ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Dottorato di Ricerca in Monitoraggio e Gestione delle Strutture e dell'Ambiente (SEHM^2)

Dipartimento di Ingegneria dell'Eenergia Elettrica e dell'Informazione "Gulgielmo Marconi" - DEI

MACHINE LEARNING FOR STRUCTURAL MONITORING AND ANOMALY DETECTION

Settore Concorsuale: 09/F2 - Telecomunicazioni
Settore Scientifico Disciplinare: ING-INF/03 - Telecomunicazioni

Presentato da:
ELIA FAVARELLI

Coordinatore Dottorato:
Prof. Ing.
ALESSANDRO MARZANI

Supervisore:
Prof. Ing.
ANDREA GIORGETTI

Co-supervisore:
Prof. Ing.
ALESSANDRO MARZANI

CICLO XXXIII
ESAME FINALE ANNO 2021

ALMA MATER STUDIORUM
UNIVERSITY OF BOLOGNA

Ph.D. Programme in Structural and Environmental Health Monitoring and Management (SEHM^2)

Department of Electrical, Electronic, and Information Engineering "Gulgielmo Marconi" - DEI

MACHINE LEARNING FOR STRUCTURAL MONITORING AND ANOMALY DETECTION

09/F2 - Telecommunications
ING-INF/03 - Telecommunications

Ph.D. Candidate:
ELIA FAVARELLI

Ph.D. Coordinator:
Prof. Eng.
ALESSANDRO MARZANI

Advisor:
Prof. Eng.
ANDREA GIORGETTI

Co-advisor:
Prof. Eng.
ALESSANDRO MARZANI

CYCLE XXXIII
FINAL EXAM YEAR 2021

Keywords

Machine Learning

Deep Learning

Neural Network

Anomaly Detection

Structural Health Monitoring

Abstract

Autonomous structural health monitoring (SHM) of a large number of structures became a topic of paramount importance for maintenance purposes and safety reasons in the last few decades. Civil infrastructures are the backbone of modern society, and the assessment of their conditions is of renowned importance. This aspect is even more exacerbated because of the existing systems, such as bridges, that are fast approaching their service life. Since the replacement of those structures is functionally and economically demanding, maintenance and retrofitting operations must be planned wisely.

Moreover, edge computing, a key part of the upcoming 6G technologies, will offer cloud applications whilst providing more resources and reduced latency. This paradigm is grounded on mobile application-specific computations between the cloud, the data-producing devices, and the network infrastructure components at the edges of wireless and wired networks. The increasing amount and variety of data generated by users and sensors interconnected to the future 6G network requires new strategies to manage several types of data with highly different characteristics and also requires solutions to power the wireless network with renewable energies [1–7]. In the last few years, the rise of Internet of things (IoT) showed us how pervasive and widespread the applications of electronic devices and intelligent sensors could be, and now heading towards 6G, the idea of Internet of everything (IoE) has become anything but visionary [8–13].

As far as bridge monitoring is concerned, some statistics highlight the relevance of the problem. For example, in Italy, there are almost 2000 bridges that require special monitoring; in France, 4000 bridges need to be restored, and 840 are considered to be in critical conditions; in Germany, 800 bridges

are reputed at risk. According to conservative estimates, at least 1 % of the bridges are considered deficient in the entire world. Besides, historic structures and highly populated buildings need special attention in terms of monitoring.

In this scenario, the adoption of artificial intelligence (AI) and in particular machine learning (ML) strategies represents a flexible and potentially powerful solution that must be investigated. To manage the big and widespread amount of data generated by the extensive usage of multiple types of sensors, several ML techniques can be investigated, with the aim to perform data fusion and reduce the amount of data. Furthermore, the usage of anomaly detection techniques to identify potentially critical situations in infrastructures and buildings represents a topic of particular interest that still needs a significant investigation effort.

In this research activity, we provide the fundamental guidelines to perform automatic damage detection, which combines SHM strategies and ML algorithms capable of performing anomaly detection on a wide set of structures. In particular, several algorithms and strategies capable of extracting relevant features from large amounts of data generated by different types of sensors are investigated. Finally, to effectively manage such an amount of data in communication constraints, we obtained some design rules for the acquisition system for bridge monitoring for the first time.

