.679 ± 0.152
38	0.606 ± 0.060	0.675 ± 0.067	0.669 ± 0.068	0.674 ± 0.062	0.637 ± 0.082	0.589 ± 0.096
39	0.510 ± 0.066	0.439 ± 0.076	0.433 ± 0.067	0.426 ± 0.060	0.432 ± 0.082	0.645 ± 0.090
40	0.602 ± 0.074	0.781 ± 0.098	0.720 ± 0.085	0.761 ± 0.086	0.702 ± 0.077	0.650 ± 0.078
41	0.801 ± 0.085	0.763 ± 0.101	0.775 ± 0.087	0.774 ± 0.089	0.761 ± 0.101	0.683 ± 0.084

Table 9: Comparative Results: AUC obtained on D on Gauss, MoG, PDE, kNN, kM and kC. Problems from 42 to 82.

	Gauss	**MoG**	**PDE**	**kNN**	**kM**	**kC**
42	0.763 ± 0.091	0.809 ± 0.132	0.802 ± 0.083	0.821 ± 0.075	0.796 ± 0.084	0.722 ± 0.107
43	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.385 ± 0.082
44	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.957 ± 0.038
45	0.973 ± 0.022	0.934 ± 0.044	0.941 ± 0.036	0.951 ± 0.034	0.937 ± 0.041	0.938 ± 0.042
46	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.998 ± 0.004	0.993 ± 0.012
47	0.990 ± 0.013	0.979 ± 0.023	0.975 ± 0.019	0.975 ± 0.017	0.971 ± 0.024	0.973 ± 0.022
48	0.486 ± 0.057	0.426 ± 0.054	0.206 ± 0.043	0.239 ± 0.040	0.412 ± 0.057	0.476 ± 0.056
49	0.470 ± 0.056	0.419 ± 0.046	0.230 ± 0.037	0.261 ± 0.033	0.410 ± 0.047	0.482 ± 0.048
50	0.626 ± 0.045	0.689 ± 0.047	0.696 ± 0.044	0.698 ± 0.037	0.678 ± 0.062	0.560 ± 0.056
51	0.518 ± 0.044	0.619 ± 0.047	0.675 ± 0.035	0.667 ± 0.031	0.608 ± 0.055	0.564 ± 0.053
52	0.460 ± 0.051	0.408 ± 0.063	0.215 ± 0.031	0.241 ± 0.029	0.396 ± 0.051	0.480 ± 0.046
53	0.502 ± 0.053	0.442 ± 0.051	0.249 ± 0.035	0.267 ± 0.032	0.428 ± 0.048	0.492 ± 0.038
54	0.798 ± 0.045	0.767 ± 0.068	0.783 ± 0.048	0.770 ± 0.050	0.768 ± 0.055	0.768 ± 0.067
55	0.666 ± 0.065	0.671 ± 0.079	0.653 ± 0.057	0.628 ± 0.058	0.635 ± 0.069	0.830 ± 0.102
56	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.965 ± 0.021
57	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.999 ± 0.001
58	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.810 ± 0.052
59	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.830 ± 0.050
60	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.853 ± 0.034
61	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.970 ± 0.012
62	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.993 ± 0.008
63	0.722 ± 0.071	0.770 ± 0.087	0.710 ± 0.083	0.782 ± 0.075	0.724 ± 0.085	0.692 ± 0.088
64	0.553 ± 0.103	0.612 ± 0.115	0.607 ± 0.105	0.678 ± 0.108	0.593 ± 0.103	0.541 ± 0.107
65	0.496 ± 0.097	0.668 ± 0.077	0.678 ± 0.095	0.720 ± 0.088	0.662 ± 0.094	0.651 ± 0.107
66	0.754 ± 0.038	0.651 ± 0.050	0.340 ± 0.036	0.742 ± 0.034	0.746 ± 0.055	0.571 ± 0.043
67	0.431 ± 0.026	0.543 ± 0.034	0.330 ± 0.022	0.558 ± 0.032	0.420 ± 0.034	0.530 ± 0.042
68	0.618 ± 0.037	0.650 ± 0.043	0.488 ± 0.038	0.669 ± 0.043	0.601 ± 0.056	0.577 ± 0.049
69	0.989 ± 0.006	0.999 ± 0.001	0.999 ± 0.002	0.999 ± 0.002	0.978 ± 0.015	0.967 ± 0.013
70	0.982 ± 0.008	0.999 ± 0.001	0.999 ± 0.002	0.999 ± 0.002	0.964 ± 0.014	0.960 ± 0.018
71	0.995 ± 0.005	0.999 ± 0.002	0.999 ± 0.001	0.999 ± 0.001	0.983 ± 0.009	0.974 ± 0.012
72	0.999 ± 0.001	0.999 ± 0.001	1.000 ± 0.001	1.000 ± 0.001	0.982 ± 0.016	0.972 ± 0.012
73	0.975 ± 0.011	0.997 ± 0.004	0.995 ± 0.004	0.995 ± 0.004	0.962 ± 0.020	0.950 ± 0.020
74	0.953 ± 0.017	0.991 ± 0.008	0.992 ± 0.006	0.992 ± 0.006	0.936 ± 0.024	0.915 ± 0.034
75	0.963 ± 0.013	0.995 ± 0.007	0.996 ± 0.004	0.996 ± 0.004	0.937 ± 0.032	0.910 ± 0.032
76	0.994 ± 0.004	0.999 ± 0.001	0.999 ± 0.001	0.999 ± 0.001	0.974 ± 0.018	0.971 ± 0.012
77	0.987 ± 0.010	0.989 ± 0.017	0.990 ± 0.014	0.990 ± 0.014	0.920 ± 0.033	0.855 ± 0.043
78	0.993 ± 0.006	0.999 ± 0.002	1.000 ± 0.001	1.000 ± 0.001	0.948 ± 0.040	0.939 ± 0.029
79	0.991 ± 0.006	0.998 ± 0.002	0.999 ± 0.002	0.999 ± 0.002	0.962 ± 0.029	0.970 ± 0.019
80	0.988 ± 0.013	0.981 ± 0.017	0.995 ± 0.006	0.995 ± 0.006	0.994 ± 0.009	0.992 ± 0.010
81	1.000 ± 0.000	1.000 ± 0.000	0.997 ± 0.006	0.997 ± 0.005	0.992 ± 0.011	0.990 ± 0.013
82	0.951 ± 0.033	0.927 ± 0.049	0.883 ± 0.059	0.870 ± 0.062	0.873 ± 0.052	0.872 ± 0.072

Table 10: Comparative Results: AUC obtained on D^* on MST, APE-1, APE-2 and N-APE. Problems from 1 to 41.

