Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Curriculum in Ingegneria Elettronica, Elettrotecnica e delle Telecomunicazioni

Ambient Intelligence: Computational Audio Processing For Human Fall Detection

Ph.D. Dissertation of:
Diego Droghini

Advisor:
Prof. Francesco Piazza

Coadvisor:
Prof. Roberto Bedini

Curriculum Supervisor:
Prof. Francesco Piazza

XVII edition - new series

Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Curriculum in Ingegneria Elettronica, Elettrotecnica e delle Telecomunicazioni

Ambient Intelligence: Computational Audio Processing For Human Fall Detection

Ph.D. Dissertation of:
Diego Droghini

Advisor:
Prof. Francesco Piazza

Coadvisor:
Prof. Roberto Bedini

Curriculum Supervisor:
Prof. Francesco Piazza

XVII edition - new series

Università Politecnica delle Marche
Scuola di Dottorato di Ricerca in Scienze dell'Ingegneria
Facoltà di Ingegneria
Via Brecce Bianche – 60131 Ancona (AN), Italy

alla mia famiglia

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Outline, obiettivi, controbuti

The decreasing birth rate [1] and the contemporary increase of the life expectancy at birth [2] in the majority of industrialized countries have been generating new challenges in the assistance of the elderly. The scientific community, companies and governments are trying to face them by investing in the development of efficient healthcare systems and solutions. The direction taken goes towards the development of smart home capable of taking care of the inhabitants by supporting and monitoring them in their daily actions [3, 4]. Since falls are one of the main cause of death for the elderly [5], several efforts have been devoted to the development of algorithms for automatically detecting these events.

1.1 Fall Detection Systems

The continuous and unprecedented growth rate of the elderly world population is one of the primary aspects of concern for society and governments. Nowadays about 8.5% of people in the world are more than 65 years old [6, 2]. Although the average life of the world population is getting longer, elderly people may not necessarily live a healthier life. It is enough to say that 37.5 million falls require medical interventions and more than 600 thousand are cause of death every year worldwide. In particular, the population segment most affected by this problem is composed of elderly over 65 years that, with the growing mobility of the population, are more frequently left alone in their homes without aid in the case of need. Moreover, since falls are the leading cause of death and hospitalizations for older adults, this phenomenon leads to a substantial increase in the cost of healthcare [7, 5]. It is not surprising, thus, that the research community is encouraged, even by governments, to find reliable and performing solutions to minimize the damage caused by the human falls problem. This is also confirmed by the presence in the literature of several reviews dedicated to this specific topic [5, 8, 9, 10, 11, 12]. In fact, in the past few years, a variety of systems have been presented. One way to divide the methodologies for ap-

proaching the falls detection problem is based on the placement of the sensing devices [5]. The main categories are wearable, vision and environmental, with each category presenting their own advantages and disadvantages. Wearable systems do not suffer from ambient condition, but people may forget to wear them and they are not operational during the charging time, thus, some people may consider them annoying. Furthermore, a device must be installed on each person to be monitored. An environmental sensor may be used to avoid this kind of problems, but with other limitations. Vision systems, although they are actually environmental sensors, deserve a dedicated category because of many systems proposed in the literature based on this type of sensors [5]. This category includes several types of sensors like, e.g., cameras for which the major limitations are field-of-view constraints, lighting condition, positioning of multiple cameras and lack of privacy. The ambient category includes several types of sensors. For example, radar doppler based systems used in [13] raise fewer privacy concerns, but they suffer from reflection and blind spots. In particular, for a data-driven system, another aspect that should not be underestimated is the need for a re-training when changing the environment to be monitored or even just some of its components such as the arrangement of furniture as happens in [14]. All this implies that there is no optimal choice, which is instead, a compromise that depends on the type of environment that is monitored as well as on personal sensitivity of the subjects under monitoring. Going into more detail, another significant distinction between falls detection systems can be made based on the type and amount of data used for the algorithm development [8]. In fact, the problem can be approached either as supervised or unsupervised based on the availability of data in the hands of the researchers as well as their goals. Most state-of-the-art works tackle the problem under fully supervised conditions assuming they have enough data for falls. Almost all of these falls are simulated with professional mannequins [15, 16] or by people with adequate protections [17, 18] that however may not correctly emulate an actual fall. Although this approach leads to more accurate results, there is no guarantee that it will generalize well in real situations. Other researchers opt for approaches based on outlier/anomaly detection [19, 20, 21] because of the plentiful availability of data that can represent normal activity. However, it is challenging to define what "normal activities" are for such approaches, and the risk is to raise several false alarms. Perhaps the situation that most closely approximates reality is a hybrid between the previous ones, in which a large amount of data representing the normality are easily available, with just a few samples of real human fall (RHF) and eventually some related synthetic or simulated data. In these situations, supervised approaches that suffer from strong data imbalance have to apply subsampling [22] or weighting [8] techniques to mitigate this effect. Thus, the need to find an effective way to exploit

the few available falls data is evident.

1.2 State-Of-The-Art

Review dei sistemi per la fall detection basati sui vari tipi di sensori accelerometers, vision, ambient. Per gli ambient paricolare enfasi sugli approcci basati su aduio.

As aforementioned, fall detection approaches can be divided based on their sensing technology, in particular if they employ wearable or ambient sensors. Regarding the first ones, the most common choice is to employ accelerometers. The algorithms proposed in [23] and [24] detect a fall by verifying if the acceleration signals exceed a certain threshold. In contrast, in [25] the authors implemented several machine learning techniques and studied their classification performance. For the experiments, a fall events dataset has been developed using six sensor units with three-axis accelerometers and worn by 14 persons who simulated falls from different angles. The best performing classifier resulted the k-Nearest Neighbour classifier. [26] employed radio tags worn on the user's chest, waist, and ankles, and an optional three-axial accelerometer worn on the chest. The algorithms performs a basic activity recognition, distinguishing from walking, standing, sitting, sitting on the ground, lying down, the process of sitting or lying down, the process of standing up, and falling. The actual fall is detected combining the results of two classifiers, an SVM and a decision tree, and hand-crafted rules. The authors performed the experiments on a laboratory scenario and reported accuracies of 100% combining radio tags and the accelerometer.

Differently from wearable sensors, the physical quantities captured by ambient sensors are more heterogeneous. Generally, fall detectors are based on vibration, video or acoustic sensors, sometimes in combination with presence detectors. In [27] the fall detector is based on a floor vibration sensor and the algorithm detects a fall when the vibration pattern matches the one of a human fall. The authors do not give further details on the algorithm and report 100% sensitivity and specificity on tests conducted on a dummy falls dataset. Yazar and colleagues [28] employ both passive infrared (PIR) sensors and floor vibration sensors. PIRs are employed to reduce false alarms, i.e., by detecting if a person is present in the region of interest. Single-tree complex wavelet transform features are extracted from the vibration signal and classified as fall or non-fall. In their dataset, the non-fall classes are represented by human (walking or running and sitting) or non-human activities (door slamming and a book falling). Three different classifiers have been compared: Euclidean distance, Mahalanobis distance and SVM, with the latter resulting in the most performing one, since it is able to classify human falls without errors regardless

the employment of PIR sensors.

Regarding approaches based on audio signals, a common solution is to install several microphones in the building, usually on the ceiling or near the walls. Indeed, also single-microphone approaches exist but they are much less robust to environmental noise, thus resulting in poor performance. For example, in [29], the authors employ a single far field microphone and they model audio segments by means of perceptual linear predictive (PLP) coefficients and GMM supervectors. An SVM with a kernel based on the Kullback-Leibler divergence, then, classifies the segment as being a fall or noise. For this purpose, nine classes of noise have been considered. In the experiments, the algorithm achieves an F_1-Measure of 67% in the classification task, and an accuracy equal to 64% in the detection task.

The difficulty in using a single microphone drove the scientific community to employ multi-channel algorithms. In [19] the authors present an unsupervised algorithm based on two microphones. The algorithm comprises a source separation and localization block to reduce the impact of background noise. Then, a one class Support Vector Machine is trained on MFCCs of non-fall events only. The SVM is then applied to distinguish normal sound events (i.e., sounds originating from normal activities) from abnormal ones (i.e., falls sounds). The authors validated the algorithm using simulated falls of persons only in presence of a television that produced the interfering sound. The results in terms of Area Under Curve are 0.9928 without interference and 0.9738 with 75% interference. The work by Li and colleagues [17] employs a circular microphone array to firstly determine the position of the sound source, and then to enhance the signal by applying a beamformer. The height of the sound source is used as first filter to discriminate falls from non falls. If the sound originates from a source positioned on the ground, MFCC features are extracted and a k-Nearest Neighbour classifier is employed to detect persons' falls. The algorithm has been tested on a dataset composed of 120 simulated fall sounds and 120 non-fall sounds recorded in different acoustic conditions. In presence of background noise and TV interference, the resulting AUC was equal to 0.989 (accuracy 95%) on clean conditions and 0.932 at 10 dB SNR (accuracy 89%).

An approach to improve the performance of fall detection systems is to combine the information coming from different sensors. The approach proposed by Zigel *et al.* [16] is based on a combination of sound and vibration sensors attached to the floor with a adhesive tape. The algorithm employs energy features extracted from the vibration signal to detect the fall event. Then, the event is classified as fall or non-fall with naive Bayes classifier employing features from both the vibration and sound signals. The experiments were conducted on a dataset containing falls of the "Rescue Randy" human mimicking doll and four objects, and the resulting sensitivity and specificity were respectively 97.5%

and 98.6%. [30] fuse the information coming from a Doppler sensor and motion sensors and classify falls with an SVM. The authors report an AUC equal to 0.98 with Doppler sensor only, and a further reduction of false alarms by 63% employing motion sensors information. Motion, sound and video signals are employed in [31]. Signals are captured both from environment sensors and from body sensors. A fall is detected by analysing sounds and motion information, while visual and motion behaviour indicates the severity of the fall. The work by Toreyin and colleagues [32] combines PIRs, microphones and vibration sensors. Signals are processed to extract features in the wavelet domain and and HMM classifier is then employed to detect falls. The authors showed the using PIR signals 100% accuracy can be obtained.

1.2.1 Problem Statement

Chapter 2

Background

In recent years, the IoT revolution has led to the creation of enormous amounts of data. The use of intelligent devices that can interface with cloud computing systems or perform complex calculations directly on board, in homes and cities, has allowed the affirmation of data-driven algorithms compared to other methodologies used so far. In fact, these approaches try to emulate the functioning of the human mind, enabling computers to perform tasks that are unthinkable until now. In this chapter are resumed the data-driven and machine learning algorithms used for developing the proposed methodologies for fall classification systems.

2.1 Support Vector Machines

Support Vector Machines (SVM) [33] are one of the most popular classification algorithms and are well known for their strong theoretical foundations, generalization performance and ability to handle high dimensional data. This section presents an overview of support vector machine, starting with linear SVMs, followed by their extension to the nonlinear case and finally the One-Class SVM for novelty detection.

Linear Support Vector Machines In the binary classification setting, let $((x_1, y_1) \dots (x_n, y_n))$ be the training dataset where $x_i \in \Re^n$ are the n-dimensional feature vectors representing the instances (i.e. observations) and $y_i \in \{-1, +1\}$ be the labels of the instances. Support vector learning is the problem of finding a separating hyperplane that separates the positive examples (labeled +1) from the negative examples (labeled -1) with the largest margin:

$$f(\vec{w}) = \text{sign}(\vec{w}^T \cdot \vec{x} + b), \tag{2.1}$$

where a value of -1 indicates one class, and a value of $+1$ the other class. In the simpler linearly separable problem, the margin of the hyperplane is defined as the shortest distance between the positive and negative instances that are

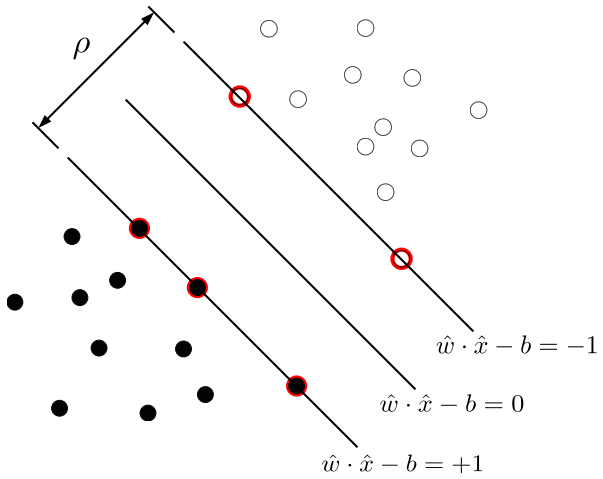


Figure 2.1: A hyperplane separating two classes with the maximum margin. The red highlighted points are the support vectors.

closest to the hyperplane. The intuition behind searching for the hyperplane with a large margin is that a hyperplane with the largest margin should be more resistant to noise than a hyperplane with a smaller margin.

Formally, suppose that all the data satisfy the constraints

$$\vec{w} \cdot \vec{x}_i + b \geq +1 \text{ for } y_i = +1, \tag{2.2}$$

$$\vec{w} \cdot \vec{x}_i + b \leq +1 \text{ for } y_i = -1, \tag{2.3}$$

where $\vec{w}$ is the normal to the hyperplane, $\frac{|b|}{\|\vec{w}\|}$ is the perpendicular distance from the hyperplane to the origin, and $\|\vec{w}\|$ is the Euclidean norm of $\vec{w}$. These two constraints can be expressed in compact form as:

$$y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1. \tag{2.4}$$

The *canonical hyperplane* is the hyperplane that separates the data and has maximal margin. The margin ρ can be computed as the distance between the two canonical hyperplanes:

$$\rho = \frac{1-b}{\|\vec{w}\|} - \frac{-1-b}{\|\vec{w}\|} = \frac{2}{\|\vec{w}\|} \tag{2.5}$$

Thus, we need to solve an optimisation problem, finding the hyperplane that maximises the margin and ensures the classes are separable

$$\min_{\vec{w}_i, b} \frac{1}{2}\|\vec{w}\|^2 \text{ subject to } y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1. \tag{2.6}$$

The problem can be expressed in the Lagrangian formulation:

$$\mathcal{L}(\vec{w}, b, \lambda) = \frac{1}{2}\|\vec{w}\|^2 + \sum_{i=1}^{m} \lambda_i (1 - y_i(\vec{w} \cdot \vec{x}_i + b)) \tag{2.7}$$

with Lagrange multipliers $\lambda_i \geq 0$ for each constraint in 2.6. The objective is then to minimize 2.7 with respect to $\vec{w}$ and b and simultaneously require that the derivatives of $\mathcal{L}(\vec{w}, b, \lambda)$ with respect to all the λ vanish. The advantage is twofold: the training vectors only appear as a scalar product among the vectors, and the constraints are easier to manage.