List of Figures

List of Acronyms

ADC analog to digital converter

AI artificial intelligence

ANN auto-associative neural network

AP access point

ARMA auto-regressive moving average

BFD basic frequency domain

CSI channel-state information

DNN deep neural network

EVD eigenvalues decomposition

FA false alarm

FFT fast Fourier transform

FIR finite impulse response

GMM Gaussian mixture model

IoT Internet of things

IoE Internet of everything

ISM industrial, scientific and medical

LDA linear discriminant analysis

LOS line-of-sight

k-NN k-nearest neighbours

KPCA kernel principal component analysis

MAC modal assurance criterion

ML machine learning

MP mean phase

MPD mean phase deviation

MSE mean square error

NLL negative log likelihood

NLOS non-line-of-sight

NN neural network

OCC one class classifier

OCCNN one class classifier neural network

ODS operational deflection shape

OMA operational modal analysis

PCA principal component analysis

PSD power spectral density

RBF radial basis function

RF radio-frequency

ReLU rectified linear unit

RMSE root mean square error

RSS received signal strength

SDR software defined radio

SHM structural health monitoring

SoOp signals of opportunity

SSE sum of squares error

SSI stochastic subspace identification

SSID service set identifier

SVD singular value decomposition

SVM support vector machine

ToA time of arrival

WOSA weighted overlapped segment averaging

Contents

Chapter 1

Introduction

1.1 Scenario

Nowadays, the widespread adoption of automation and sensors in a broad set of applications (i.e., industry, telecommunication, structural monitoring, autonomous driving, crowdsensing) generated the opportunity to access a variety of data that can be used for new types of services and to increase the reliability of the existing ones (see Fig. 1.1). Moreover, the increasing number of devices per km^2 that with the 6G aims to become greater than 10 million leads to new networks paradigms that must be properly designed [18–23], and represents a tremendous source of data that must be organized and compressed.

n this scenario, where several new research areas have grown, both in data analysis and big data management, this thesis focuses on the adoption of innovative ML strategies applied to the important field of SHM.

The study of methods able to determine the state of health of a structure, and the localization and quantification of the damage, generated a vast literature, reviewed in several works [24–28]. Over the years, with the aim to perform damage detection on structures, several techniques have been developed to extract the most significant damage-sensitive features. Such techniques can be divided into model-free and model-based. In model-free methods, the only information is gathered by measurements (e.g., acceleration, temper-

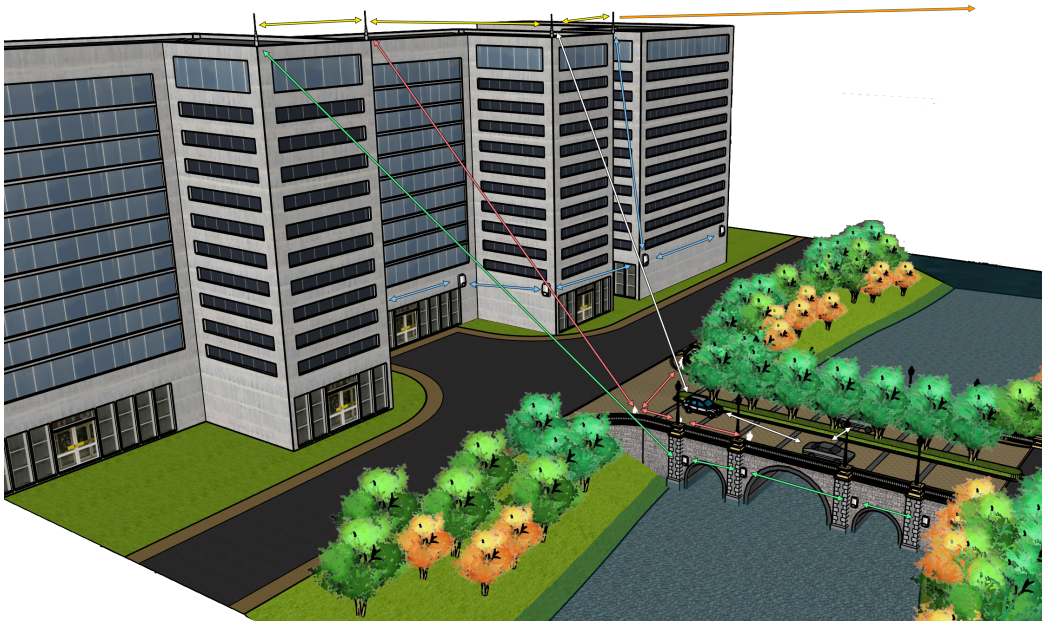

Figure 1.1: An example of wireless networks for SHM in an outdoor scenario.

ature, position), while in model-based approaches, information comes from measurements and prior knowledge of a model of the structure [14]. With the aim to develop a general strategy that can be applied to a broad set of structures and situations, this thesis is focused on the investigation of model-free methods.