	MST	APE-ONE	APE-TWO	N-APE
1	0.901 ± 0.023	0.922 ± 0.048	0.933 ± 0.030	0.937 ± 0.030
2	0.598 ± 0.045	0.707 ± 0.207	0.951 ± 0.065	0.951 ± 0.065
3	0.903 ± 0.022	0.924 ± 0.039	0.935 ± 0.034	0.935 ± 0.034
4	0.561 ± 0.040	0.608 ± 0.097	0.828 ± 0.024	0.828 ± 0.041
5	0.551 ± 0.040	0.560 ± 0.061	0.548 ± 0.176	0.627 ± 0.186
6	0.927 ± 0.080	0.934 ± 0.013	0.867 ± 0.154	0.867 ± 0.188
7	0.899 ± 0.094	0.916 ± 0.025	0.836 ± 0.202	0.836 ± 0.236
8	0.944 ± 0.076	0.938 ± 0.012	0.808 ± 0.190	0.808 ± 0.241
9	0.833 ± 0.084	0.868 ± 0.099	0.843 ± 0.174	0.843 ± 0.174
10	0.878 ± 0.072	0.864 ± 0.141	0.768 ± 0.229	0.819 ± 0.229
11	0.769 ± 0.076	0.590 ± 0.183	0.581 ± 0.183	0.699 ± 0.224
12	0.608 ± 0.055	0.526 ± 0.068	0.619 ± 0.110	0.627 ± 0.116
13	0.514 ± 0.041	0.551 ± 0.064	0.667 ± 0.062	0.670 ± 0.062
14	0.853 ± 0.050	0.717 ± 0.188	0.853 ± 0.117	0.853 ± 0.143
15	0.832 ± 0.060	0.746 ± 0.035	0.778 ± 0.049	0.882 ± 0.049
16	0.928 ± 0.028	0.486 ± 0.002	0.971 ± 0.003	0.971 ± 0.153
17	0.956 ± 0.016	0.981 ± 0.000	0.981 ± 0.000	0.981 ± 0.034
18	1.000 ± 0.000	0.910 ± 0.000	0.910 ± 0.000	0.998 ± 0.001
19	0.997 ± 0.004	0.864 ± 0.001	0.864 ± 0.001	0.996 ± 0.005
20	1.000 ± 0.000	0.988 ± 0.001	0.988 ± 0.001	0.998 ± 0.002
21	1.000 ± 0.000	0.981 ± 0.000	0.981 ± 0.000	0.999 ± 0.000
22	1.000 ± 0.000	0.983 ± 0.001	0.983 ± 0.001	0.998 ± 0.001
23	1.000 ± 0.000	0.922 ± 0.000	0.922 ± 0.000	0.998 ± 0.002
24	1.000 ± 0.001	0.941 ± 0.001	0.941 ± 0.001	0.998 ± 0.001
25	1.000 ± 0.000	0.975 ± 0.001	0.975 ± 0.001	0.998 ± 0.001
26	1.000 ± 0.000	0.984 ± 0.000	0.984 ± 0.000	0.999 ± 0.001
27	1.000 ± 0.000	0.954 ± 0.000	0.954 ± 0.000	0.998 ± 0.002
28	0.719 ± 0.061	0.622 ± 0.148	0.723 ± 0.117	0.730 ± 0.117
29	0.690 ± 0.078	0.746 ± 0.069	0.674 ± 0.091	0.674 ± 0.091
30	0.724 ± 0.081	0.630 ± 0.119	0.743 ± 0.112	0.743 ± 0.116
31	0.696 ± 0.071	0.741 ± 0.068	0.680 ± 0.090	0.682 ± 0.090
32	0.429 ± 0.027	0.461 ± 0.051	0.557 ± 0.069	0.557 ± 0.069
33	0.646 ± 0.036	0.528 ± 0.106	0.550 ± 0.157	0.732 ± 0.170
34	0.586 ± 0.028	0.630 ± 0.155	0.709 ± 0.138	0.726 ± 0.138
35	0.973 ± 0.019	0.920 ± 0.022	0.968 ± 0.024	0.968 ± 0.030
36	0.854 ± 0.072	0.927 ± 0.032	0.870 ± 0.168	0.870 ± 0.168
37	0.725 ± 0.086	0.924 ± 0.026	0.922 ± 0.102	0.922 ± 0.226
38	0.667 ± 0.062	0.574 ± 0.079	0.791 ± 0.040	0.791 ± 0.040
39	0.436 ± 0.064	0.517 ± 0.068	0.516 ± 0.226	0.541 ± 0.226
40	0.768 ± 0.086	0.446 ± 0.042	0.737 ± 0.107	0.800 ± 0.107
41	0.779 ± 0.088	0.808 ± 0.132	0.896 ± 0.049	0.896 ± 0.058

Table 11: Comparative Results: AUC obtained on D^* on MST, APE-1, APE-2 and N-APE. Problems from 42 to 82.