With the formulation presented above, the SVM fails in some situation. In fact, there is no solution if samples can not be separated by a hyperplane. Moreover, although data are linearly separable the SVM may overfit to some outlier compromising system performance. For dealing with this type of problem, has been developed the soft margin SVM [33] which allows data points to lie within the margins. Introducing *slack variables* ξ_i into the constraints and penalize them in objective, the new problem becomes

$$\min_{\vec{w}_i, b, \vec{\xi}} \frac{1}{2}\|\vec{w}\|^2 + C\sum_{i=1}^{m} \xi_i \tag{2.8}$$

$$\text{subject to } y_i(\vec{w} \cdot \vec{x}_i + b) \geq 1 - \xi_i \text{ and } \xi_i \geq 0 \text{ for } i = 1 \cdots m.$$

The cost coefficient C > 0 is a hyper-parameter that specifies the misclassification penalty and is tuned by the user based on the classification task and dataset characteristics.

Non-Linear Support Vector Machines A way to solve the problem when data are not linearly separable, is to map the data on to a higher dimensional space and then to use a linear classifier in the higher dimensional space. This methods is referred to as "the kernel trick " that exploit the fact that the training data appears as a dot product between vectors in the Lagrangian formulation to from non-linear decision boundaries. Suppose to use a transformation $\Phi : \vec{x} \rightarrow \phi(\vec{x})$ to map every data sample into higher dimensional space, the dot product becomes $\phi(\vec{x}_i)^T \phi(\vec{x}_j)$. By the use of a kernel function

$$K(\vec{x}_i, \vec{x}_j) = \langle \phi(\vec{x}_i), \phi(\vec{x}_j) \rangle, \tag{2.9}$$

it is possible to compute the separating hyperplane without explicitly carrying out the mapping into feature space. The classifier become:

$$f(\vec{x}) = \text{sign}(\sum_i \lambda_i y_i K(\vec{x}_i, \vec{x}_j) + b) \tag{2.10}$$

The most popular kernel functions are:

- Linear Kernel:
$$K(\vec{x}_i, \vec{x}_j) = \langle \vec{x}_i, \vec{x}_j \rangle \tag{2.11}$$
- Polynomial Kernel:
$$K(\vec{x}_i, \vec{x}_j) = (\langle \vec{x}_i, \vec{x}_j \rangle)^d \tag{2.12}$$
- Sigmoid Kernel:
$$K(\vec{x}_i, \vec{x}_j) = tanh(\gamma \langle \vec{x}_i, \vec{x}_j \rangle - \theta) \tag{2.13}$$
- RBF Kernel:
$$K(\vec{x}_i, \vec{x}_j) = \exp(-\frac{\|\vec{x}_i - \vec{x}_j\|}{2\sigma^2}) \tag{2.14}$$

Up to now the SVM algorithm for binary classification has been described. This algorithm can be extended to the multi-class case using the "one vs all" technique [34].

2.1.1 One-Class Support Vector Machines

One-Class SVM (OCSVM) proposed by Schölkopf et al. [35] is the extension of the support vector machine to the case of unlabeled data that makes them useful for novelty detection problems. In the OCSVM, a new parameter ν that controls the trade-off between maximizing the distance of the hyperplane from the origin and the number of data points contained by the hyperplane has been introduced. To separate the data from the origin, the following quadratic program has to be solved:

$$\min_{\vec{w}_i, \vec{\xi}, \rho} \frac{1}{2}\|\vec{w}\|^2 + \frac{1}{\nu l}\sum_{i=1}^{m} \xi_i - \rho \tag{2.15}$$

$$\text{subject to} \quad (\vec{w} \cdot \phi(\vec{x}_i)) \geq \rho - \xi_i \text{ and } \xi_i \geq 0 \text{ for } i = 1 \cdots m.$$

In fact, One-Class SVM consists in a discriminant function that takes the value $+1$ in a small region that captures the majority of the data points of a set and -1 outside that region [36]. The discriminant function has the following expression:

$$f(\mathbf{x}) = \text{sgn}\left(\sum_i \alpha_i \cdot k(\mathbf{x}_i, \mathbf{x}) - \rho\right), \tag{2.16}$$

where $\vec{x}_i$ denotes the i-th support vector. The position of the hyperplane, thus, defines the region that represents normal data points. For each point $\mathbf{x}$ that lies outside this region, the function $f(\vec{x})$ takes the value -1, whereas for point

inside the region, it takes the value +1. The terms λ_i can be found by solving the solution to the dual problem:

The terms λ_i can be found by solving the solution to the dual problem:

$$\min_{\lambda} \frac{1}{2} \sum_{ij} K(\vec{x_i}, \vec{x_j}) \tag{2.17}$$

$$\text{subject to} \quad 0 \leq \lambda_i \leq \frac{1}{\nu l} \quad \text{and} \quad \sum_i \lambda_i = 1,$$

where λ_i is a Lagrange multiplier and l is the number of points in the training dataset. The term $\nu \in (0, 1]$ is an hyperparameter of the algorithm that is determined on a validation set.

The offset ρ can be obtained from the Karush-Kuhn-Tucker (KKT) condition with the expression [37]:

$$\rho = \sum_j \lambda_i k(\vec{x}_j, \vec{x}_i), \tag{2.18}$$

which is satisfied for any λ_i that is not at the upper or lower bound.

2.2 Gaussian Mixture Model

https://pdfs.semanticscholar.org/734b/07b53c23f74a3b004d7fe341ae4fce462fc6.pdf
A Gaussian Mixture Model (GMM) is a parametric probability density function represented as a weighted sum of Gaussian component densities. Generally, GMMs are used as a parametric model of the probability distribution of some features. To estimate the parameter of GMM the algorithm Expectation-Maximization (EM) algorithm or Maximum A Posteriori (MAP) are used starting from a well-trained prior model usually named Universal Background Model (UBM). A Gaussian mixture model is a weighted sum of M component Gaussian densities as given by the equation

$$p(x, \vec{\lambda}) = \sum_{i=1}^{M} w_i g(\vec{x}|\vec{\mu}_i, \vec{\Sigma}_i) \tag{2.19}$$

where $\vec{x}$ is a D-dimensional features vector, $g(\vec{x}|\vec{\mu}_i), \vec{\Sigma}_i)$ are the components of the mixture and w_i are the weight of each component. Each component of the mixture is a D-variate Gaussian density function expressed as

$$a \tag{2.20}$$

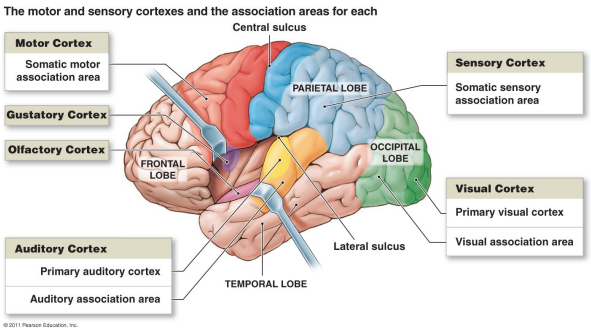


Figure 2.2: The human brain.

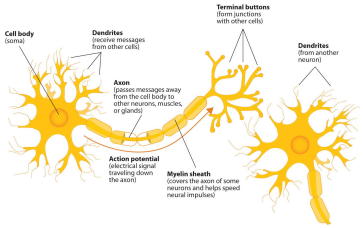


Figure 2.3: The neuron model.

2.3 K-Nearest Neighbor

2.4 Deep Neural Network (DNN)

A *biological Neural Networks* is a big set of specialized cells (*neurons*) connected among them, which memorize and process information, thus controlling the body activities they belong to.

The *neuron* model is composed of:

- DENDRITE: input terminal
- CELL BODY (Nucleus): processing core
- AXON: output way-out
- SYNAPSES: output terminal (with weight)

The *neuron* properties can be described in:

- LOCAL SIMPLICITY: the neuron receives stimuli (excitation or inhibition) from dendrites and produces an impulse to the axon which is proportional to the weighted sum of the inputs;

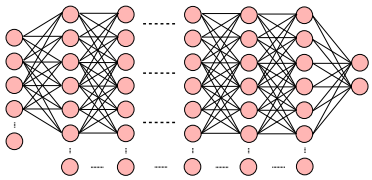


Figure 2.4: The Artificial Neural Network.

- GLOBAL COMPLEXITY: the human brain possess $\mathcal{O}(10^{10})$ neurons, with more than 10K connections each;
- LEARNING: even though the network topology is relatively fixed, the strength of connections (synaptic weights) can change when the network is exposed to external stimuli;
- DISTRIBUTED CONTROL: no centralized control, each neuron reacts only to its own stimuli;
- TOLERANCE TO FAILURES: performance slowly decrease with the increase of failures.

The biological Neural Networks are able to solve very complex tasks in few time instants (like memorization, recognition, association, and so on.)

The *Artificial Neural Networks* (ANNs) are defined as *Massively parallel distributed processors made up of simple processing units having a natural propensity for storing experiential knowledge and making it available for use* (Haykin, 2008).

An ANN resembles the brain in two aspects:

1. Knowledge is acquired by the network from its environment through a learning process;
2. Synaptic weights are used to store the acquired knowledge.

A *neuron* is an information-processing unit that is fundamental to the operation of a neural network. The model of a neuron is composed of three basic elements of the neural model:

- a *set of synapses*, or connecting links, each of which is characterized by a weight or strength of its own, w_{kj};
- an *adder* for summing the input signals, weighted by the respective synaptic strengths of the neuron; the operations described here constitute a linear combiner;

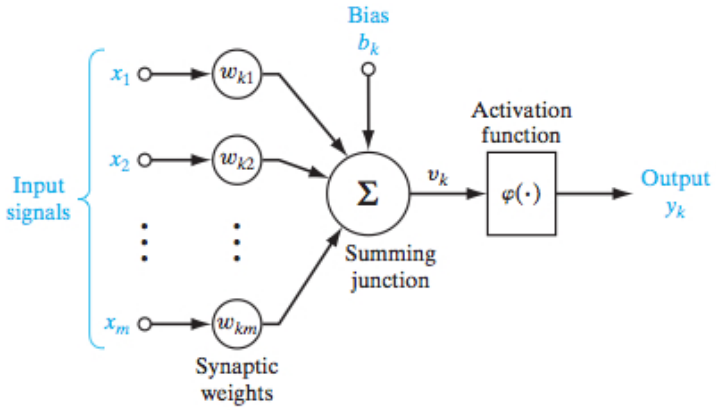


Figure 2.5: The artificial neuron model.

- an *activation function* for limiting the amplitude of the output of a neuron. Typically, the normalized amplitude range of the output of a neuron is written as the closed unit interval [0,1], or, alternatively, [-1,1].

The neural model also includes an externally applied *bias*, denoted by b_k.

Therefore, the mathematical description of neuron activity can be defined as:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{2.21}$$

$$y_k = \varphi\left(u_k + b_k\right) \tag{2.22}$$

where:

- $x_1, x_2, \cdots, x_m$ are the input signals;
- $w_{k1}, w_{k2}, \cdots, w_{km}$ are the respective synaptic weights of neuron k;
- u_k is the linear combiner output due to the input signals;
- b_k is the bias;
- $\varphi(\cdot)$ is the activation function;
- y_k is the output signal of the neuron.

The types of *activation non-linear functions* $\varphi(x)$ are:

- the *threshold function*: in engineering, this form of a threshold function is commonly referred to as a Heaviside function;

$$\varphi\left(v\right) = 1 \quad if \quad v \geq 0 \tag{2.23}$$

$$\varphi\left(v\right) = 0 \quad if \quad v < 0 \tag{2.24}$$

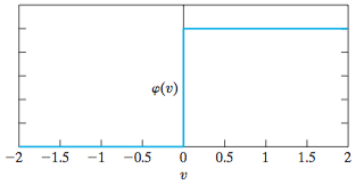


Figure 2.6: The threshold non-linear function.

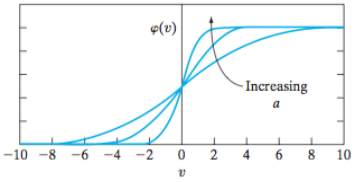


Figure 2.7: The sigmoid non-linear function.

- the *sigmoid function*: it is defined as a strictly increasing function that exhibits a graceful balance between linear and nonlinear behavior; an example of the sigmoid function is the *logistic function* defined by:

$$\varphi(v) = \frac{1}{1 + exp(-av)} \tag{2.25}$$

- the *hyperbolic tangent* (*tanh*): it is simply a scaled and shifted version of the sigmoid function:

$$\varphi(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{2.26}$$

- the *Rectifier Linear Unit* (*ReLU*):

$$\varphi(x) = \max(0, x) \tag{2.27}$$

- the $softmax$: it is used on the last layer of a classifier setup: the outputs of the softmax layer represent the probabilities that a sample belongs to

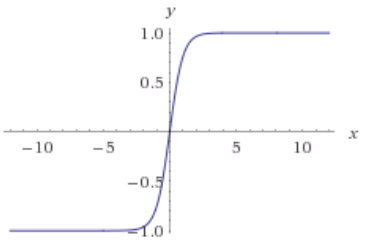


Figure 2.8: The *tanh* non-linear function.