Since the whole process is quite complicated and requires fine-tuning of several parameters, specifically for the structure at hand, recently, some works put forward the idea to adopt ML techniques to detect changes in the damage sensitive features [29–32]. However, despite the considerable amount of literature on SHM, much effort is still necessary to analyze data automatically and ensure the massive monitoring of a large number of infrastructures. Unfortunately, performing automatic SHM is not trivial because of the variety of structures and their peculiar characteristics, which, again, tend to require specific settings.

ML represents a comprehensive set of powerful and flexible tools that can

solve general tasks if adequately used. After the extraction of the most significant damage-sensitive features performed by SHM techniques, a wide set of ML algorithm able to perform anomaly detection on a general dataset will be presented and extensively discussed. Moreover, a collection of innovative tools with increased detection capabilities, compared to classical approaches, will be proposed and meticulously examined.

1.2 Main Contributions

In this thesis, we attempt to provide a method to automatically detect anomalies in structures starting from vibrational data. The proposed framework starts from a feature extraction phase, performed through OMA, a widely known SHM strategy, to extract the fundamental frequencies of the structure, followed by their clustering and tracking in the time domain. The processed modal frequencies are then used as features for a one class classifier (OCC) to perform anomaly detection. In particular, the main contributions of this thesis can be summarized as follows:

- We present a complete processing chain capable of monitoring the structural health of a structure autonomously using model-free system identification based on accelerometric data.
- After performing modal frequencies clustering in the stabilization diagram, we propose a new time-domain tracking algorithm based on modal frequencies density.
- We investigate strategies to select the best features among all the extracted ones, comparing feature extraction and feature selection techniques in order to provide the best solution for bridge monitoring.
- We propose a new anomaly detector, namely OCCNN, which finds the normal class (the boundary of the features space in normal operating conditions) through a two-step approach.
- We propose a variant of OCCNN, named OCCNN^2, which combines the two-step approach of OCCNN with an ANN.

- We applied the proposed framework on data collected from the *Z-24* bridge to assess its performance, and we made comparisons with several anomaly detection techniques such as PCA, KPCA, GMM, and ANN, in terms of F_1 score, accuracy, recall, and precision.
- We test the robustness and responsiveness of the proposed solution to different sensor configurations.
- We investigate the effect of sensor failures in order to provide the minimum number of sensors necessary to ensure predefined anomaly detection performance.
- We investigate the minimum observation time and the minimum number of bits that must be used to achieve the target performance, to reduce the amount of data stored for anomaly detection.

1.3 Thesis Structure and Notation

The rest of the thesis is organized as follows:

Chapter 2 presents an overview of the ML literature, considering the problems of regression, classification, dimensionality reduction, and anomaly detection, and several algorithms able to accomplish these tasks. Moreover, two novel anomaly detectors OCCNN and OCCNN^2 are proposed and extensively analyzed.

Chapter 3 investigates some strategies of SHM, whose aim is to extract damage sensitive features, with particular focus on the model-free methods. Among the several possibilities, SSI has been selected as a good candidate to solve this task because of its characteristics and wide applicability.

Chapter 4 proposes a connection between ML and SHM, the complete chain for acquisition, feature extraction, and damage detection is presented and also tested on a real structure, the *Z-24* bridge, a benchmark widely known in the literature.

Chapter 5 analyzes strategies to select the most reliable frequencies among all the extracted ones, comparing feature extraction and feature selection techniques.

Chapter 6 explores the reliability of the monitoring system, and the minimum operating conditions necessary to ensure a target performance of the anomaly detection chain, varying the number of sensors available, the acquisition time, and the number of bits per sample used to quantize the measurements.

Chapter 7 proposes a method for passive human activity classification exploiting ground vibrations observed by a biaxial geophone. The solution is grounded on the idea that some activities can be better analyzed by the horizontal channel while others by the vertical one.