	MST	APE-ONE	APE-TWO	N-APE
42	0.827 ± 0.081	0.667 ± 0.219	0.709 ± 0.187	0.730 ± 0.231
43	0.500 ± 0.000	0.821 ± 0.002	0.821 ± 0.002	0.821 ± 0.044
44	0.500 ± 0.000	0.679 ± 0.002	0.679 ± 0.002	0.960 ± 0.017
45	0.953 ± 0.033	0.916 ± 0.031	0.965 ± 0.014	0.965 ± 0.025
46	1.000 ± 0.000	0.925 ± 0.026	0.982 ± 0.005	0.982 ± 0.016
47	0.976 ± 0.017	0.908 ± 0.020	0.963 ± 0.016	0.963 ± 0.020
48	0.239 ± 0.040	0.440 ± 0.076	0.722 ± 0.021	0.722 ± 0.055
49	0.261 ± 0.034	0.472 ± 0.069	0.738 ± 0.018	0.738 ± 0.059
50	0.696 ± 0.037	0.605 ± 0.039	0.791 ± 0.080	0.791 ± 0.100
51	0.663 ± 0.032	0.546 ± 0.059	0.769 ± 0.059	0.769 ± 0.104
52	0.241 ± 0.029	0.441 ± 0.064	0.733 ± 0.012	0.733 ± 0.054
53	0.267 ± 0.032	0.465 ± 0.070	0.734 ± 0.019	0.734 ± 0.059
54	0.768 ± 0.049	0.736 ± 0.070	0.830 ± 0.046	0.830 ± 0.057
55	0.636 ± 0.055	0.731 ± 0.102	0.705 ± 0.118	0.726 ± 0.118
56	0.500 ± 0.000	0.929 ± 0.000	0.929 ± 0.000	0.957 ± 0.083
57	0.500 ± 0.000	0.929 ± 0.000	0.929 ± 0.000	0.982 ± 0.038
58	0.500 ± 0.000	0.929 ± 0.000	0.929 ± 0.000	0.929 ± 0.243
59	0.500 ± 0.000	0.929 ± 0.000	0.929 ± 0.000	0.932 ± 0.078
60	0.500 ± 0.000	0.929 ± 0.000	0.929 ± 0.000	0.929 ± 0.182
61	0.500 ± 0.000	0.929 ± 0.000	0.929 ± 0.000	0.953 ± 0.120
62	0.500 ± 0.000	0.929 ± 0.000	0.929 ± 0.000	0.929 ± 0.189
63	0.783 ± 0.075	0.676 ± 0.168	0.860 ± 0.051	0.860 ± 0.120
64	0.678 ± 0.109	0.504 ± 0.124	0.697 ± 0.151	0.697 ± 0.158
65	0.721 ± 0.088	0.508 ± 0.122	0.737 ± 0.160	0.737 ± 0.176
66	0.175 ± 0.028	0.793 ± 0.092	0.826 ± 0.037	0.826 ± 0.077
67	0.227 ± 0.025	0.402 ± 0.045	0.528 ± 0.055	0.528 ± 0.055
68	0.404 ± 0.033	0.451 ± 0.022	0.488 ± 0.022	0.580 ± 0.161
69	0.999 ± 0.002	0.986 ± 0.006	0.994 ± 0.004	0.994 ± 0.012
70	0.999 ± 0.002	0.983 ± 0.007	0.994 ± 0.003	0.994 ± 0.039
71	0.999 ± 0.001	0.985 ± 0.006	0.996 ± 0.001	0.996 ± 0.007
72	1.000 ± 0.001	0.988 ± 0.005	0.995 ± 0.003	0.995 ± 0.013
73	0.995 ± 0.004	0.986 ± 0.005	0.994 ± 0.003	0.994 ± 0.013
74	0.992 ± 0.006	0.983 ± 0.007	0.992 ± 0.003	0.992 ± 0.013
75	0.996 ± 0.004	0.974 ± 0.035	0.993 ± 0.004	0.993 ± 0.139
76	0.999 ± 0.001	0.973 ± 0.034	0.995 ± 0.002	0.995 ± 0.011
77	0.990 ± 0.014	0.971 ± 0.035	0.993 ± 0.003	0.993 ± 0.015
78	1.000 ± 0.001	0.989 ± 0.005	0.994 ± 0.004	0.994 ± 0.097
79	0.999 ± 0.002	0.976 ± 0.034	0.994 ± 0.003	0.994 ± 0.010
80	0.996 ± 0.006	0.860 ± 0.016	0.980 ± 0.008	0.980 ± 0.011
81	0.997 ± 0.007	0.880 ± 0.012	0.984 ± 0.006	0.984 ± 0.011
82	0.876 ± 0.060	0.853 ± 0.024	0.938 ± 0.035	0.938 ± 0.092

Table 12: Comparative Results: User Verification from Walkin Patterns on Gauss, MoG, PDE, kNN, kM and kC.