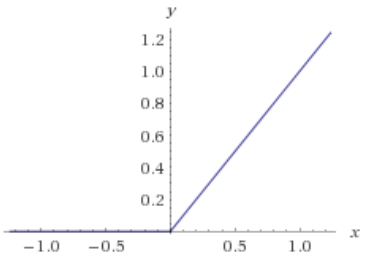


Figure 2.9: The *ReLU* non-linear function.

the different classes. Indeed, the sum of all the output is equal to 1.

$$\varphi(x_k) = \frac{e^{x_k}}{\sum_{j=1}^{N} e^{x_j}} \text{ for } k = 1, \ldots, K \tag{2.28}$$

The manner in which the neurons of a neural network are structured is intimately linked with the learning algorithm used to train the network. There, the *network architectures* (structures) is defined. In general, two different classes of network architectures are identified:

1. *Multilayer Feedforward Networks* - (FFNN):

   it is characterized by the presence of one or more hidden layers, whose computation nodes are correspondingly called *hidden neurons* (or hidden units); the term *hidden* refers to the fact that this part of the neural network is not seen directly from either the input or output of the network. The function of hidden neurons is to intervene between the external input and the network output in some useful manner. By adding one or more hidden layers, the network is enabled to extract higher-order statistics from its input.

   The MLP is a well known kind of artificial neural network introduced in 1986 [38]. Each node applies an activation function over the weighted sum of its inputs. The units are arranged in layers, with feed forward

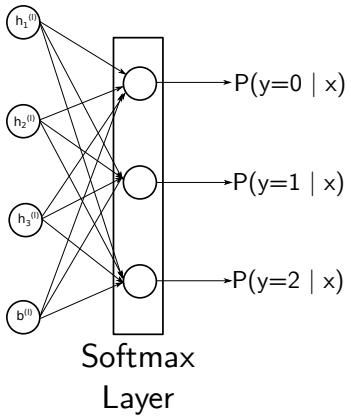


Figure 2.10: The *softmax* layer in a neural network classifier.

connections from one layer to the next. The stochastic gradient descent with error back-propagation algorithm is used for the supervised learning of the network. In the forward pass, input examples are fed to the input layer, and the resulting output is propagated via the hidden layers towards the output layer. At the backward pass, the error signal originating at the output neurons is sent back through the layers and the network parameters (i.e., weights and biases) are tuned.

A single neuron can be formally described as:

$$g(\mathbf{u}[n]) = \varphi\left(\sum_{j=1}^{D} w_j u_j[n] + b\right), \tag{2.29}$$

where $\mathbf{u}[n] \in \mathbb{R}^{D\times 1}$, the bias b is an externally applied term and $\varphi(\cdot)$ is the non-linear activation function. Thus, the mathematical description of a one-hidden-layer MLP is a function $\mathbf{f} : \mathbb{R}^D \to \mathbb{R}^{D'}$, where D' is the size of the output vector, so:

$$\mathbf{f}(\mathbf{u}[n]) = \varphi\left(\mathbf{b}_2 + \mathbf{W}_2\left(\varphi\left(\mathbf{b}_1 + \mathbf{W}_2 \cdot \mathbf{u}[n]\right)\right)\right), \tag{2.30}$$

where $\mathbf{W}_i$ and $\mathbf{b}_i$ are the respective synaptic weights matrix and the bias vector of the i-th layer. The behaviour of this architecture is parametrized by the connection weights, which are adapted during the supervised network training.

2. *Convolutional Neural Networks*(CNN)

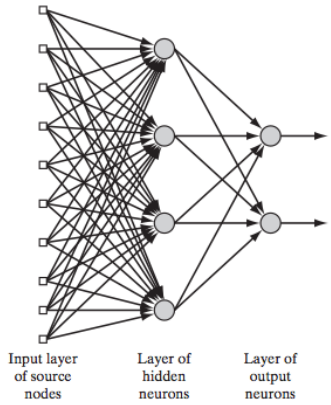


Figure 2.11: The Multilayer Feedforward Network.

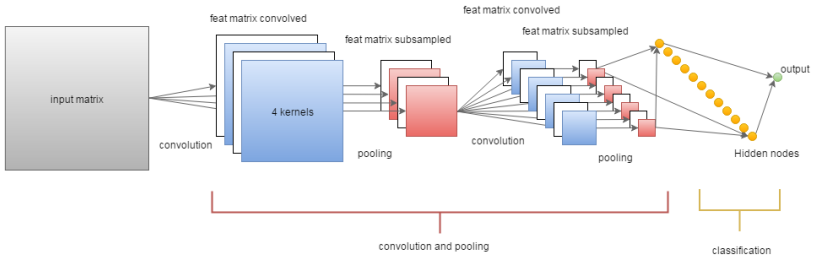


Figure 2.12: The Convolutional Neural Network.

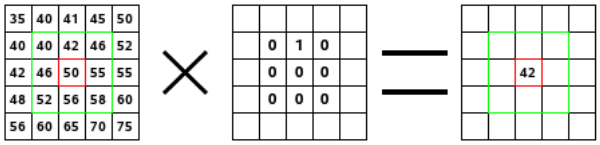


Figure 2.13: The convolution operation.

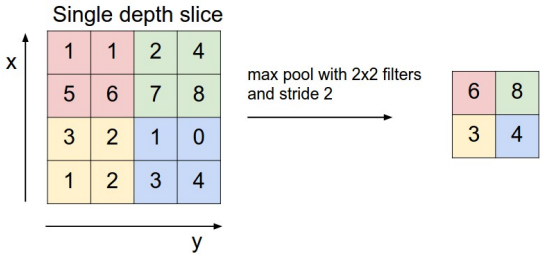


Figure 2.14: The max-pooling layer.

Convolutional neural networks are feedforward neural networks similar to multilayer perceptron, with some special layers.

Convolution kernels process the input data matrix by dividing it in *local receptive fields*, a region of the same size of the kernel, and sliding the local receptive field across the entire input. Each hidden neuron is thus connected to a local receptive field, and all the neurons form a matrix called *feature map*. The weights in each *feature map* are *shared*: all hidden neurons are aimed to detect exactly the same pattern just at different locations in the input image.

The main advantages of this network is the robust pattern recognition system characterized by a strong immunity to pattern shifts.

Pooling layer just reduces the dimension of the matrix by a rule: a submatrix of the input is selected, and the output is the maximum value of this submatrix.

The pooling process introduces tolerance against shifts of the input patterns. Together with convolution layer it allows the CNN to detect if a particular event occurs, regardless its deformation or its position.

CNN is a feed-forward neural network [39] usually composed of three types of layers: convolutional layers, pooling layers and layers of neurons. The convolutional layer performs the mathematical operation of convolution between a multi-dimensional input and a fixed-size kernel. Successively, a non-linearity is applied element-wise. The kernels are generally small compared to the input, allowing CNNs to process large inputs with

few trainable parameters. Successively, a pooling layer is usually applied, in order to reduce the feature map dimensions. One of the most used is the *max-pooling* whose aim is to introduce robustness against translations of the input patterns. Finally, at the top of the network, a layer of neurons is applied. This layer does not differ from MLP, being composed by a set of activation and being fully connected with the previous layer. For clarity, the units contained in this layer will be referred as *Hidden Nodes* (HN).

Denoting with $\mathbf{W}_m \in \mathbb{R}^{K_{1m} \times K_{2m}}$ the m-th kernel and with $\mathbf{b}_m \in \mathbb{R}^{D_1 \times D_2}$ the bias vector of a generic convolutional layer, the m-th feature map $\mathbf{h}_m \in \mathbb{R}^{D_1 \times D_2}$ is given by:

$$\mathbf{h}_m = \varphi\left(\sum_{d=1}^{D_3} \mathbf{W}_m * \mathbf{u}_d + \mathbf{b}_m\right), \tag{2.31}$$

where $*$ represent the convolution operation, and $\mathbf{u}_d \in \mathbb{R}^{D_1 \times D_2}$ is a matrix of the three-dimensional input tensor $\mathbf{u} \in \mathbb{R}^{D_1 \times D_2 \times D_3}$. The dimension of the m-th feature map $\mathbf{h}_m$ depends on the zero padding of the input tensor: here, padding is performed in order to preserve the dimension of the input, i.e., $\mathbf{h}_m \in \mathbb{R}^{D_1 \times D_2}$. Please note that for the sake of simplicity, the time frame index n has been omitted. Commonly, (2.31) is followed by a pooling layer in order to be more robust against patterns shifts in the processed data, e.g. a max-pooling operator that calculates the maximum over a $P_1 \times P_2$ matrix is employed.

A *Deep Learning* definition: *A class of machine learning techniques that exploit many layers of non-linear information processing for supervised or unsupervised feature extraction and transformation, and for pattern analysis and classification.* Artificial Neural Networks are often referred as deep when they have more than 1 or 2 hidden layers.

2.4.1 Stochastic gradient descent (SGD)

Most deep learning training algorithms involve optimization of some sort. The most widely used is the gradient based optimization, which belongs to the first order type.

Optimization is the task of either minimizing some function $f(x)$ by altering x: $f(x)$ is called *objective function*, but in the case when it has to be minimized, it is also call the *cost function*, *loss function*, or *error function*. The aim of the optimization is reached doing small change ϵ in the input x, to obtain the

corresponding change in the output $f(x)$:

$$f(x+\epsilon) \approx f(x) + \epsilon\, f'(x). \tag{2.32}$$

This formulation is based on the calculation of the derivative $f'(x)$. The *gradient descent* is the technique based on the reduction of $f(x)$ by moving x in small steps with the opposite sign of the derivative. The aim is to find the minimum of the cost function: when $f'(x) = 0$, the derivative provides no information about which direction to move, therefore this point is defined as stationary points. A local minimum is a point where $f(x)$ is lower than at all neighbouring and it is no longer possible to decrease $f(x)$ by making infinitesimal steps. The absolute lowest value of $f(x)$ is a *global minimum*.

For the concept of minimization to make sense, there must still be only one (scalar) output. For functions that have multiple inputs $f : \mathbb{R}^n \to \mathbb{R}$, the concept of *partial derivatives* is introduced. The gradient $\nabla_{\mathbf{x}} f(\mathbf{x})$ is the vector containing all the partial derivatives.

The method of *steepest descent* or *gradient descent* states that decrease f by moving in the direction of the negative gradient.

$$\mathbf{x'} = \mathbf{x} - \epsilon\, \nabla_{\mathbf{x}} f(\mathbf{x}), \tag{2.33}$$

where ϵ is the *learning rate*, a positive scalar determining the size of the step.

Large training sets are necessary for good generalization, but large training sets are also more computationally expensive. The cost function decomposes as a sum over training example of per-example loss function: i.e., the negative conditional log-likelihood of the training data is defined as:

$$J(\theta) = \mathbb{E}(L(\mathbf{x}, y, \theta)) = \frac{1}{m} \sum_{i=1}^{m} L(\mathbf{x}^{(i)}, y^{(i)}, \theta), \tag{2.34}$$

where L is the per-example loss $L(\mathbf{x}, y, \theta) = -\log p(y|\mathbf{x}; \theta)$. The gradient descent requires computing:

$$\nabla_\theta J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \nabla_\theta L(\mathbf{x}^{(i)}, y^{(i)}, \theta). \tag{2.35}$$

The computational cost of this operation is proportional to the number of example m, therefore as the training set size grows the time to take a single gradient step becomes prohibitively long.

Stochastic gradient descent (SGD) is an extension of the gradient descent algorithm: the insight is that the gradient is an expectation estimated using a small set of samples. On each step of the algorithm, a sample of example

$\mathbb{B} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m')}\}$, called *minibatch*, is drawn uniformly from the training set. The minibatch size m' is typically chosen to be a relatively small number of examples. The estimate of the gradient is: $\mathbf{g} = \frac{1}{m'} \nabla_\theta \sum_{i=1}^{m'} L(\mathbf{x}^{(i)}, y^{(i)}, \theta)$ using examples from the minibatch $\mathbb{B}$. The SGD algorithm then follows the estimated gradient downhill:

$$\theta \leftarrow \theta - \epsilon\, \mathbf{g} \tag{2.36}$$

where ϵ is the learning rate.

2.4.2 Autoencoder

An Autoencoder is a kind of neural network typically consisting of only one hidden layer, trained to set the target values to be equal to the inputs.

$$\tilde{x} = f(W_2 h(x) + b_2) \tag{2.37}$$

Given an input set of examples $\mathcal{X}$, autoencoder training consists in finding parameters $\theta = \{W_1, W_2, b_1, b_2\}$ that minimize the Reconstruction Error:

$$\mathcal{J}(\theta) = \sum_{x \in \mathcal{X}} \|x - \tilde{x}\|^2 \tag{2.38}$$

Defining M the number of hidden units, and N the number of input units, output units, features size:

- (a): $M = N \rightarrow$ Basic Autoencoder (AE);
- (b): $M < N \rightarrow$ Compression Autoencoder (CAE);
- (c): $M > N$ and Gaussian Noise $\rightarrow$ Denoising Autoencoder (DAE);

An AE – a kind of neural network typically consisting of only one hidden layer –, sets the target values to be equal to the input. It is used to find common data representation from the input [40, 41]. Formally, in response to an input example $x \in \mathbf{R}^n$, the hidden representation $h(x) \in \mathbf{R}^m$ is

$$h(x) = f(W_1 x + b_1), \tag{2.39}$$

where $f(z)$ is a non-linear activation function, typically a logistic sigmoid function $f(z) = 1/(1+\exp(-z))$ applied component-wisely, $W_1 \in \mathbf{R}^{m \times n}$ is a weight matrix, and $b_1 \in \mathbf{R}^m$ is a bias vector.

The network output maps the hidden representation h back to a reconstruction $\tilde{x} \in \mathbf{R}^n$:

$$\tilde{x} = f(W_2 h(x) + b_2), \tag{2.40}$$

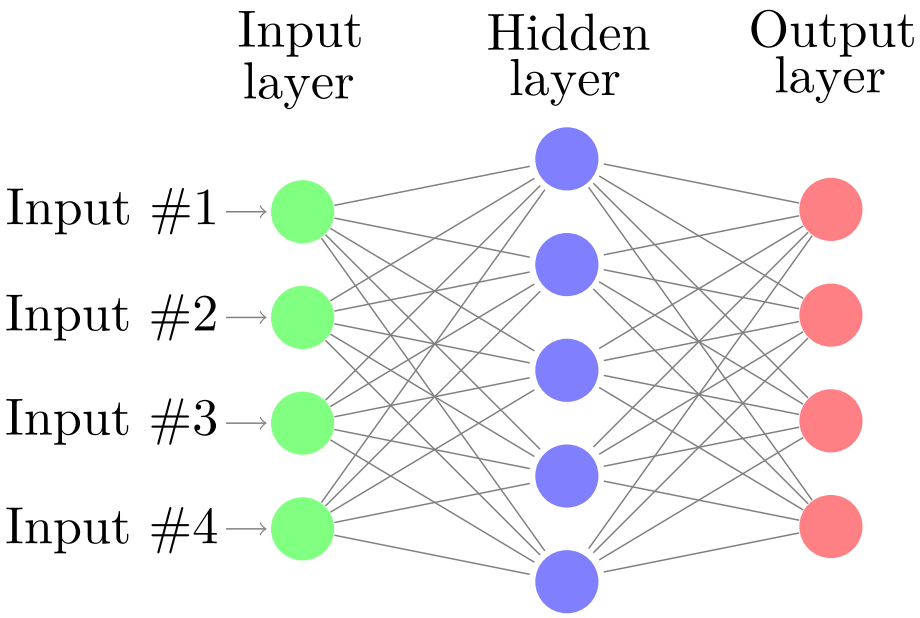


Figure 2.15: The different types of Autoencoders.

where $W_2 \in \mathbf{R}^{n \times m}$ is a weight matrix, and $b_2 \in \mathbf{R}^n$ is a bias vector.