Chapter 8 investigates the problem of detecting the presence of a target in a room to perform intrusion detection, evaluating the power spectral density (PSD) variation of signals of opportunity via anomaly detection techniques.

Throughout the thesis, capital boldface letters denote matrices and tensors, lowercase bold letters denote vectors, $(\cdot)^{\mathrm{T}}$ stands for transposition, $(\cdot)^{+}$ indicates the Moore-Penrose pseudoinverse operator, $||\cdot||_2$ is the ℓ_2-norm of a vector, $||\cdot||_{\mathrm{F}}$ is the Frobenius norm of a matrix, $\Re\{\cdot\}$ and $\Im\{\cdot\}$ are the real and imaginary parts of a complex number, respectively, $\mathbb{V}\{\cdot\}$ is the variance operator, and $\mathbb{1}\{a, b\}$ is the indicator function equal to 1 when $a = b$, and zero otherwise.

Chapter 2

Machine Learning Algorithms

2.1 Introduction

Recognising patterns, classifying elements, clusterize data, and perform regression from sampled data, are fundamental task to solve problems in engineering. In the last few decades many efforts have been spent to develop algorithms able to accomplish these tasks with few hyper-parameters, i.e., few parameters to be tuned.

Generally, a ML algorithm workflow is composed by a training phase and a test phase. During the training phase the algorithm is fed with a training dataset with the aim to set some parameters $\boldsymbol{\theta}$ typical of the considered algorithm in order to minimize a cost function. After that, a validation dataset is used to set some hyper-parameters $\boldsymbol{\phi}$ of the algorithm and finally a test dataset is used to evaluate the performance. More in detail, given a matrix of data $\mathbf{D}$ of dimension $N_\mathrm{d} \times D$ where N_d represents the number of observations and D stands for the dimensionality of the observation (number of features), we can define the following subsets:

- Training set $\mathbf{X}$ of dimension $N_\mathrm{x} \times D$, where N_x is the number of observations used to train the algorithm and set the parameters $\boldsymbol{\theta}$, with the aim to minimize an error function $E(\boldsymbol{\theta}, \boldsymbol{\phi})$ with a direct solution (where possible) or with some minimization strategies (i.e., stochastic descent algorithms and Newton-Raphson method).

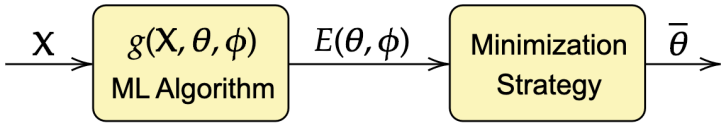

Figure 2.1: Block diagram of a general ML algorithm.

- Validation set $\mathbf{V}$ of dimension $N_\mathrm{v} \times D$, where N_v is the number of observations used to set the hyper-parameters $\boldsymbol{\phi}$ of the algorithm, in this case the hyper-parameters are set manually with trial and error strategies or with some metrics that can be defined depending on the problem.

- Test set $\mathbf{Y}$ of dimension $N_\mathrm{y} \times D$, where N_y is the number of observations used to test the algorithm performance; in this phase neither the parameters nor the hyper-parameters can be modified, in order to preserve generalization with respect to the training points.

It can be noted that the dimensionality D is constant among the sets of data and the total number of observations can be written as $N_\mathrm{d} = N_\mathrm{x} + N_\mathrm{v} + N_\mathrm{y}$. It is good practice to partition the data with the following proportions: $N_\mathrm{x} = 60\%\, N_\mathrm{d}$, $N_\mathrm{v} = 20\%\, N_\mathrm{d}$, $N_\mathrm{y} = 20\%\, N_\mathrm{d}$. When the number of observations N_d is low, some strategies like k-fold validation, cross validation, and one-vs-rest can be implemented to use a greater number of observations for the training set [29]. Another important step to apply before using the data is the normalization. Let us define the offset $\hat{\mathbf{x}}$ as the column vector containing the row-wise mean of the matrix $\mathbf{X}$, and the rescaling factor $x_\mathrm{m} = \max_{n,d} |\bar{x}_{n,d} - \hat{x}_n|$. Before proceeding with the application of ML algorithms, the matrices $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{V}$ are centered and normalized subtracting the offset $\hat{\mathbf{x}}$ row-wise and dividing each entry by the rescaling factor x_m, in this way the training data will fall in the interval $[+1, -1]$. A target matrix $\mathbf{T}(\cdot)$ is defined for the whole dataset of dimensions $N_\mathrm{d} \times D_{out}$ where D_{out} stands for the dimensionality of the output; this matrix represents the output of the observations. For instance, in a classification problem, for the training, the target matrix is actually a vector $\mathbf{t}(\mathbf{X})$ of dimension $N_\mathrm{x} \times 1$ that contains the classes to which each training point belong.