	Gauss	MoG	PDE	kNN	kM	kC
1	0.964 ± 0.006	0.966 ± 0.008	0.974 ± 0.004	0.978 ± 0.012	0.984 ± 0.001	0.978 ± 0.001
2	0.863 ± 0.012	0.938 ± 0.006	0.939 ± 0.005	0.951 ± 0.014	0.960 ± 0.004	0.948 ± 0.005
3	0.928 ± 0.009	0.966 ± 0.005	0.975 ± 0.006	0.976 ± 0.007	0.976 ± 0.007	0.975 ± 0.006
4	0.965 ± 0.006	0.981 ± 0.005	0.980 ± 0.004	0.979 ± 0.004	0.982 ± 0.004	0.982 ± 0.005
5	0.889 ± 0.015	0.916 ± 0.015	0.926 ± 0.010	0.923 ± 0.018	0.936 ± 0.006	0.936 ± 0.012
6	0.909 ± 0.011	0.954 ± 0.013	0.965 ± 0.007	0.959 ± 0.008	0.975 ± 0.004	0.970 ± 0.005
7	0.841 ± 0.016	0.863 ± 0.014	0.874 ± 0.012	0.867 ± 0.023	0.871 ± 0.013	0.875 ± 0.012
8	0.813 ± 0.012	0.955 ± 0.004	0.958 ± 0.003	0.944 ± 0.010	0.961 ± 0.004	0.962 ± 0.004
9	0.981 ± 0.004	0.985 ± 0.003	0.987 ± 0.002	0.987 ± 0.005	0.988 ± 0.002	0.988 ± 0.002
10	0.937 ± 0.007	0.982 ± 0.004	0.983 ± 0.003	0.983 ± 0.005	0.986 ± 0.004	0.984 ± 0.004
11	0.902 ± 0.012	0.931 ± 0.003	0.932 ± 0.004	0.934 ± 0.008	0.937 ± 0.005	0.935 ± 0.006
12	0.506 ± 0.022	0.760 ± 0.022	0.781 ± 0.021	0.842 ± 0.011	0.830 ± 0.012	0.790 ± 0.016
13	0.812 ± 0.027	0.863 ± 0.028	0.877 ± 0.027	0.889 ± 0.025	0.892 ± 0.024	0.877 ± 0.025
14	0.803 ± 0.024	0.824 ± 0.016	0.840 ± 0.018	0.846 ± 0.012	0.846 ± 0.013	0.839 ± 0.016
15	0.869 ± 0.019	0.883 ± 0.014	0.886 ± 0.008	0.924 ± 0.013	0.913 ± 0.010	0.896 ± 0.007

Table 13: Comparative Results: User Verification from Walkin Patterns on MST, SVDD, APE-1, APE-2 and N-APE.

	MST	SVDD	APE-ONE	APE-TWO	N-APE
1	0.966 ± 0.006	0.972 ± 0.004	0.967 ± 0.011	0.992 ± 0.003	0.996 ± 0.141
2	0.928 ± 0.004	0.942 ± 0.006	0.938 ± 0.008	0.962 ± 0.009	0.982 ± 0.011
3	0.971 ± 0.007	0.975 ± 0.005	0.966 ± 0.008	0.985 ± 0.003	0.995 ± 0.012
4	0.980 ± 0.004	0.982 ± 0.004	0.971 ± 0.007	0.994 ± 0.001	0.997 ± 0.051
5	0.914 ± 0.009	0.920 ± 0.007	0.893 ± 0.077	0.947 ± 0.009	0.979 ± 0.103
6	0.953 ± 0.007	0.967 ± 0.007	0.805 ± 0.145	0.941 ± 0.017	0.979 ± 0.026
7	0.854 ± 0.015	0.878 ± 0.010	0.971 ± 0.005	0.998 ± 0.000	0.999 ± 0.033
8	0.944 ± 0.006	0.957 ± 0.003	0.903 ± 0.066	0.837 ± 0.143	0.982 ± 0.155
9	0.985 ± 0.002	0.988 ± 0.002	0.960 ± 0.007	0.991 ± 0.003	0.998 ± 0.005
10	0.980 ± 0.003	0.982 ± 0.004	0.968 ± 0.007	0.980 ± 0.022	0.994 ± 0.170
11	0.929 ± 0.005	0.934 ± 0.007	0.965 ± 0.007	0.984 ± 0.003	0.994 ± 0.150
12	0.765 ± 0.017	0.780 ± 0.023	0.763 ± 0.099	0.699 ± 0.136	0.907 ± 0.034
13	0.857 ± 0.021	0.876 ± 0.018	0.969 ± 0.010	0.866 ± 0.173	0.993 ± 0.181
14	0.833 ± 0.018	0.850 ± 0.018	0.914 ± 0.010	0.939 ± 0.007	0.975 ± 0.081
15	0.876 ± 0.010	0.886 ± 0.010	0.834 ± 0.118	0.884 ± 0.055	0.970 ± 0.097

Table 14: Comparative Results: Training Time computed in seconds

Gauss	0.0348 ± 0.0918
MoG	0.2851 ± 0.0968
PDE	0.2633 ± 0.1669
kNN	0.1685 ± 0.0324
kM	0.0184 ± 0.0067
kC	0.0778 ± 0.0141
MST	0.3129 ± 0.0199
SVDD	84.7442 ± 4.8212
APE-1	0.0022 ± 0.0059
APE-2	0.1043 ± 0.0341
N-APE	4.9872 ± 4.9610

Table 15: Comparative Results: Testing Time computed in seconds

Gauss	0.0156 ± 0.0320
MoG	0.0104 ± 0.0108
PDE	0.0231 ± 0.0072
kNN	0.0151 ± 0.0076
kM	0.0064 ± 0.0065
kC	0.0260 ± 0.0136
MST	0.0625 ± 0.0094
SVDD	0.0208 ± 0.0067
APE-1	0.0002 ± 0.0002
APE-2	0.0026 ± 0.0002
N-APE	0.0768 ± 0.0752

List of Pubblications

.1 Book Chapters

P. Casale, O. Pujol, and P. Radeva, **Embedding Random Projections in Regularized Gradient Boosting Machines**, in: Ensemble in Machine Learning Applications, Studies in Computational Intelligence, Volume 373, pp:201-217, 2011, DOI:10.1007/978-3-642-22910-7, Springer. URL: `http://www.springerlink.com/content/451515p006h5501r/`

.2 Journal Papers

P. Casale, O. Pujol, and P. Radeva, **Approximate Polytope Ensemble for One-Class Classification**, submitted to: Patterns Analysis and Machine Intelligence, IEEE Transactions on, November 2011

P. Casale, O. Pujol, and P. Radeva, **Personalization and User Verification in Wearable Systems using Biometric Walking Pattern**, in: Personal and Ubiquitous Computing, in press, 2011, DOI:10.1007/978-3-642-22910-7, Springer. URL: `https://springerlink3.metapress.com/content/wx41085567168485/resource-secured/?target=fulltext.pdf&sid=c5h12wa5o2j1yolflne4b4o2\&sh=www.springerlink.com`