Given an input set of examples $\mathcal{X}$, AE training consists in finding parameters $\theta = \{W_1, W_2, b_1, b_2\}$ that minimise the reconstruction error, which corresponds to minimising the following objective function:

$$\mathcal{J}(\theta) = \sum_{x \in \mathcal{X}} \|x - \tilde{x}\|^2 . \tag{2.41}$$

The minimisation is usually realised by stochastic gradient descent as in the training of neural networks. The structure of the AE is given in Figure 2.15a.

Chapter 3

Dataset

The importance of using public data sets for algorithm evaluation is very important. Only in this way can a direct comparison be made between the different approaches to determine which of these is actually the best. There are publicly available datasets for the fall detection task, the majority of them are all related to wearable or vision sensors and often include both types [42, 43, 44, 45, 46]. Since in this work, we face the problem fall detection from an audio perspective, only one dataset containing audio recording has been found [47]. However, the audio files available in the dataset are suitable for speech recognition related works rather than sound event detection. In fact, only the utterance of short sentences or interjections of the actors involved during the human falls recordings have been annotated. As in this work, several data-driven approaches for pattern recognition produced by the sound generated by the human fall are presented, the dataset [47] result useless. Given the lack of available audio data sets, we have created a suitable audio dataset in order to assess the proposed approaches. This choice was also forced by the fact that in these works an innovative acoustic sensor was explicitly developed for the fall detection and described in Section 3.1 has been used. In this chapter, the instrumentation, the procedure for recording the audio corpus and its composition are described.

3.1 The floor acoustic sensor

The floor acoustic sensor (FAS) is composed of a resonant enclosure and a microphone located inside it (Figure 3.1) [48]. At the bottom of the enclosure, a membrane is in direct contact with the floor and guarantees the acoustic coupling with the surface. The inner container accommodates the microphone and is where the acoustic resonance phenomenon takes place. It can be covered by a layer of acoustic isolation material and it is enclosed by the outer container that further reduces the intensity of the acoustic waves that propagate through air. The enclosure has been manufactured in Polylactic Acid with a 3D printer, its diameter is 16.5 cm and its height 5.5 cm.

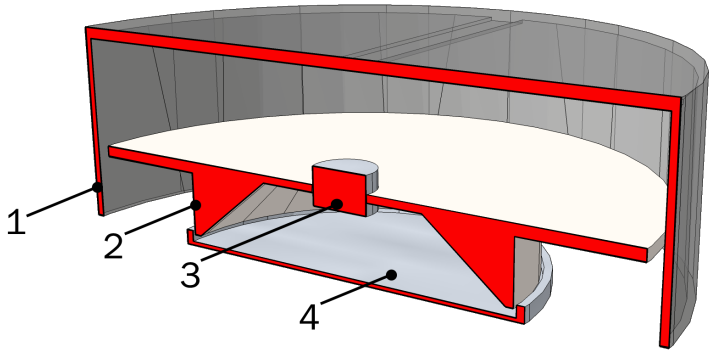


Figure 3.1: The floor acoustic sensor: conceptual scheme. 1 - The outer container. 2 - The inner container. 3 - The microphone slot. 4 - The membrane touching the floor.

Regarding the microphone, an AKG C 400 BL[1] has been inserted in the enclosure. The outer case of the microphone has been removed to extract the capsule that has then been inserted in the sensor enclosure. The AKG C 400 BL is characterized by an hypercardiod directivity pattern, thus it has been oriented so that the maximum gain is towards floor.

3.2 The fall events dataset: A3Fall-v1.0

The performance of the floor acoustic sensor has been evaluated on a corpus of audio events corresponding to falls of several objects recorded in different conditions[2]. The dataset has been specifically created by the authors and it will be presented in this section.

3.2.1 The recording setup

Fall events have been recorded in 3 different rooms with the following characteristics:

- The first, is a rectangular room, hereafter named R0, measuring about $7\,\mathrm{m} \times 2\,\mathrm{m}$ (Figure 3.5). The room is particularly suitable for the propagation of acoustic waves through the floor since it is obtained from a cantilever beam. In addition, the considerable distance of the supporting pillars facilitates the transmission of a fall vibrations through the floor.

[1] http://www.akg.com/pro/p/c400-bl
[2] The dataset is available at the following URL: `http://www.a3lab.dii.univpm.it/research/fasdataset`

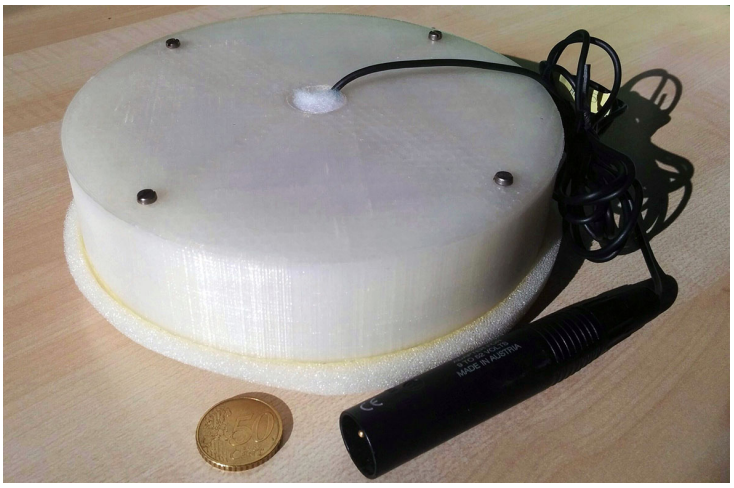


Figure 3.2: A picture of the floor acoustic sensor used during the recordings.

- the second location for the recording was the university auditorium room (R1) in which the flooring is composed of fitted carpet. This makes it particularly suitable for evaluating system performance on surfaces with acoustical behavior that can mitigate the impact sound transmitted through the floor and in the air; all the recordings were performed near the auditorium stage in an area of 8×3 m.

- a recording studio (R2) was selected as the third location for its particular characteristics. Here, it was possible to make the acquisitions by placing the sensors in the live room while the audio events were performed in the control room. In particular, the sensors were positioned immediately behind the soundproof wall with the window overlooking the live room. The size of the live room is 5×7 m, while the size of the control room is 3×8 m.

The recording equipment comprises the floor sensor, a linear array of three aerial microphones (the same AKG 400 BL included in the floor sensor) and a Presonus AudioBox 44VSL sound card connected to a laptop. The microphones of the array are separated by 4 cm and positioned on a table 80 cm high. Signals were sampled at 44.1 kHz with a resolution of 32 bits. Levels were calibrated to assure the maximum dynamic range at the smallest distance.

3.2.2 Description

In Table 3.1 the composition of the dataset is summarized. For the R0, the dataset comprises recordings of fall events related to everyday objects and to

Table 3.1: Composition of the A3Fall-v2.0 dataset.

Class	**R0**	**R1**	**R2**
	Nr. of occurrences		
Basket	64	40	40
Fork	64	40	40
Ball	64	40	40
Book	64	40	40
Bag	64	30	40
Chair	96	40	40
Table	0	40	40
Guitar Slide	0	40	40
Nipper	0	40	40
Keys	0	40	40
Hook	0	40	40
Coat Hook	0	40	40
Manikin Doll	44	0	0
Human Fall	0	40	40
	Total length (s)		
Background	2530	9055	5550

a human mimicking doll. The objects were chosen according to the recent literature on the topic [27] and are the following: a ball, a metal basket, a book, a metal fork, a plastic chair, and a bag (Figure 3.3). Objects have been dropped at four distances from the sensors, i.e., 1 m, 2 m, 4 m, and 6 m, and with various angles in order to reproduce realistically different fall patterns. With the exception of the chair and the basket, which have been overturned from their natural position, half of the falls has been performed at a height of 0.5 m and the other half at 1 m. For each object and for each distance, 16 fall events have been performed for a total of 64 events per object. Instead, the chair has been overturned 8 times for each side of fall (back, front, side) and for each distance, thus obtaining a total of 96 events.

Human falls have been simulated by employing the "Rescue Randy" doll[3], a professional equipment employed in water rescues. It weights 75 kg, it is 1.85 m high, and it is equipped with articulated joints. The doll is made of vinyl and its weight is distributed according to the human weight distribution chart. The doll has been dropped from upright position and from a chair, both forward and backward, for a total of 44 events (Figure 3.4). Differently from the everyday objects, the distribution of the fall events with the distance is not uniform: 10 events have been performed from 2 m, 18 from 4 m (7 of which from the chair),

[3]http://www.simulaids.com/1475.htm

Figure 3.3: Objects employed for creating the fall events dataset.

and 16 from 6 m (6 of which from the chair).

Moreover, several backgrounds sounds has been added to the dataset. Normal activities sounds have been recorded while persons were performing common actions, such as walking, talking, and dragging chairs. Three musical tracks have been played from a loudspeaker and acquired back with the FAS. The first track contained classical music[4], while the second[5] and the third[6] rock music. Musical tracks and normal activities sounds have been divided in segments whose lengths have mean and standard deviation estimated from instances of fall events. In addition, they have been employed alone and to create noisy versions of human and object falls occurrences in order to assess the algorithm in presence of interferences.

In R2 and R1 other every-days objects,in addition to those used in R0, have been recorded for a total of 12 different object fall classes and 1420 instances. While the manikin doll has been used only in R0, in R1 and R2 a total 80 human falls have been performed by 4 people. These falls were performed in different ways: forward, backward and on the side, trying to use the arms to cushion the fall and without any protections. As in R0, also in R2 and R2 all events were performed from 1, 2, 4 and 6 m away from the FAS.

As shown in Table 3.1 background noises have been recorded also in R1 and R2 rooms rooms, which include: human activities noise as, i.e., footsteps, human and phone conversation, dragging objects and so on; classic, rock and pop music played from loudspeakers; TV shows like newscast and satiric.

Since the data relating to rooms R1 and R2 have been collected at different times to those of room R0, in the following chapters, for each proposed ap-

[4]W. A. Mozart, "Piano trio in C major"
[5]Led Zeppelin, "Dazed and confused"
[6]Led Zeppelin, "When the levee breaks"

(a) Fall from upright position. (b) Fall from the chair.

Figure 3.4: Falls of the "Rescue Randy" doll from upright position (a) and from the chair (b).

proach, it will be specified which subset of the total dataset has been used as well as the usage of the noisy version of falls events.

3.2.3 Signal analysis

The signal related to the same fall event acquired with the floor sensor and with the aerial microphone exhibits different spectral characteristics. In this section and in depth analysis of the audio signals acquired in the R0 room is presented. Figure 3.6 shows the spectrograms of a doll fall acquired with the floor sensor (above) and with the aerial microphone (below) in the clean acoustic condition. Observing the figures, it can be noticed that the aerial microphone is more sensitive to high frequencies, in particular to the ones above 1.5 kHz. On the contrary, the majority of the energy of the signal acquired with the floor sensor concentrates below 1 kHz.

This is even more evident by plotting the values of the mel coefficients (Figure 3.7): the first and second mel channels of the FAS, corresponding to the frequency bands 0–128.10 Hz and 61.30-200.60 Hz, are higher respect to the aerial microphone. Channels 3 to 7, respectively corresponding to bands 128.10–279.50 Hz and 458.70–670.70 Hz, are almost equivalent, while from channel 8 (560.30–790.80 Hz) to 29 (6654.60–8000.00 kHz) the aerial microphone mels are greater.

The analysis of noisy signals highlights the different behaviour of the floor sensor respect to the aerial microphone in presence of external interferences.