The general procedure of a ML algorithm can be summarized as shown in Fig. 2.1, and the goal is to set the parameters $\boldsymbol{\theta}$ and the hyper-parameters $\boldsymbol{\phi}$ to make a prediction $\mathbf{g}(\mathbf{X}, \boldsymbol{\theta}, \boldsymbol{\phi})$ of a target matrix $\mathbf{T}(\mathbf{X})$ in order to minimize an error function $E(\boldsymbol{\theta}, \boldsymbol{\phi})$.

2.1.1 Metrics

Several metrics can be defined to estimate the performance of a ML algorithm, usually related to the particular application. Generally, the most common metrics for regression purpose are the following:

- Sum of squares error (SSE)

$$SSE(\boldsymbol{\theta}, \boldsymbol{\phi}) = \sum_{n=1}^{N} (g(\mathbf{x}_n, \boldsymbol{\theta}, \boldsymbol{\phi}) - \mathbf{T}(\mathbf{x}_n))^2$$

- Mean square error (MSE)

$$MSE(\boldsymbol{\theta}, \boldsymbol{\phi}) = \frac{1}{N} \sum_{n=1}^{N} (g(\mathbf{x}_n, \boldsymbol{\theta}, \boldsymbol{\phi}) - \mathbf{T}(\mathbf{x}_n))^2$$

- Root mean square error (RMSE)

$$RMSE(\boldsymbol{\theta}, \boldsymbol{\phi}) = \frac{1}{\sqrt{N}} \sqrt{\sum_{n=1}^{N} (g(\mathbf{x}_n, \boldsymbol{\theta}, \boldsymbol{\phi}) - \mathbf{T}(\mathbf{x}_n))^2}$$

- Negative log likelihood (NLL)

$$NLL(\boldsymbol{\theta}, \boldsymbol{\phi}) = -\sum_{n=1}^{N} g(\mathbf{x}_n, \boldsymbol{\theta}, \boldsymbol{\phi}) \ln \mathbf{T}(\mathbf{x}_n).$$

In several applications a regularization term is introduced in order to make the error function convex. This term helps the minimization algorithms to converge to an optimal solution, usually written in the form $\lambda ||\boldsymbol{\theta}||_2^2$ where

λ represents the regularization hyper-parameter that belongs to the hyper-parameters set $\boldsymbol{\phi}$. For classification purposes we can define a different set of metrics, more useful to understand the performance of our algorithms:

- Accuracy
$$\mathrm{Acc} = \frac{T_\mathrm{P} + T_\mathrm{N}}{T_\mathrm{P} + T_\mathrm{N} + F_\mathrm{P} + F_\mathrm{N}}$$
- Precision
$$\mathrm{Prec} = \frac{T_\mathrm{P}}{T_\mathrm{P} + F_\mathrm{P}}$$
- Recall
$$\mathrm{Rec} = \frac{T_\mathrm{P}}{T_\mathrm{P} + F_\mathrm{N}}$$
- F_1 score
$$\mathrm{F}_1 = 2 \cdot \frac{\mathrm{Rec} \cdot \mathrm{Prec}}{\mathrm{Rec} + \mathrm{Prec}}$$

where T_P, T_N, F_P, and F_N, represent respectively true positive, true negative, false positive, and false negative predictions. Such indicators are obtained comparing the actual labels $\mathbf{t}(\mathbf{X})$, with those predicted by the ML algorithm $\mathbf{g}(\mathbf{X}, \boldsymbol{\theta}, \boldsymbol{\phi})$. In the case of unbalanced classes, the F_1 score represents a more reliable metric to evaluate the performance with respect to the accuracy. These metrics can be defined only for two-class classification problems; for multi-class classification problems, the accuracy is still valid, defined as number of correct predictions over the total number of points, and confusion matrix can be adopted to estimate the correct prediction over the different classes.