.3 Lecture Notes in Computer Science

P. Casale, O. Pujol, and P. Radeva, **Approximate Convex Hulls Family for One-Class Classification**, in: Proceedings of 10th International Workshop on Multiple Classifier Systems, Vol. 6713, pp:106–115, 2011, ISBN 978-3-642-21556-8, Springer. Preview URL: `http://books.google.com/books/about/Multiple_Classifier_Systems.html?id=lNQtKQEACAAJ`

P. Casale, O. Pujol, and P. Radeva, **Human Activity Recognition from Accelerometer Data using a Wearable Device**, in: Proceedings of Pattern Recognition and Image Analysis - 5th Iberian Conference, IbPRIA 2011,, Vol. 6669, pp:289-296 2011, DOI: $1007/978 - 3 - 642 - 21257 - 4_36$, Springer URL:`http://dx.doi.org/10.1007/978-3-642-21257-4_36`

P. Casale, O. Pujol, and P. Radeva, **Face-to-face social activity detection using data collected with a wearable device**, in: Proceedings of Pattern Recognition and Image Analysis - 4th Iberian Conference, IbPRIA 2009,, Vol. 5524/2009, pp:56-63 2009, DOI: $110.1007/978 - 3 - 642 - 02172 - 5_9$, Springer URL:`http://www.springerlink.com/content/9740t48602gw17t4/`

.4 International Conference and Workshops

P. Casale, O. Pujol, and P. Radeva, **User Verification From Walking Activity. First Steps towards a Personal Verification System**, in: Proceedings of 1st International Conference On Pervasive and Embedded Computing and Communication Systems, PECCS2011, Vol. 1, pp:78-86 20011, ISBN: 978-989-8425-48-5, ISTICC

P. Casale, O. Pujol, and P. Radeva, **Embedding Random Projections in Regularized Gradient Boosting Machines**, in: Workshop on Supervised and Unsupervised Methods and their Applications(SUEMA), pp:44-54 2010, European Conference on Machine Learning, ECML-PKDD 2010, URL:`http://eprints.pascal-network.org/archive/00007717/01/SUEMA10_proceedings.pdf`

.5 Local Conference and Workshops

P. Casale, O. Pujol, and P. Radeva, **BeaStreamer-v0.1: a new platform for Multi-Sensors Data Acquisition in Wearable Computing Applications**, in: Fourth CVC Workshop on the Progress of Research & Development CVCRD, 2009. ISBN: 978-84-937261-1-9 (Best Application Paper Award)

P. Casale, O. Pujol, P. Radeva, and J. Vitrià, **A first approach to Activity Recognition using Topic Models**, in: Proceedings of the 12th Catalan Conference on Artificial Intelligence Research and Development, CCIA 2009, ISBN 978-1-60750-061-2.

P. Casale, O. Pujol, and P. Radeva, **Social Environment Description from Data Collected with a Wearable Device**, in: Current Challenges in Computer Vision. Proceedings of CVC Internal Workshop. CVCRD, 2008, ISBN: 978-84-935251-9-4.

P. Casale, O. Pujol and P. Radeva, **Classifying Agitation in Sedated ICU Patients**, In: MICCAT Workshop, Girona, 2010.

List of Technician/BsC/MsC Thesis Supervised

.6 Technician Thesis in Informatic Engineering

Muñoz Gomez, Miguel, "Assembling and Testing of a Multisensorial Wearable Device for Biometric Signals Monitoring in Uncontrolled Environments", University of Barcelona, Director: Oriol Pujol, Tutor: Pierluigi Casale, 2010, `http://cataleg.ub.edu:2082/record=b1974303~S1*cat`

.7 BsC Thesis in Informatic Engineering

Ganzalez Rivera, Eduardo, "Physical Activity Recognition Using an Android Phone", University of Barcelona, Director: Santi Segui, Tutor: Pierluigi Casale, 2011

Muñoz Gomez, Miguel, "Gesture Recognition Using IMUs sensors: New perspective in HCI", University of Barcelona, Director: Oriol Pujol, Tutor: Pierluigi Casale, 2011

Trigo Chavez, David, "Pedestrian detector on devices working with limited resources", University of Barcelona, Director: Oriol Pujol, Tutor: Pierluigi Casale, 2011

Ruiz Fernandez, Raul, "Augmented Reality on BeaStreamer", University of Barcelona, Director: Oriol Pujol, Tutor: Pierluigi Casale, 2011

Orihuela Salvaterra, Helena, "Towards the optimal task-manager on BeaStreamer", University of Barcelona, Director: Oriol Pujol, Tutor: Pierluigi Casale, 2011

Lascorz Mauriz, Gerard, “Preliminar Study on Augmented Reality based on Beagleboard”, University of Barcelona, Director: Oriol Pujol, Tutor: Pierluigi Casale, 2010, `http://cataleg.ub.edu:2082/record=b1982475~S1*cat`

Esteban Soto, Angel, “Face Detection and Recognition on Beagleboard”, University of Barcelona, Director: Petia Radeva, Tutor: Pierluigi Casale, 2010, `http://cataleg.ub.edu:2082/record=b1981543~S1*cat`

.8 MsC Thesis in Electronics Engineering

De Santis, Massimiliano, “Computational optimization methods in Audio Analysis for OMAP-3530 based embedded devices”, University of Rome ”la Sapienza”, Rome, Italy, Director: Marco Balsi, Tutor: Pierluigi Casale, 2011

Salvetti, Luca, “Computational optimization methods in Image Analysis for OMAP-3530 based embedded devices”, University of Rome ”la Sapienza”, Rome, Italy, Director: Marco Balsi, Tutor: Pierluigi Casale, 2010

Bibliography

[AAliance(2010)] European AAL Innovation AAliance. Ambien assisted living roadmap, version 2, 2010. URL `http://www.aaliance.eu/public/documents/aaliance-roadmap/aaliance-aal-roadmap.pdf`.

[Achlioptas(2001)] Dimitris Achlioptas. Database-friendly random projections. In *20th Symposium on Principles of Database Systems. Proceedings of*, pages 274–281. ACM, 2001.