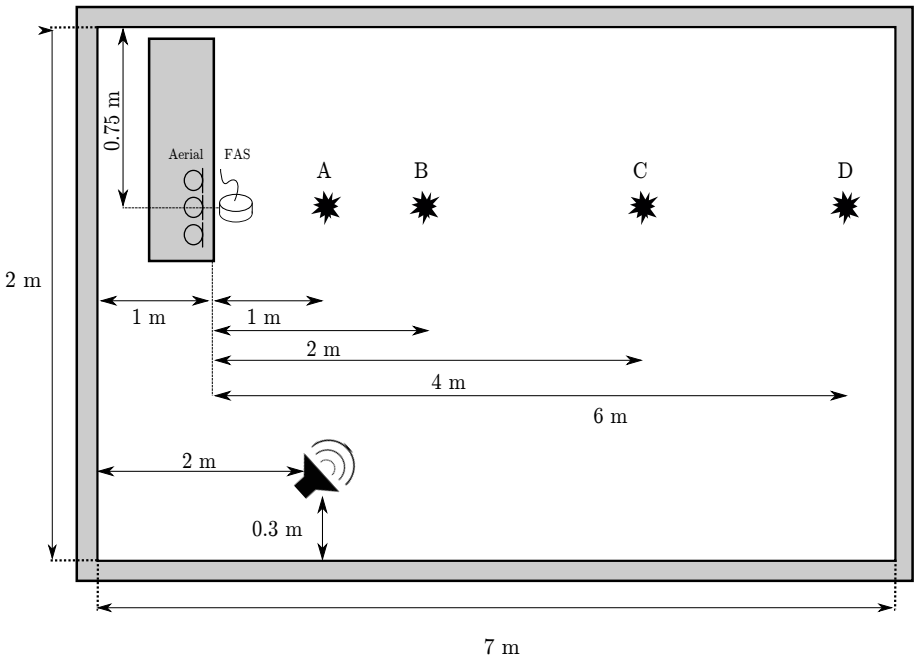
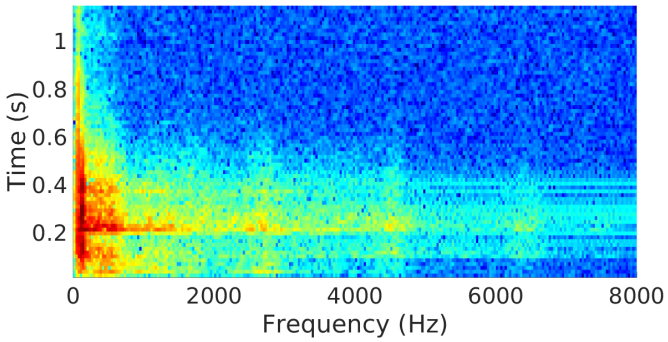
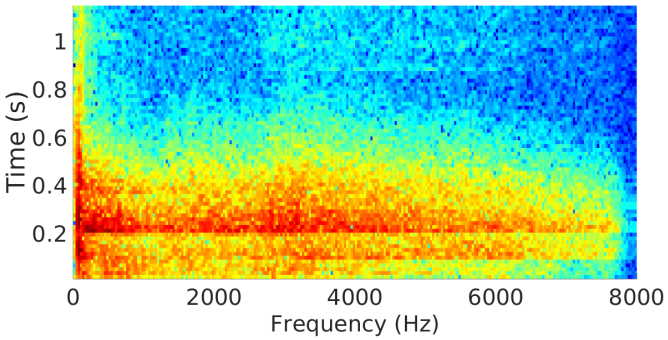


Figure 3.5: The recording room: the letters A, B, C and D indicate the positions of fall events.

In fact, taking into account the backgrounds tracks as interferences, the floor sensor has a global signal-to-noise ratio (SNR) equal to 20.94 dB and a segmental SNR equal to 7.28 dB. The global SNR of the central aerial microphone is 8.92 dB and the segmental SNR is -1.47 dB. The values of the aerial microphone SNRs are thus considerably lower than the ones of the floor sensor.The global SNR of the floor sensor noisy dataset is 13.66 dB higher than the one of the aerial microphone, highlighting the superior ability of the former to isolate fall signals from external interferences. However, it is worth investigating how the SNR distributes over the frequency range of the acquired signals in order to have a better insight of the physical phenomenon. Figure 3.8 shows the SNR calculated for each mel channel and averaged across the noisy datasets. It can be noticed that the SNR of the floor sensor exceeds the one of the aerial microphone for channels below the fourth. Then, the opposite occurs and the SNR of the aerial microphone assumes greater values. The "valley" in the curves are due to the pitch of the music signal.



(a) Spectrogram of the signal acquired with the floor sensor.



(b) Spectrogram of the signal acquired with the aerial microphone.

Figure 3.6: Frequency content of the same fall event (file "rndy_d2st_bar_0.wav") acquired with the FAS (a) and with the aerial microphone (b).

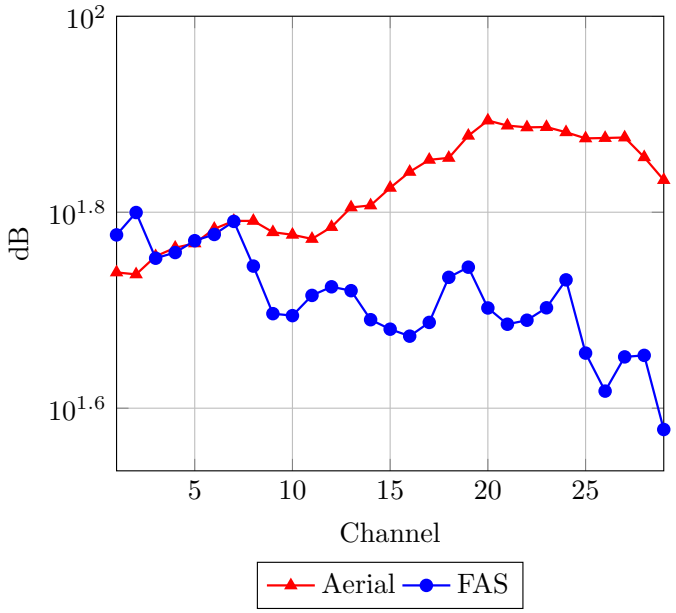


Figure 3.7: Average value of the mel channels.

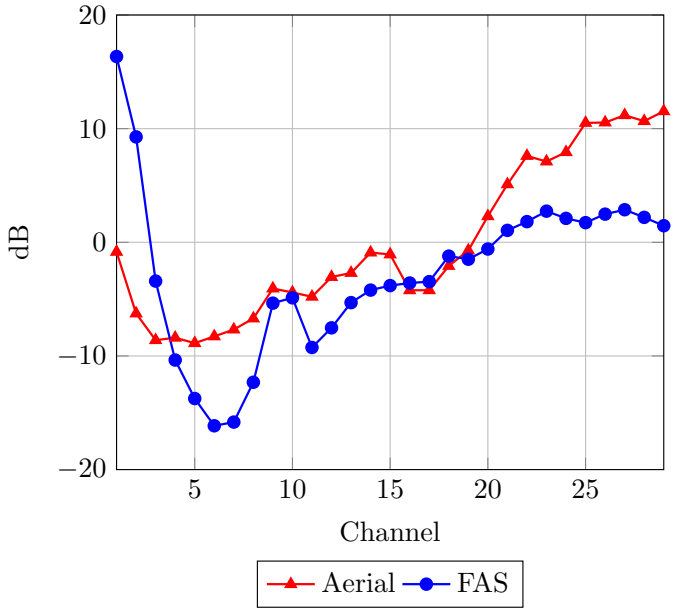


Figure 3.8: Average value of the SNR for each mel channel.

Chapter 4

Supervised Approach

qua vengono presentati sia il metodo multilabel classifier GMM-UBM SVM (ESWN) che il binary GMM-UBM SVM (WIRN2016)

4.1 Support-Vector Machine based algorithm for evens fall classification

The fall detection task consists in recognize which object produced the pattern of a signal and it mainly consists of two subtasks: the location of the time boundaries of the fall event and the classification of that event. In this section, we concentrate on the second subtask, since the main objective is determining the performance of the acoustic sensor as compared to common aerial microphones. The entire falls detection activity will be addressed in the works presented below.

The fall classification algorithm is composed of a feature extraction stage and a classification stage. The first extracts MFCCs from the input audio signal, while the second classifies the audio event by means of Gaussian means supervectors and SVM. Following is a detailed description of the two stages.

4.1.1 Feature extraction

Despite MFCCs were originally developed for speech and speaker recognition tasks, they have been successfully applied also for acoustic event classification [49] and fall detection [16].

The block-scheme of the MFCC feature extraction pipeline is shown in Figure 4.1. The first processing step is the pre-emphasis of the input signal, which consists in applying a filter whose transfer function is:

$$G(z) = 1 - \alpha z^{-1}. \tag{4.1}$$

Usually $0.9 < \alpha \leq 1.0$ and here it has been set to 0.97. The objective of pre-emphasis is to remove the DC components and to raise the high-frequency part

of the spectrum.

The signal is then segmented in frames 16 ms long overlapped by 8 ms and multiplied with a Hamming window. For each frame, the Discrete Fourier Transform (DFT) is calculated and filtered with a filterbank composed of 29 triangular filters uniformly spaced on the mel scale.

Denoting with $S(i)$ the DFT of a frame and i the frequency bin, the output of the "Mel Filterbank & Frequency Integration" block in Figure 4.1 is

$$mel(k) = \sum_{i=ini(k)}^{end(k)} |S(i)|^2 W_k(i), \qquad k = 1, 2, \ldots, N \tag{4.2}$$

where $mel(k)$ is the energy of the k-th subband, $W_k(i)$ is the frequency response of the k-th filter, $ini(k)$ and $end(k)$ are starting and ending frequency indices of that filter and N is the number of filters in the bank, which in this case is 29. The terms $mel(k)$ are often named "mel coefficients".

The final steps for the calculation of the j-th MFCC $c(j)$ is the logarithm and the Discrete Cosine Transform (DCT):

$$c(j) = \sum_{k=1}^{N} \log\left[H(k)\right] \cos\left[\frac{\pi j}{N}(k - 0.5)\right], \quad j = 0, 1, \ldots, M - 1 \leq N \tag{4.3}$$

The set $\{c(0), c(1), \ldots, c(M-1)\}$ forms the static coefficients elements of the feature vector. Here M has been set to 13. The final feature vector is composed of 39 coefficients, i.e., the 13 static coefficients plus their first and second derivatives.

4.1.2 Classification stage

The classification stage employs Gaussian Mean Supervectors (GMS) and a Support Vector Machine classifier as in speaker recognition systems [50]. The algorithms consists in modelling the entire acoustic space with a Universal Background Model (UBM) represented by mixture of gaussians (Gaussian Mix-

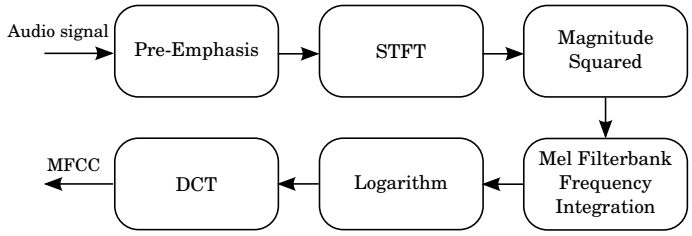


Figure 4.1: The MFCC feature extraction pipeline.

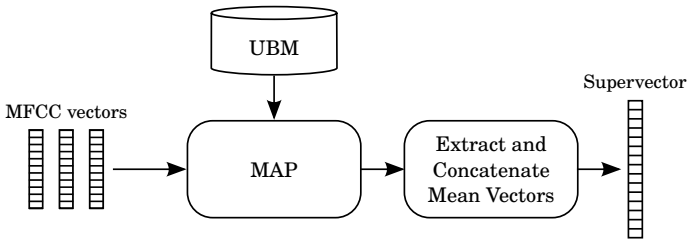


Figure 4.2: Block scheme for extracting a gaussian mean supervector from MFCCs and a trained UBM.

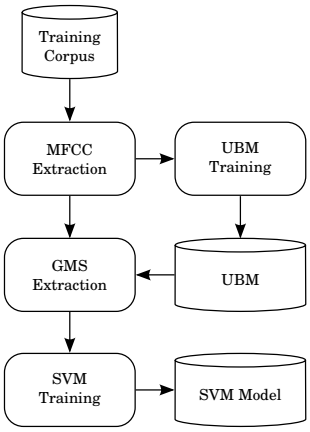


Figure 4.3: Block scheme for training the UBM and SVM models.

ture Model, GMM). The GMM is trained using the Expectation Maximization (EM) algorithm [51] on a large corpus of acoustic events. Then, for each acoustic event class in the training corpus a GMS is calculated by adapting the UBM with the Maximum A Posteriori (MAP) algorithm [52] and concatenating the adapted GMM mean values. The block diagram of the approach is shown in Figure 4.2. The final step of the training phase is the estimation of the SVM parameters. In this work, an SVM with a radial basis function has been used. The complete diagram of the training phase is shown in Figure 4.3.

Classification is performed by extracting the supervector from an input audio signal as in the training phase, and then determining the acoustic event class evaluating the SVM discriminant function (Figure 4.4). Since the number of classes is greater than two and SVMs are binary classifiers, the "one versus all" technique has been adopted [34]. LIBSVM [53] has been employed both in the training and testing phases of the SVM.

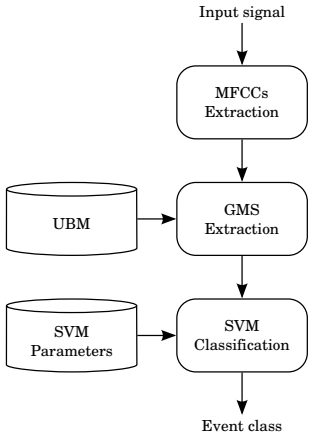


Figure 4.4: Block-scheme of the fall classification phase.

4.2 Experiments

This section firstly presents presents which portion of the dataset described in Section 3.2 has been used in this work, then the classification performance of the algorithm.

4.2.1 Dataset

All the experiments have been conducted data related to R0 room only. In particular, in order to compare the performance of the FAS with a standard aerial microphone, not all data concerning all the aerial microphones array have been used but only the one of the microphone closest to the wall (see Figure 3.5). Moreover, signals have been downsampled to 16 kHz and the resolution has been reduced to 16 bit. In order to obtain a balanced dataset, for each object and for each distance, 11 fall events have been randomly picked for a total of 44 events per object, while all the 44 simulated human fall with the manikin has been selected. In addition, the three musical track recored in the R0 room have been employed to create a noisy version of the dataset by digitally adding a random segment of the recorded musical tracks to the "clean" datasets as was done for the analysis of the signals in Section 3.2.3. In particular, the analysis presented in Section 3.2.3 suggested the authors that the standard MFCC extraction pipeline can be modified in order to better exploit the properties the floor acoustic sensor. In particular, the analysis led to the following considerations:

- The pre-emphasis filter is an inheritance of automatic speech recognition systems, where the input signal is the human voice. Since the effect of the filter is to enhance the high frequency components of the signal

Table 4.1: Data related to R0 room used in this work.

Classes	**Nr. of occurrences**
Basket	44
Fork	44
Ball	44
Book	44
Bag	44
Chair	44
Manikin Doll	44
Backgrounds	**Total length (s)**
Classic Music	882
Rock Music	616

and the floor sensor is more sensitive to low frequencies, removing the pre-emphasis could improve the classification performance.

- The majority of the energy of the signals acquired with the floor acoustic sensor is concentrated at frequencies below 1.5 kHz. This suggest that introducing a low-pass filter that removes low SNR frequencies may improve the classification performance.

In Table 4.1 the data used for this approach are summarized.

4.2.2 Experimental setup

The MFCC extraction pipeline has been firstly parametrised as described in Section 4.1.1, then, based on the study presented in the previous section, two additional pipelines have been tested:

- no pre-emphasis (NOPRE): the pre-emphasis stage has been removed from the feature extraction pipeline;
- no pre-emphasis plus low-pass filtering (NOPRELP): the pre-emphasis stage has been removed and the maximum frequency of the mel filterbank has been set to 4 kHz.