[Allen et al.(2006)] Felicity R Allen, Eliathamby Ambikairajah, Nigel H Lovell, and Branko G Celler. Classification of a known sequence of motions and postures from accelerometry data using adapted gaussian mixture models. *Physiological Measurement*, 27(10), 2006.

[Amft and Lukowicz(2009)] Oliver Amft and Paul Lukowicz. From backpacks to smartphones: Past, present and future of wearable computers. *IEEE Pervasive Computing*, 8(3):8–13, 2009.

[Android(2007)] Android, 2007. URL `http://www.android.com/`.

[Arduino(2008)] Arduino, 2008. URL `http://www.arduino.cc`.

[Arriaga and Vempala(2006)] Rosa I. Arriaga and Santosh Vempala. An algorithmic theory of learning: Robust concepts and random projection. *Machine Learning*, 63:161–182, 2006.

[Augusto(2007)] Juan Carlos Augusto. Ambient intelligence: the confluence of ubiquitous/pervasive computing and artificial intelligence. *Intelligent Computing Everywhere*, pages 213–234, 2007.

[Balcan et al.(2006)] Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. Kernels as features: On kernels, margins, and low-dimensional mappings. *Machine Learning*, 65(1):79–94, 2006.

[Bao and Intille(2004)] L. Bao and S. S. Intille. Activity Recognition from User-Annotated Acceleration Data. *IEEE Pervasive Computing*, pages 1–17, 2004.

[Beagleboard(2008)] Beagleboard, 2008. URL `http://www.beagleaboard.org/`.

[Beagleboard(2011)] Projects Beagleboard, 2011. URL `http://beagleboard.org/project`.

[Bell and Gemmel(2009)] G. Bell and J. Gemmel. *Total Recall: How the E-Memory Revolution will change Everything*. Penguin Group, 2009.

[Bennett and Bredensteiner(2000)] Kristin P. Bennett and Erin J. Bredensteiner. Duality and Geometry in SVM Classifiers. In *17th International Conference on Machine Learning, ICML00. Proceedings of*, pages 57–64. Morgan Kaufmann Publishers Inc., 2000.

[Bhattacharya(1982)] B. K. Bhattacharya. *Application of computational geometry to pattern recognition problems*. PhD thesis, 1982.

[Bi and Bennett(2001)] J. Bi and K. P. Bennett. Duality, geometry, and support vector regression. In *Advances in Neural Information Processing Systems 14,NIPS 2001. Proceedings of*, pages 593–600, 2001.

[Bianchi et al.(1998)] L. Bianchi, D. Angelini, and F. Lacquaniti. Individual characteristics of human walking mechanics. *Pflügers Archive European Journal of Physiology*, 436:343–356, 1998.

[Bingham and Mannila(2001)] Ella Bingham and Heikki Mannila. Random projection in dimensionality reduction: applications to image and text data. In *7th International Conference on Knowledge Discovery and Data Mining, KDD01. Proceedings of*, pages 245–250. ACM, 2001.

[Bishop(1995)] C.M. Bishop. *Neural networks for pattern recognition*. Oxford University Press, USA, 1995.

[Blum(2006)] Avrim Blum. Random Projection, Margins, Kernels, and Feature-Selection. pages 52–68. 2006.

[Breiman(1996)] Leo Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

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

[Brieman(1984)] Leo Brieman. *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.

[Choudhury et al.(2008)] Tanzeem Choudhury, Gaetano Borriello, Sunny Consolvo, Dirk Haehnel, Beverly Harrison, Bruce Hemingway, Jeffrey Hightower, Predrag "Pedja" Klasnja, Karl Koscher, Anthony LaMarca, James A. Landay, Louis LeGrand, Jonathan Lester, Ali Rahimi, Adam Rea, and Danny Wyatt. The mobile sensing platform: An embedded activity recognition system. *IEEE Pervasive Computing*, 7:32–41, 2008.

[Clarkson and Pentland(1999)] B. Clarkson and A. Pentland. Unsupervised clustering of ambulatory audio and video. In *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP99. Proceedings of*, pages 3037–3040, 1999.

[Dasgupta(2000)] Sanjoy Dasgupta. Experiments with random projection. In *Conference in Uncertainty in Artificial Intelligence, UAI00. Proceedings of*, pages 143–151. Morgan Kaufmann, 2000.

[Demšar(2006)] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal Machine Learning Research*, 7:1–30, December 2006.

[Derawi et al.(2010)] Mohammad Omar Derawi, Claudia Nickel, Patrick Bours, and Christoph Busch. Unobtrusive user-authentication on mobile phones using biometric gait recognition. *IEEE International Conference on Intelligent Information Hiding and Multimedia Signal Processing. Proceedings of*, pages 306–311, 2010.

[Dietterich and Bakiri(1991)] T. G. Dietterich and G. Bakiri. Error-correcting output codes: a general method for improving multiclass inductive learning programs. In T. L. Dean and K. Mckeown, editors, *9th AAAI National Conference on Artificial Intelligence. Proceedings of*, pages 572–577. AAAI Press, 1991.

[Dietterich(2002)] Thomas G. Dietterich. Machine learning for sequential data: A review. In *Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition. Proceedings of*, pages 15–30, 2002.

[Duda and Hart(1973)] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, 1973.

[European Commision(2001)] Community Research European Commision. Scenarios for ambient intelligence in 2010, 2001. URL `ftp://ftp.cordis.lu/pub/ist/docs/istagscenarios2010.pdf`.

[Eurotech(2008)] Eurotech. Zypad, 2008. URL `http://www.zypad.com/zypad/wearablecomputers`.

[Fawcett(2010)] T. Fawcett. Comparing patterns classifiers, 2010. URL `http://home.comcast.net/~tom.fawcett/public_html/ML-gallery/pages/index.html`.

[Fradkin and Madigan(2003)] Dmitriy Fradkin and David Madigan. Experiments with random projections for machine learning. In *9th International Conference on Knowledge Discovery and Data Mining, KDD03. Proceedings of*, pages 517–522. ACM, 2003.

[Frank and Asuncion(2010)] A. Frank and A. Asuncion. UCI machine learning repository, 2010. URL `http://archive.ics.uci.edu/ml`.

[Freund and Schapire(1999)] Yoav Freund and Robert E. Schapire. A short introduction to boosting. In *International Joint Conference on Artificial Intelligence, IJCAI99. Proceedings of*, pages 1401–1406, 1999.

[Friedman(2000)] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.

[Gafurov et al.(2006)] Davrondzhon Gafurov, Einar Snekkenes, and Tor Buvarp. Robustness of biometric gait authentication against impersonation attack. In Robert Meersman, Zahir Tari, and Pilar Herrero, editors, *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops*, volume 4277 of *Lecture Notes in Computer Science*, pages 479–488. Springer Berlin / Heidelberg, 2006.

[GNU(1989)] GNU. Gnu general public license, 1989. URL `http://www.gnu.org/copyleft/gpl.html`.

[GStreamer(1999)] GStreamer, 1999. URL `http://gstreamer.freedesktop.org/`.

[Gumstix(2011)] Gumstix, 2011. URL `http://www.gumstix.com/`.

[Hewitt(2002)] P. S. Hewitt. Depopulation and ageing in europe and japan: The hazardous transition to a labor shortage economy, 2002.

[ISO(2010)] ISO. Iso/iec 18092:2004 information technology – telecommunications and information exchange between systems – near field communication – interface and protocol (nfcip-1), 2010. URL `http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=38578`.

[Johnson and Lindenstauss(1984)] W. Johnson and J. Lindenstauss. Extensions of lipschitz maps into a hilbert space. *Contemporary Mathematics*, 1984.

[Juszczak et al.(2009)] Piotr Juszczak, David M. J. Tax, and Robert P. W. Duin. Minimum spanning tree based one-class classifier. *Neurocomputing*, 72:1859–1869, 2009.

[Karantonis et al.(2006)] D. M. Karantonis, M. R. Narayanan, M. Mathie, N. H. Lovell, and B. G. Celler. Implementation of a Real-Time Human Movement Classifier Using a Triaxial Accelerometer for Ambulatory Monitoring. *Information Technology in Biomedicine, IEEE Transactions on*, 10(1):156–167, 2006.

[Koppel and Schler(2004)] Moshe Koppel and Jonathan Schler. Authorship verification as a one-class classification problem. In *21st International Conference on Machine learning, ICML04. Proceedings of*, pages 62+. ACM, 2004.

[Kulikowski et al.(1999)] Casimir Kulikowski, Nathalie Japkowicz, Nathalie Japkowicz, Nathalie Japkowicz, Dissertation Directors, Stephen J. Hanson, and Stephen J. Hanson. Concept-Learning In The Absence Of Counter-Examples: An Autoassociation-Based Approach To Classification. In *International Joint Conference on Artificial Intelligence, IJCAI-95. Proceedings of*, pages 518–523, 1999.

[Kuncheva(2004)] Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, 2004.

[Lee et al.(2003)] S. H. Lee, H. D. Park, S. Y. Hong, K. J. Lee, and Y. H. Kim. volume 3, pages 2941–2943, 2003.

[Lester et al.(2006)] Jonathan Lester, Tanzeem Choudhury, and Gaetano Borriello. A Practical Approach to Recognizing Physical Activities. volume 3968 of *Lecture Notes in Computer Science*, pages 1–16. Springer Berlin / Heidelberg, 2006.

[Lin et al.(2011)] Zongfang Lin, Allen R. Hanson, Leon J. Osterweil, and Alexander Wise. Precise process definitions for activities of daily living: a basis for real-time monitoring and hazard detection. In *Workshop on Software Engineering in Health Care, SEHC11. Proceedings of*, pages 13–16, 2011.

[Lukowicz et al.(2001)] Paul Lukowicz, Urs Anliker, Gerhard Tröster, Steven J. Schwartz, and Richard W. DeVaul. The weararm modular, low-power computing core. *IEEE Micro*, 21:16–28, May 2001.

[Mannini and Sabatini(2010)] Andrea Mannini and Angelo Maria Sabatini. Machine learning methods for classifying human physical activity from on-body accelerometers. *Sensors*, 10(2):1154–1175, 2010.

[Mäntyjärvi et al.(2005)] Jani Mäntyjärvi, Mikko Lindholm, Elena Vildjiounaite, Satu marja Mäkelä, and Heikki Ailisto. Identifying users of portable devices from gait pattern with accelerometers. In *IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings of*, pages 973–976, 2005.

[Mavroforakis and Theodoridis(2006)] Michael E. Mavroforakis and Sergios Theodoridis. A geometric approach to support vector machine (svm) classification. *Neural Networks, IEEE Transactions on*, 17(3):671–682, 2006.

[Murray(1924)] A.T. Murray. Homer, the iliad, 1924.

[Nokia(2005)] Nokia. N770, 2005. URL `http://europe.nokia.com/support/product-support/nokia-770`.

[OpenCV(2007)] OpenCV, 2007. URL `http://opencv.willowgarage.com/`.

[OpenEmbedded(2008)] OpenEmbedded, 2008. URL `http://www.openembedded.org/`.

[OSHW(2010)] OSHW. Oepn source hardware, 2010. URL `http://freedomdefined.org/OSHW`.

[OSI(1998)] OSI. Open source iniciative, 1998. URL `http://www.opensource.org/licenses/mit-license.php`.

[Pal and Bhattacharya(2007)] S. Pal and S. Bhattacharya. *Neural Networks, IEEE Transactions on*, 18(2):600–605, 2007.

[Parzen(1962)] Emanuel Parzen. On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, 1962.