The standard MFCC extraction pipeline described in Section 4.1.1 will be denoted with "STD" in the results.

Regarding the UBM, it has been trained until convergence with the EM algorithm setting the threshold value to 10^{-3}. The same value has been employed in the MAP adaptation algorithm, but the maximum number of iterations was set to 5.

Due to the limited amount of data, the experiments have been conducted with the leave-one-label-out method: fall events recorded at all distances but one were employed for training and validation, while the remaining for testing. The validation phase consisted in searching for the number of components of the UBM and for the SVM hyperparameters which yielded the best results. In particular, the number of components of the UBM assumed the values $\{1, 2, \ldots, 64\}$, while the SVM hyperparameters C and γ the values $\{2^{-5}, 2^{-3}, \ldots, 2^{15}\}$ and $\{2^{-15}, 2^{-13}, \ldots, 2^{3}\}$ respectively.

The performance was evaluated both on clean data and on data corrupted with the music interference. In particular, the system was evaluated in matched condition, where the acoustic scenario of the training, validation and test data is the same, in mismatched condition, where training data is clean and testing data is corrupted with music, and in multi-condition, where training data and testing data contain both clean signals and corrupted signals. In the latter case, the training, validation and test sets have been divided so that they contain 1/3 of clean data and 2/3 of noisy data.

The performance has been evaluated in terms of precision (P), recall (R) and F_1-Measure (F) per class and the values have then been averaged. The metrics definitions for class i is:

$$P_i = \frac{C(i,i)}{\sum_j C(j,i)}, \tag{4.4}$$

$$R_i = \frac{C(i,i)}{\sum_j C(i,j)}, \tag{4.5}$$

$$F_i = \frac{2P_iR_i}{P_i + R_i}, \tag{4.6}$$

where $C = [C(i,j)]$ is the confusion matrix.

4.2.3 Results in matched condition

Figure 4.5 shows the results obtained in matched condition. As shown in the figure, the FAS exceeds the F_1-Measure of the aerial microphone both with clean and noisy signals regardless the feature setup employed. In particular, in clean conditions the aerial microphone achieves the highest F_1-Measure with standard MFCCs, while the FAS with the NOPRELP ones, which results in 6.50% absolute improvement. In noisy conditions, the aerial microphones best performance is again achieved with STD features, while the FAS one is achieved with the NOPRE features, with the NOPRELP setup performing almost the same. The absolute improvement of the FAS respect to the aerial microphone is 5.36%.

Focusing on the FAS performance with the various feature extraction pipelines,

Table 4.2: Aerial microphone confusion matrix related to the STD configuration in clean condition. The average precision is 91.48%, the average recall 91.23%, and the average F_1-Measure 91.04%.

	Doll	**Bag**	**Ball**	**Basket**	**Book**	**Chair**	**Fork**	**F_1**
Doll	93.18	0	0	0	2.27	4.55	0	90.11
Bag	0	86.36	0	0	13.64	0	0	82.61
Ball	0	0	100.00	0	0	0	0	100.00
Basket	2.27	0	0	97.73	0	0	0	98.85
Book	0	22.73	0	0	77.27	0	0	78.16
Chair	11.36	0	0	0	4.55	84.09	0	89.16
Fork	0	0	0	0	0	0	100	100.00
Precision	87.27	79.17	100.00	100.00	79.07	94.87	100.00	

Table 4.3: FAS confusion matrix related to the NPRELP configuration in clean condition. The average precision is 98.13%, the average recall 98.05%, and the average F_1-Measure 98.06%.

	Doll	**Bag**	**Ball**	**Basket**	**Book**	**Chair**	**Fork**	**F_1**
Doll	100.00	0	0	0	0	0	0	100.00
Bag	0	97.73	0	0	2.27	0	0	97.73
Ball	0	0	100.00	0	0	0	0	100.00
Basket	0	0	0	95.45	2.27	2.27	0	97.67
Book	0	2.27	0	0	97.73	0	0	94.51
Chair	0	0	0	0	4.55	95.45	0	96.55
Fork	0	0	0	0	0	0	100.00	100.00
Precision	100.00	97.73	100.00	100.00	91.49	97.67	100.00	

the highest F_1-Measure is obtained by removing the pre-emphasis and low-pass filtering the signals (NOPRELP) in clean conditions. Notice, however, the standard pipeline performs almost the same, thus in this condition the influence is minimal. The sole removal of the pre-emphasis, on the other hand, is detrimental for the classification performance. The opposite occurs in noisy condition, where the removal of the pre-emphasis improves the results by 3.99%. Low-pass filtering, on the other hand, does not give further improvements. Overall, the feature pipeline that gives the highest F_1-Measure is the NOPRELP, as argued in Section 3.2.3.

Further insights on the results are given by the confusion matrices of the aerial microphone (Table 4.2) and of the FAS (Table 4.3) for the feature pipeline giving the highest F_1-Measure. With the only exception of the basket falls, the FAS is able to achieve superior performance for all the objects in the dataset. In particular, the doll F_1-Measure improves by 9.89% respect to the aerial microphone. The object exhibiting the lowest performance regardless the sensor is the book: this can denote a limit in the algorithm suggesting that further improvements could be obtained by properly modifying the features or the classifier.

4.2.4 Results in mismatched condition

Figure 4.6 shows the results obtained in mismatched condition, i.e., with training and validation performed on clean data and testing on noisy data. As expected, both the performance of the aerial microphone and of the floor sensor decrease. Notice, however, that regardless the features employed, the floor sensor is able to achieve considerably higher F_1-Measures respect to the aerial microphone. In particular, the floor sensor highest F_1-Measure, obtained with the NPRE features, exceeds the one of the aerial microphone obtained with the standard MFCC pipeline by 8.76%. This value confirms that the higher SNR of the floor sensor signals actually results in better classification performance.

Regarding the features, the hypothesis that the pre-emphasis could be detrimental for the performance of the sensor is confirmed: indeed, the highest F_1-Measure has been obtained with the NOPRE configuration and exceeds the one of the standard MFCC pipeline by 1.32%. Differently, the introduction of the low-pass filter does not result in better performance respect to the NOPRE case.

4.2.5 Results in multicondition

Figure 4.7 shows the results obtained in multicondition, i.e., when training and test sets both contain clean and noisy data. The results further confirm the superiority of the FAS respect to the aerial microphone, with the first reaching 90.82% using the NOPRE features and the latter reaching 85.28% using the standard MFCC pipeline. Regarding the features, Figure 4.7 confirms that removing the pre-emphasis indeed improves classification results, while introducing the low-pass filtering does not give further benefits.

4.2.6 Final remarks

The experimental results taken as a whole provide important insights on the system behavior and allow us to express possible future directions for its improvement. In particular, the experiments demonstrated the superiority of the floor acoustic sensor compared to the aerial microphone. The results in matched condition are notable, since the F_1-Measure is greater than 98% in clean condition and close to 90% in noisy condition. The experiment in matched condition is important to compare the performance of the two sensors, but it puts the classifier in a favourable scenario. The mismatched condition is closer to a real scenario, where the classifier is trained on a dataset whose characteristics differ from testing ones. The floor sensor still achieves higher results compared to the aerial microphone, but respect to the matched condition they are considerably lower. This suggests that there is room for improvement both on the sensor

side and on the algorithmic side.

Regarding the sensor, the insertion of isolating material in the enclosure would further reduce the impact of external interferences. On the algorithmic side, different low-level features could be employed instead of MFCCs. In particular, recalling the works on speech recognition systems, Power Normalized Cepstral Coefficients (PNCCs) [54] exhibited good performance in noisy conditions and could be considered also for fall classification. In addition, the SVM classifier here employed has not been designed to recognize a specific class. In the latter case, e.g., focusing on the human fall class, the problem can be tackled in two ways: as a two class problem, where one class is the class of interest and the other comprises "the rest of the world" that will be addressed in Section 4.3 or as a novelty detection problem, where the "novel" events are represented by the specific class occurrence that will be addressed in Section 5.1. In both cases, the classification task is simpler, since the number of classes to discriminate is reduced and the overall performance is expected to increase [34].

4.3 Binary SVM based classifier for human fall detection

The classification algorithm is based on the previous work. It employs both low-level features, i.e., MFCCs [55], and high-level features, i.e., Gaussian Mean Supervectors which are then used by a Support Vector Machine to distinguish falls from no-falls. Despite MFCCs were originally developed for speech and speaker recognition tasks, they have been successfully applied also for acoustic event classification [56] and fall detection [16]. Differently from the algorithm presented in the previous section, where the algorithm discriminated falls of general objects, here the focus is specifically on the classification of human falls. The classifier, thus, has been designed to discriminate between two classes: human falls and generic sound events. In order to assess the performance of the approach, the dataset described in Section 4.2.1 has been augmented with instances of everyday sounds (speech, footsteps, etc.), thus making the task more challenging. As a reference, results employing the original dataset (Section 4.2.1) are also reported.

The classification algorithm is composed of two main parts. The first is the features extraction phase where we have extract the Mel Frequency Cepstral Coefficients from all audio files that comprise the dataset. For doing this, the feature extraction pipeline is the same used in Section 4.1.1. In particular, the signal is segmented in frames 16 ms long overlapped by 8 ms, the parameter α of the pre-emphasis filter has been set to 0.97 and the number of filters which

compose the filterbank has been set to 29. At the end, after the Discrete Cosine Transform, 13 statics coefficients are extracted that, together with their first and second derivatives, form the final feature vector of a signal.

The classification phase is similar to that one adopted in Section 4.1.2: first it uses a mixture of gaussians (GMM), trained on a large corpus of audio events with the Expectation Maximization algorithm to model the acoustic space (Universal Background Model, UBM). Then, for each audio segment, the Maximum a Posteriori (MAP) algorithm is used to calculate the Gaussian Mean Supervector (GMS) from the MFCCs. In contrast with the previous work Section 4.1.2, where we used a multi-class approach, here we employ a binary SVM to discriminate the class "fall" from "rest" which allows to distinguish human falls from the other types of sounds. In addition, the class decision is usually performed by evaluating the sign of the SVM discriminative function, which ultimately consists in deciding whether in example belongs to a class by setting a threshold equal to zero. However, in a human fall classification task, it is important to minimize the probability of missing a fall event, i.e., false negatives. In order to push the system towards this direction, we decided to consider the entire value of the SVM discriminative function, and then to set an appropriate threshold in order to minimize the occurrence of false negatives.

4.3.1 Dataset

As can be seen in the Table 4.4, the dataset used for training and assess the proposed method is an enlarged version of the one used in Section 4.1. Moreover, in order to further stress the system, a 20 minutes long recording session in which 2 persons have produced everyday noises such as talking, walking, dragging chairs, and playing with the ball was used. Then the track obtained in this session has been divided in 665 sub-tracks. The lengths of these sub-tracks have been randomly generated with gaussian distribution. Mean value and standard deviation of this distribution have been calculated based on the lengths of the other files that form the dataset. In addition to the clean dataset, a noisy version has been created to assess the performance in noisy condition. The noisy dataset consists in a musical background recorded with both sensors and digitally added to the clean events.

4.3.2 Experiments

In this section, the experimental procedure is firstly discussed and then the algorithm performance is presented. In the experiments, the signals of the dataset described above have been downsampled to 8 kHz and the bit depth has been reduced to 16 bit. Both the choice of the sampling frequency and the choice of MFCCs are justified by the analysis performed in the Section 3.2.3,

Table 4.4: Data related to R0 room used in this work.

Classes	**Nr. of occurrences**
Basket	44
Fork	44
Ball	44
Book	44
Bag	44
Chair	44
Manikin Doll	44
Backgrounds	**Total length (s)**
Classic Music	882
Rock Music	616
human activities	665

where it was shown that the signals recorded with the FAS have the majority of the energy concentrated at low frequencies (below 1 kHz). Indeed, the mel scale used in the feature extraction pipeline has a higher resolution at low frequencies, that allows to better describe the portion of the spectrum where the majority of the energy resides. In addition, the algorithm has been evaluated using two features extraction pipelines:

- the first is the same described in the Section **??** and will be denoted as STD;
- the second pipeline does not include the pre-emphasis filter and will be denoted with NOPRE.

The experiments have been conducted with a 4-fold cross-validation strategy and a three-way data split in three different operating conditions:

- matched, where the training, validation and test sets share the same acoustic condition, i.e., clean or noisy;
- mismatched, where the training set is composed of clean signals while the validation and test sets are composed of noisy signals;
- multicondition, where the training, validation and test sets contain both clean and noisy data. In this case the sets have been divided so that they contain 1/3 of clean data and 2/3 of noisy data.

The tests have been conducted also without using the signals of everyday noises, i.e., using the same dataset employed in Section 4.2.1. The performance is evaluated in terms of F_1-Measure per class and the values have then

been averaged to obtain a single performance metric. To better describe the algorithm behavior, we have used the Detection Error Trade-off (DET) curve, as defined in [58]. This graph allows evaluating the false alarm probability when miss probability is equal to 0 (henceforward named as FPM0), which is particularly relevant in a fall classification task.

Results in matched condition

Figure 4.8 shows the results obtained in the matched condition case study using the enlarged version of the dataset. In Figure 4.8a, the FAS achieves higher F_1-Measures both in clean and noisy conditions. In the first case, the floor sensor exceeds the F_1-Measure of the aerial microphone obtained with the standard MFCC pipeline by 0.11%. In the noisy condition case study, the FAS with STD features achieves an F_1-Measure greater than 0.3% with respect to the aerial microphone with NOPRE features.

The DET curves are shown in Figure 4.8b. Note that the lines relative to the FAS in clean condition are both absent. This is because independently from the threshold, either the miss probability or the false alarm probability is 0 and the DET curves assume values towards minus infinite (in logarithmic scale). This means that the FPM0 are 0 for both these configurations, while the lower FPM0 for the aerial microphone in clean condition is 1.3%. Regarding the tests with noisy signals, the performance difference between the two sensors increases, since the FPM0 is equal to 2.2% with the FAS (NOPRE) and is equal to 12.5% with the aerial sensor (STD).