[Piekarski and Thomas(2009)] W. Piekarski and B.H. Thomas. Through-walls collaboration. *IEEE Pervasive Computing*, 8(3):42–49, 2009. URL `www.tinmith.net`.

[Preparata and Shamos(1985)] Franco P. Preparata and Michael I. Shamos. *Computational geometry: an introduction.* Springer-Verlag New York, Inc., 1985.

[Pujol(2010)] Oriol Pujol. Boosted geometry-based ensembles. In *International Workshop on Multiple Classifier Systems, MCS2010. Proceedings of*, volume 5997 of *Lecture Notes in Computer Science*, pages 195–204. Springer, 2010.

[Rahimi and Recht(2008)] Ali Rahimi and Benjamin Recht. Weighted Sums of Random Kitchen Sinks: Replacing minimization with randomization in learning. In *Advances in Neural Information Processing Systems,NIPS 2008. Proceedings of*, pages 1313–1320. MIT Press, 2008.

[Randell and Muller(2000)] C. Randell and H. Muller. Context awareness by analysing accelerometer data. pages 175–176, 2000.

[Ravi et al.(2005)] Nishkam Ravi, Nikhil Dandekar, Prreetham Mysore, and Michael L. Littman. Activity recognition from accelerometer data. *American Association for Artificial Intelligence*, pages 1541–1456, 2005.

[Sekine et al.(2002)] M. Sekine, T. Tamura, M. Akay, T. Fujimoto, T. Togawa, and Y. Fukui. Discrimination of walking patterns using wavelet-based fractal analysis. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 10(3): 188–196, 2002.

[Sharp(2001)] Sharp. Zaurus, 2001. URL `http://www.ezaurus.com/`.

[Shimmer(2008)] Research Shimmer, 2008. URL `http://www.shimmer-research.com/`.

[Song and Wang(2005)] Kai-Tai Song and Yao-Qing Wang. Remote activity monitoring of the elderly using a two-axis accelerometer. In *CACS Automatic Control Conference. Proceedings of*, pages 18–19, 2005.

[Spriggs et al.(2009)] Ekaterina H. Spriggs, Fernando De la Torre, and Martial Hebert. Temporal segmentation and activity classification from first-person sensing. In *IEEE Workshop on Egocentric Vision, CVPR 2009*, June 2009.

[Sprott(2003)] Julien Clinton Sprott. *Chaos and Time-Series Analysis.* Oxford University Press, 2003.

[Starner(1993)] Thad Starner. Lizzy: Mit's wearable computer design, 1993. URL `www.media.mit.edu/wearables/lizzy/lizzy/index.html`.

[Starner(1999)] Thad Starner. *Wearable Computing and Contextual Awarenesss.* PhD thesis, 1999.

[Takahashi and Kudo(2010)] Tetsuji Takahashi and Mineichi Kudo. Margin preserved approximate convex hulls for classification. pages 4052–4055, 2010.

[Tanenbaum and Goodman(1998)] Andrew S. Tanenbaum and James R. Goodman. *Structured Computer Organization.* Prentice Hall PTR, 4th edition, 1998.

[Tax(2001)] D. M. J. Tax. *One-class classification*. PhD thesis, 2001.

[Tax and Duin(2001)] David M.J. Tax and Robert P.W. Duin. Uniform object generation for optimizing one-class classifiers. *Journal of Machine Learning Research*, 2:155–173, 2001.

[Terrier and Schutz(2003)] Philippe Terrier and Yves Schutz. Variability of gait patterns during unconstrained walking assessed by satellite positioning (gps). *European Journal of Applied Physiology*, 90:554–561, 2003.

[TexasInstruments(2010)] TexasInstruments, 2010. URL `http://www.ti.com/tool/c6run-dsparmtool?DCMP=dsp-omapl132-110819&HQS=dsp-omapl132-pr-pf5`.

[Thorp(1998)] Edward O. Thorp. The invention of the first wearable computer. In *IEEE International Symposium on Wearable Computers, ISWC98. Proceedings of*, pages 4–, 1998.

[Thrun et al.(1991)] S. Thrun, J. Bala, E. Bloedorn, I. Bratko, B. Cestnik, J. Cheng, K. De Jong, S. Dzeroski, R. Hamann, K. Kaufman, S. Keller, I. Kononenko, J. Kreuziger, R.S. Michalski, T. Mitchell, P. Pachowicz, B. Roger, H. Vafaie, W. Van de Velde, W. Wenzel, J. Wnek, and J. Zhang. The MONK's problems: A performance comparison of different learning algorithms. (CMU-CS-91-197), 1991.

[Toussaint(1978)] G. T. Toussaint. The convex hull as a tool in pattern recognition. In *AFOSR Workshop in Communication Theory and Applications*, 1978.

[Turk and Pentland(1991)] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. In *IEEE International Conference on Computer Vision and Pattern Recognition, CVPR91. Proceedings of*, pages 586–591, 1991.

[Vapnik(1995)] Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., 1995.

[Vempala(2004)] S. Vempala. *The Random Projection Method*. American Mathematical Society, 2004.

[Vildjiounaite et al.(2007)] Elena Vildjiounaite, Satu-Marja Makela, Mikko Lindholm, Vesa Kyllonen, and Heikki Ailisto. Increasing Security of Mobile Devices by Decreasing User Effort in Verification. page 80, 2007.

[Vuzix(2006)] Vuzix, 2006. URL `http://www.vuzix.com/`.

[Weiser(1991)] Mark Weiser. The computer for the 21st century. *Scientific American*, 265(3):66–75, 1991.

[Xybernaut(1990)] Corp. Xybernaut. Xibernaut, 1990. URL `www.xybernaut.com`.

[Ypma et al.(1999)] A. Ypma, D. M. J. Tax, and R. P. W. Duin. Robust machine fault detection with independent component analysis and support vector data description. In *Neural Networks for Signal Processing IX. Proceedings of the 1999 IEEE Signal Processing Society Workshop*, pages 67–76, 1999.