In order to facilitate the analysis, Table 4.5 summarises the results obtained with the dataset version used in Section 4.2.1, which does not comprise everyday noises. As it can be observed, with respect to the previous results, the performance decreases in all tests and for both the F_1-Measure and the FPM0, although the gap between the FAS and the aerial microphone increases.

Results in mismatched condition

The mismatched condition is the most difficult case for the classifier. As expected, the performance decrease for both sensors. Focusing on Figure 4.9a, the best F_1-Measure is obtained with the NOPRE pipeline for both the FAS and aerial microphone, and is equal to 99.14% and 98.43% respectively. A consistent performance difference between the two sensors can be observed in Figure 4.9b, where the smallest FPM0 for the FAS, obtained with NOPRE, is around 13% while for the aerial one is 95%, this time obtained with STD.

The results of the mismatched experiments obtained with the dataset without everyday noises are reported in Table 4.7a. Regarding the F_1-Measure, a performance decrease can be observed compared to the results in Figure 4.9a.

Table 4.5: Fall classification performance in matched condition with the dataset excluding everyday noises.

		STD		NOPRE	
		F_1-Measure	FPM0	F_1-Measure	FPM0
Clean	Aerial	96.29	9.85	93.11	4.95
	FAS	98.81	0.00	100.00	0.00
Noisy	Aerial	97.22	43.00	96.92	73.00
	FAS	99.63	6.05	99.62	9.85

Table 4.6: Fall classification performance in mismatched condition (a) and multicondition (b) with the dataset excluding everyday noises. F_1 denotes the F_1-Measure.

(a)

%	STD		NOPRE	
	F_1	FPM0	F_1	FPM0
Aerial	96.20	87.50	95.44	76.00
FAS	97.80	34.50	98.11	12.50

(b)

%	STD		NOPRE	
	F_1	FPM0	F_1	FPM0
Aerial	96.80	16.50	96.78	31.4
FAS	99.62	0.00	99.43	0.00

Differently, the aerial microphone FPM0 improves, but it is again below the FPM0 achieved by the FAS with NOPRE MFCCs (12.5%).

Results in multicondition

Figure 4.10 shows the results in the multicondition case. The FAS superiority is confirmed, since it achieves an F_1-Measure equal to 99.78% regardless the feature extraction pipeline, while the aerial sensor obtains an F_1-Measure equal to 99.19% with the STD pipeline.

Regarding the DET plot (Figure 4.10b), an FPM0 equal to 2.9% is achieved by the FAS, but differently from the previous cases, with the STD feature. The aerial microphone achieves an FPM0 equal to 9.8% with the NOPRE pipeline.

In the last table (Table 4.7b) are shown the result of the multicondition tests obtained by using the smaller dataset. Again there is an overall decrease for the F_1-Measure, greater for the aerial microphone, while the FAS exhibiting a FPM0 equal to 0% contrary to the increasing FPM0 for the aerial sensor.

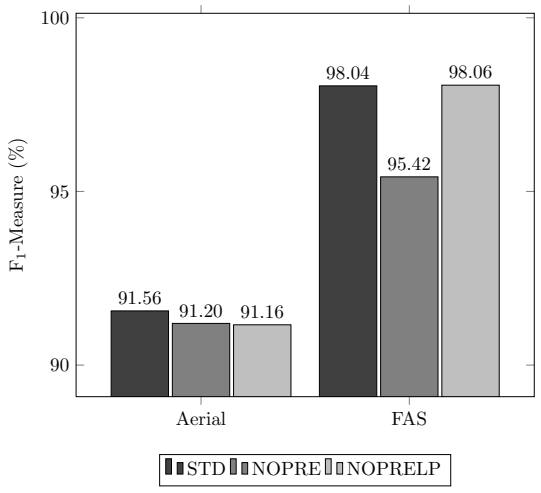
4.3.3 Final remarks

In this section, a human fall classification system based on the previous work discussed in Section 4.1 has been described. The main sensor used in the FAS the sensor operates similarly to stethoscopes, with a microphone embedded in a resonant enclosure and a membrane in contact with the floor that captures the acoustic waves resulting from a fall. The performance of that sensor has been compared with the one obtained by a standard aerial microphone. The classification algorithm extracts MFCC features from the signal acquired with the FAS, and then discriminates a fall from a generic event by using GMM supervectors and an SVM classifier. Differently from the works discussed in Section 4.1, here we specifically addressed the human fall classification task by designing the classifier to discriminate human falls from other events.

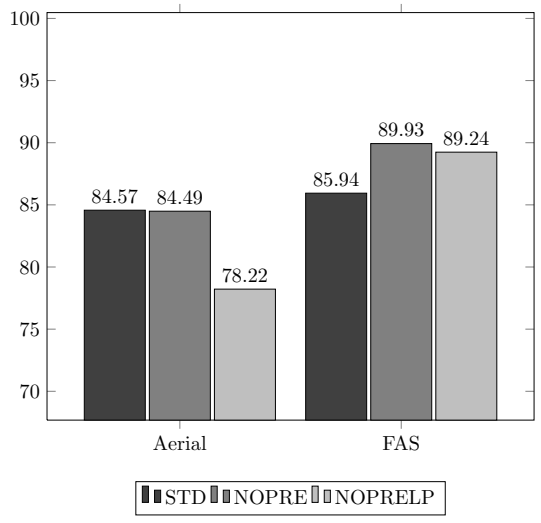
The performance of the system has been evaluated on a corpus containing recordings of several events: falls of a human mimicking doll, falls of common objects and everyday noises (speech, footsteps, etc.). In order to assess the performance of the solution in adverse acoustic conditions, a noisy version of the dataset has been created. The experiments have been performed in three operating conditions: matched, mismatched and multicondition, and the performance has been evaluated in terms of average F_1-Measure and false alarm probability when the miss probability is equal to 0. The superiority of the FAS resulted evident in all the addressed conditions, in particular with an F_1-Measure equal to 100% and an FP0 equal to 0% in clean matched conditions, and an F_1-Measure equal to 99.14% and an FP0 equal to 13% in noisy mismatched conditions.

Moreover, by looking at the results obtained with the two different features pipeline, the FAS based solution always show robust performance and it is difficult to determine which represents the best choice. The increasing performance (F_1-Measure) obtained with the largest version of dataset, in truth, are due to the fact that signals corresponding to the everyday noises are extremely easy to classify being very different from a fall.

As future works, the datasets will be expanded in order to evaluate the system performance in case the falls occurs in a different room respect to the FAS one or in different scenario as falls occurs in presence of furniture or with a different paving. Both this aspect will be addressed in Section 6.3. In addition, the detection of the time boundaries of the fall event will be addressed and techniques originally developed for enhancing speech will be evaluated to increase the robustness to acoustic distortions [**?**, 59].



(a)



(b)

Figure 4.5: Fall classification performance in matched condition with clean (a) and noisy signals (b).

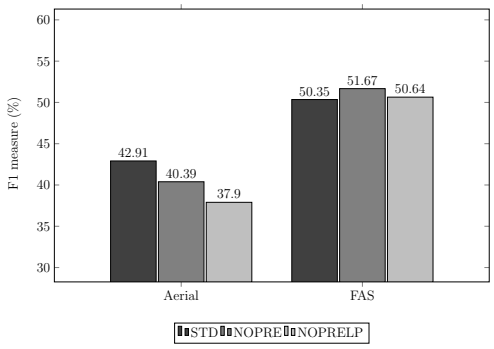


Figure 4.6: Fall classification performance in mismatched condition.

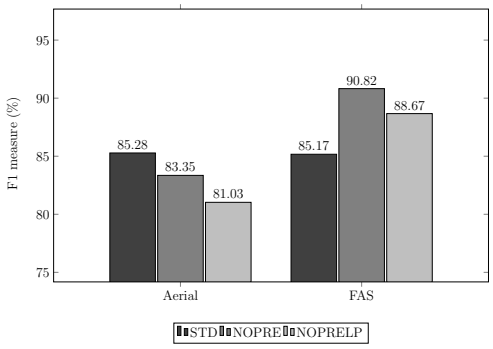
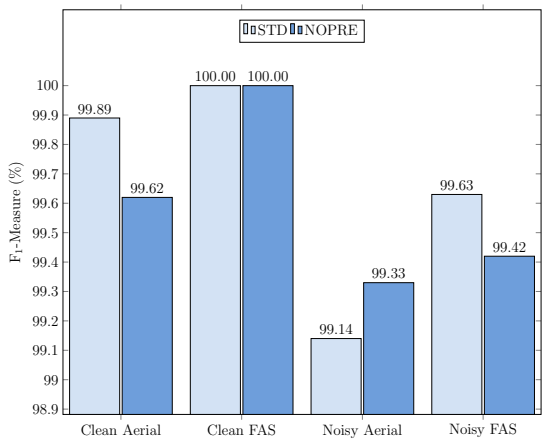
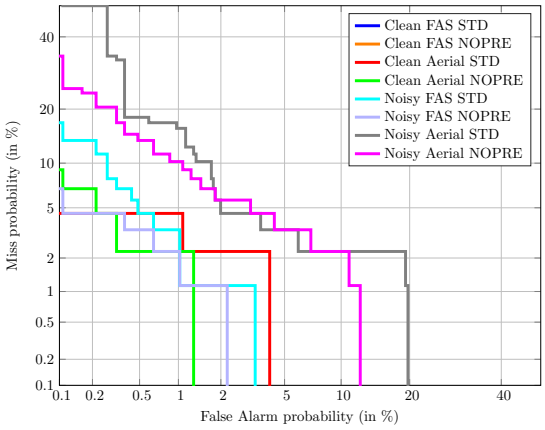


Figure 4.7: Fall classification performance in multicondition.

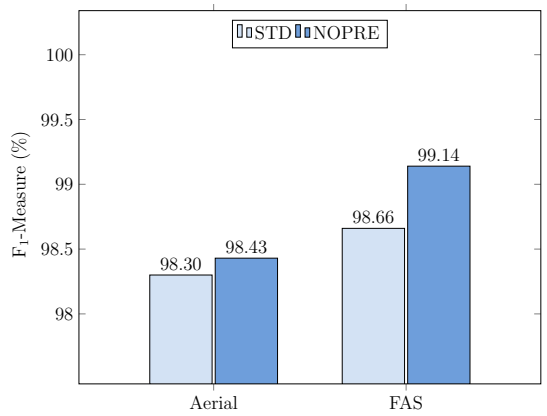


(a) F_1-Measure histogram plot.

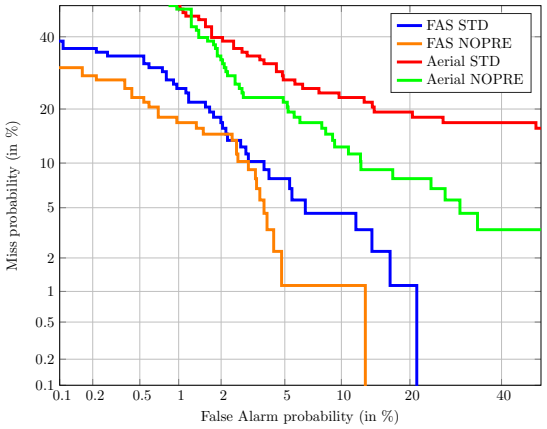


(b) Comparison of DET graphs.

Figure 4.8: Fall classification performance in matched condition with the dataset comprising everyday noises.

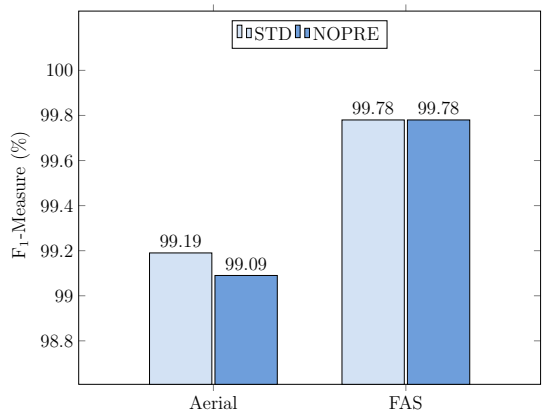


(a) F_1-Measure histogram plot.

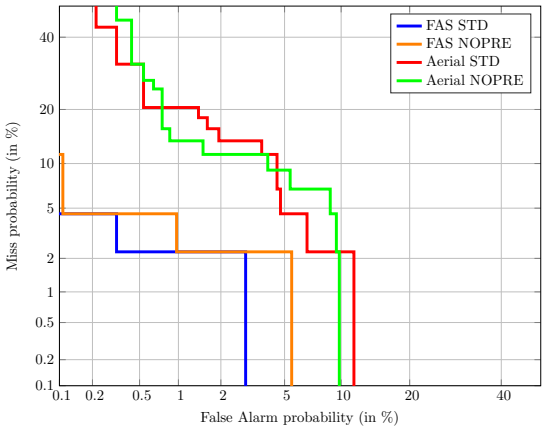


(b) Comparison of DET curves.

Figure 4.9: Fall classification performance in mismatched condition with the dataset comprising everyday noises.



(a) F_1-Measure histogram plot.



(b) Comparison of DET curves.

Figure 4.10: Fall classification performance in multicondition with the dataset comprising everyday noises.

Chapter 5

Unsupervised Approach

Autoencoder wirn 2017 + Qua veine presentato il metodo solo OCSVM

5.1 Novelty detection algorithm for human fall detection based on One-Class Support Vector Machine

5.2 End-To-End Unsupervised Approach employing Convolutional Neural Network Autoencoders for Human Fall Detection

Chapter 6

Semi-Unsupervised Approach

modifica user-aided(CIN) siamese semplice siamese autoencdoer

6.1 A Combined One-Class SVM and Template Matching Approach for User-Aided Human Fall Detection

6.2 Few-shot Siamese Neural Networks employing Audio features for Human-Fall Detection

6.3 Audio Metric Learning by using Siamese Autoencoders for One-Shot Human Fall Detection

Chapter 7

Other contributions

List of Publications

[1] E. Principi, D. Droghini, S. Squartini, P. Olivetti, and F. Piazza, "Acoustic cues from the floor: a new approach for fall classification," *Expert Systems with Applications*, pp. 51–60. vol. 60. Elsevier, 2016.

[2] D. Droghini, E. Principi, S. Squartini, P. Olivetti, and F. Piazza, "Human Fall Detection by Using an Innovative Floor Acoustic Sensor," *Proc. of WIRN*, Vietri sul Mare, Italy. May, 18-20, 2016.

[3] E. Principi, D. Droghini, S. Squartini, P. Olivetti, and F. Piazza, "A Combined One-Class SVM and Template Matching Approach for User-Aided Human Fall Detection by Means of Floor Acoustic Features," *Computational Intelligence and Neuroscience*, Hindawi, 2017.

[4] D. Droghini, D. Ferretti, E. Principi, S. Squartini, and F. Piazza, "An End-To-End Unsupervised Approach employing Convolutional Neural Network Autoencoders for Human Fall Detection," *Proc. of WIRN*, Vietri sul Mare, Italy. June, 14-16, 2017.

[5] L. Gabrielli, C. E. Cella, F. Vesperini, D. Droghini, E. Principi, and S. Squartini, "Deep learning for timbre modification and transfer: An evaluation study," *Proc. of 144th AES*, Milan, Italy, 24-26 May 2018, Audio Engineering Society.

[6] F. Vesperini, D. Droghini, E. Principi, L. Gabrielli, and S. Squartini, "Hierarchic ConvNets framework for rare sound event detection," *Proc. of EUSIPCO*. IEEE, Sept. 3-7 2018.

[7] D. Droghini, F. Vesperini, E. Principi, S. Squartini, and F. Piazza, "Few-shot siamese neural networks employing audio features for human-fall detection," *Proc. of The International Conference on Pattern Recognition and Artificial Intelligence*, Union, NJ, USA, Aug. 15-17 2018.

[8] E. Principi, D. Droghini, S. Squartini, P. Olivetti, and F. Piazza, "A Combined One-Class SVM and Template Matching Approach for User-Aided Human Fall Detection by Means of Floor Acoustic Features," *Engineering Applications of Artificial Intelligence*, Elsevier, 2018. Submitted.

Others:

[1] F. Vesperini, D. Droghini, D. Ferretti, E. Principi, L. Gabrielli, S. Squartini, and F. Piazza, "A hierarchic multi-scaled approach for rare sound event detection," Ancona, Italy, 2017, DCASE Tech. Report. Copyright-free.

[2] L. Gabrielli, F. Vesperini, D. Droghini, and S. Squartini, "Rima Glottidis: Experimenting generative raw audio synthesis for a sound installation," *XXII Colloquium of Musical Informatics*, Udine, Italy, 20-23 Nov. 2018.

Bibliography

[1] "Eurostat statistic explained: Fertility statistics," March 2016.

[2] G. Carone and D. Costello, "Can europe afford to grow old?," *Finance and Development*, vol. 43, no. 3, pp. 28–31, 2006.

[3] P. Dawadi, D. Cook, and M. Schmitter-Edgecombe, "Automated cognitive health assessment from smart home-based behavior data," *IEEE Journal of Biomedical and Health Informatics*, vol. 20, no. 4, pp. 1188–1194, 2016.

[4] E. Principi, S. Squartini, R. Bonfigli, G. Ferroni, and F. Piazza, "An integrated system for voice command recognition and emergency detection based on audio signals," *Expert Systems with Applications*, vol. 42, no. 13, pp. 5668–5683, 2015.

[5] M. Mubashir, L. Shao, and L. Seed, "A survey on fall detection: Principles and approaches," *Neurocomputing*, vol. 100, pp. 144–152, 2013.

[6] "Department of Health and Human Services: World's older population grows dramatically," `http://www.who.int/en/news-room/fact-sheets/detail/falls`, [Online; accessed 30-Oct-2018].

[7] "World Healt Organization: Falls," `http://www.who.int/en/news-room/fact-sheets/detail/falls`, [Online; accessed 30-Oct-2018].

[8] S. S. Khan and J. Hoey, "Review of fall detection techniques: A data availability perspective," *Medical engineering and physics*, vol. 39, pp. 12–22, 2017.

[9] N. Lapierre, N. Neubauer, A. Miguel-Cruz, A. R. Rincon, L. Liu, and J. Rousseau, "The state of knowledge on technologies and their use for fall detection: A scoping review," *International journal of medical informatics*, 2017.

[10] N. Pannurat, S. Thiemjarus, and E. Nantajeewarawat, "Automatic fall monitoring: a review," *Sensors*, vol. 14, no. 7, pp. 12900–12936, 2014.

[11] T. Xu, Y. Zhou, and J. Zhu, "New advances and challenges of fall detection systems: A survey," *Applied Sciences*, vol. 8, no. 3, pp. 418, 2018.

[12] N. El-Bendary, Q. Tan, F. C. Pivot, and A. Lam, "Fall detection and prevention for the elderly: A review of trends and challenges.," *International Journal on Smart Sensing & Intelligent Systems*, vol. 6, no. 3, 2013.

[13] Q. Wu, Y. D. Zhang, W. Tao, and M. G. Amin, "Radar-based fall detection based on doppler time–frequency signatures for assisted living," *IET Radar, Sonar & Navigation*, vol. 9, no. 2, pp. 164–172, 2015.

[14] T. Liu, H. Yao, R. Ji, Y. Liu, X. Liu, X. Sun, P. Xu, and Z. Zhang, "Vision-based semi-supervised homecare with spatial constraint," in *Pacific-Rim Conference on Multimedia*. Springer, 2008, pp. 416–425.

[15] F. Werner, J. Diermaier, S. Schmid, and P. Panek, "Fall detection with distributed floor-mounted accelerometers: An overview of the development and evaluation of a fall detection system within the project ehome," in *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2011 5th International Conference on*. IEEE, 2011, pp. 354–361.

[16] Y. Zigel, D. Litvak, and I. Gannot, "A method for automatic fall detection of elderly people using floor vibrations and sound—proof of concept on human mimicking doll falls," *IEEE Trans. Biomed. Eng.*, vol. 56, no. 12, pp. 2858–2867, 2009.

[17] Y. Li, K. Ho, and M. Popescu, "A microphone array system for automatic fall detection," *IEEE Trans. Biomed. Eng.*, vol. 59, no. 5, pp. 1291–1301, 2012.

[18] M. Popescu, Y. Li, M. Skubic, and M. Rantz, "An acoustic fall detector system that uses sound height information to reduce the false alarm rate," in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*. IEEE, 2008, pp. 4628–4631.

[19] M. S. Khan, M. Yu, P. Feng, L. Wang, and J. Chambers, "An unsupervised acoustic fall detection system using source separation for sound interference suppression," *Signal processing*, vol. 110, pp. 199–210, 2015.

[20] X.-X. Zhang, H. Liu, Y. Gao, and D. H. Hu, "Detecting abnormal events via hierarchical dirichlet processes," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2009, pp. 278–289.

[21] M. Popescu and A. Mahnot, "Acoustic fall detection using one-class classifiers," in *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*. IEEE, 2009, pp. 3505–3508.

[22] E. E. Stone and M. Skubic, "Fall detection in homes of older adults using the microsoft kinect," *IEEE J. Biomed. Health Inform.*, vol. 19, no. 1, pp. 290–301, 2015.

[23] A. Bourke, J. O'Brien, and G. Lyons, "Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm," *Gait and Posture*, vol. 26, no. 2, pp. 194–199, 2007.

[24] Y. Charlon, N. Fourty, W. Bourennane, and E. Campo, "Design and evaluation of a device worn for fall detection and localization: Application for the continuous monitoring of risks incurred by dependents in an Alzheimer's care unit," *Expert Systems with Applications*, vol. 40, no. 18, pp. 7316–7330, 2013.

[25] A. Özdemir and B. Barshan, "Detecting Falls with Wearable Sensors Using Machine Learning Techniques," *Sensors*, vol. 14, no. 6, pp. 10691–10708, 2014.

[26] M. Lustrek, H. Gjoreski, N. Gonzalez Vega, S. Kozina, B. Cvetkovic, V. Mirchevska, and M. Gams, "Fall detection using location sensors and accelerometers," *Pervasive Computing*, vol. 14, no. 4, pp. 72–79, Oct 2015.

[27] M. Alwan, P. J. Rajendran, S. Kell, D. Mack, S. Dalal, M. Wolfe, and R. Felder, "A smart and passive floor-vibration based fall detector for elderly," in *Proc. of Inf. Commun. Technol.*, 2006, vol. 1, pp. 1003–1007.

[28] A. Yazar, F. Keskin, B. U. Töreyin, and A. E. Çetin, "Fall detection using single-tree complex wavelet transform," *Pattern Recognition Letters*, vol. 34, pp. 1945–1952, 2013.

[29] X. Zhuang, J. Huang, G. Potamianos, and M. Hasegawa-Johnson, "Acoustic fall detection using Gaussian mixture models and GMM supervectors," in *Proc. of ICASSP*, Taipei, Taiwan, Apr. 19-24 2009, pp. 69–72.

[30] L. Liu, M. Popescu, M. Skubic, and M. Rantz, "An automatic fall detection framework using data fusion of Doppler radar and motion sensor network," in *Proc. of the 36th International Conference of the Engineering in Medicine and Biology Society (EMBC)*, 2014, pp. 5940–5943.

[31] C. N. Doukas and I. Maglogiannis, "Emergency fall incidents detection in assisted living environments utilizing motion, sound, and visual perceptual components," *IEEE Trans. Inf. Technol. Biomed.*, vol. 15, no. 2, pp. 277–89, Mar. 2011.

[32] B. Toreyin, A. Soyer, I. Onaran, and E. Cetin, "Falling person detection using multi-sensor signal processing," *EURASIP Journal on Advances in Signal Processing*, vol. 2008, no. 1, pp. 7, 2008.

[33] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[34] C. Bishop, *Pattern Recognition and Machine Learning*, Springer Science+Business Media, LLC, New York, 2006.

[35] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt, "Support vector method for novelty detection," in *Advances in neural information processing systems*, 2000, pp. 582–588.

[36] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt, "Support vector method for novelty detection," in *Advances in Neural Information Processing Systems*. 2000, vol. 12, pp. 582–588, MIT Press.

[37] S. Boyd and L. Vandenberghe, *Convex optimization*, Cambridge University Press, 2004.

[38] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, Oct. 1986.

[39] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov 1998.

[40] I. Goodfellow, H. Lee, Q. V. Le, A. Saxe, and A. Y. Ng, "Measuring invariances in deep networks," in *Advances in Neural Information Processing Systems 22*, Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, Eds., pp. 646–654. Curran Associates, Inc., 2009.

[41] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. Platt, and T. Hoffman, Eds., pp. 153–160. MIT Press, 2007.

[42] E. C. E. G. S. G. S. Spinsante, "Tst fall detection dataset v1," 2016.

[43] B. Kwolek and M. Kepski, "Human fall detection on embedded platform using depth maps and wireless accelerometer," *Computer methods and programs in biomedicine*, vol. 117, no. 3, pp. 489–501, 2014.

[44] I. Charfi, J. Miteran, J. Dubois, M. Atri, and R. Tourki, "Optimized spatio-temporal descriptors for real-time fall detection: comparison of support vector machine and adaboost-based classification," *Journal of Electronic Imaging*, vol. 22, no. 4, pp. 041106, 2013.

[45] A. T. Özdemir and B. Barshan, "Detecting falls with wearable sensors using machine learning techniques," *Sensors*, vol. 14, no. 6, pp. 10691–10708, 2014.

[46] E. Casilari, J.-A. Santoyo-Ramón, and J.-M. Cano-García, "Analysis of public datasets for wearable fall detection systems," *Sensors*, vol. 17, no. 7, 2017.

[47] M. Vacher, S. Bouakaz, M.-E. B. Chaumon, F. Aman, R. A. Khan, S. Bekkadja, F. Portet, E. Guillou, S. Rossato, and B. Lecouteux, "The cirdo corpus: Comprehensive audio/video database of domestic falls of elderly people," in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, N. C. C. Chair), K. Choukri, T. Declerck, S. Goggi, M. Grobelnik, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk, and S. Piperidis, Eds., Paris, France, may 2016, European Language Resources Association (ELRA).

[48] P. Olivetti, "Sistema per la rilevazione e prevenzione di caduta anziani, mediante cassa di risonanza a pavimento," Italian Patent 0001416548, July 1, 2015.

[49] A. Temko, C. Nadeu, D. Macho, R. Malkin, and M. Omologo, "Acoustic Event Detection and Classification," in *Hum. Comput. Interact. XXII, Comput. Hum. Interact. Loop*, W. Waibel and R. Stieflhagen, Eds., chapter 7, pp. 61–73. Springer-Verlag, Berlin, 2009.

[50] T. Kinnunen and H. Li, "An overview of text-independent speaker recognition: From features to supervectors," *Speech Communication*, vol. 52, no. 1, pp. 12–40, 2010.

[51] J. A. Bilmes, "A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models," Tech. Rep. ICSI-TR-97-021, University of Berkeley, 1997.

[52] D. Reynolds and T. Quatieri, "Speaker verification using adapted gaussian mixture models," *Digital Signal Process.*, vol. 10, no. 1, pp. 19–40, 2000.

[53] C.-C. Chang and C.-J. Lin, "LIBSVM: a library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 27, 2011.

[54] C. Kim and R. M. Stern, "Power-normalized coefficients (PNCC) for robust speech recognition," in *Proc. of ICASSP*, Kyoto, Japan, Mar. 2012, pp. 4101–4104.

[55] S. B. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 28, no. 4, pp. 357–366, 1980.

[56] A. Temko and C. Nadeu, "Classification of acoustic events using SVM-based clustering schemes," *Pattern Recognition*, vol. 39, no. 4, pp. 682–694, Apr. 2006.

[57] E. Principi, D. Droghini, S. Squartini, P. Olivetti, and F. Piazza, "Acoustic cues from the floor: a new approach for fall classification," *Expert Systems with Applications*, vol. 60, pp. 51–61, 2016.

[58] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki, "The DET curve in assessment of detection task performance," in *Proc. of the European Conference on Speech Communication and Technology*, 1997, pp. 1895–1898.

[59] S. Cifani, E. Principi, C. Rocchi, S. Squartini, and F. Piazza, "A multichannel noise reduction front-end based on psychoacoustics for robust speech recognition in highly noisy environments," in *Proc. of Hands-Free Speech Communication and Microphone Arrays (HSCMA)*, Trento, Italy, May 6-8 2008, pp. 172–175.