2.1.2 Minimization Strategies

Now that we have defined the most common methods to evaluate the algorithm performance, we are ready to define some generic strategies to set the weights $\boldsymbol{\theta}$ in order to minimize the selected error function. The most common strategy is the gradient descent. The basic idea with gradient descent is to build a linear model of the selected error function $E(\boldsymbol{\theta})$ (to simplify the

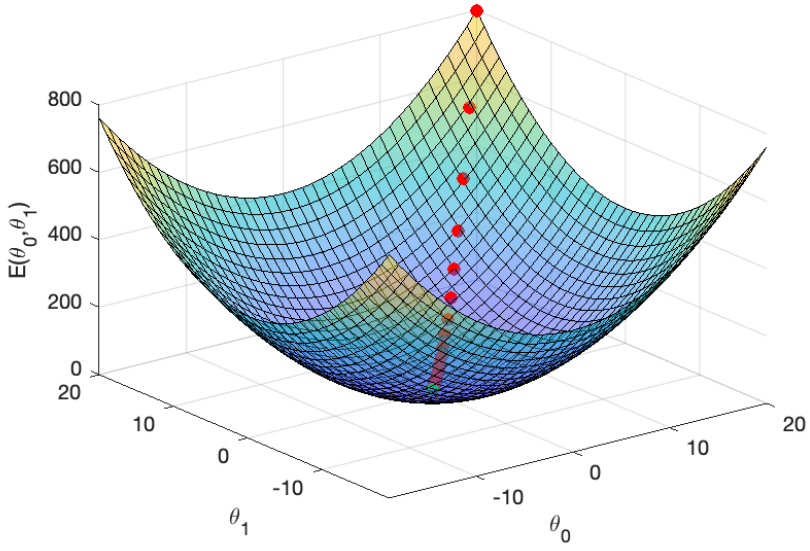

Figure 2.2: Gradient descent applied to a convex function, red points represent the iterative evaluation of the error function $E(\boldsymbol{\theta})$ due to the update of the weight vector $\boldsymbol{\theta}$.

notation we neglected $\boldsymbol{\phi}$, because it is not influent in this process), determine the downward direction on this function, travel a short distance along this direction, hop back on to the error function, and repeat until convergence. Formally, beginning at an initial configuration of weights $\boldsymbol{\theta}^{(0)}$ the linear model of $E(\boldsymbol{\theta})$ at this point is given precisely by the first order Taylor series approximation centered at $\boldsymbol{\theta}^{(0)}$:

$$\widetilde{E}(\boldsymbol{\theta}) = E(\boldsymbol{\theta}^{(0)}) + \nabla E(\boldsymbol{\theta}^{(0)})^T(\boldsymbol{\theta} - \boldsymbol{\theta}^{(0)}). \tag{2.1}$$

Now we take the first step by travelling in the direction in which the tangent hyper-plane most sharply angles downward (referred to as the steepest descent direction). It can be shown that this steepest descent direction is given precisely as $-\nabla E(\boldsymbol{\theta}^{(0)})$. Thus, we descend in the direction of the negative gradient (hence the name of the algorithm gradient descent) taking our first step to a point $\boldsymbol{\theta}^{(1)}$ where

$$\boldsymbol{\theta}^{(1)} = \boldsymbol{\theta}^{(0)} - \rho \nabla E(\boldsymbol{\theta}^{(0)}). \tag{2.2}$$

The same process can be repeated iteratively; in general the number of iteration represents the number of epochs N_e (if all the dataset is used for each iteration of the algorithm) and the hyper-parameter ρ represents the learning rate. For the kth iteration we can write the update rule as follows:

$$\boldsymbol{\theta}^{(k)} = \boldsymbol{\theta}^{(k-1)} - \rho \nabla E(\boldsymbol{\theta}^{(k-1)}). \tag{2.3}$$

The number of epochs N_e can be substituted with several convergence criteria based on the error value or the error decreasing rate [30]. Several variations of this technique have been introduced over the years to reduce the oscillation problems and increase the velocity of convergence [30]. Furthermore, when the second derivative of $E(\boldsymbol{\theta})$ can be computed, the Newton-Raphson method can be implemented to strongly increase the convergence rate [29]. An example of gradient descent algorithm applied in a two-dimensional weight vector space is shown in Fig. 2.2.

2.1.3 Example: Polynomial curve fitting

To clarify the notation, a toy regression problem is presented. For instance, observe a one-dimensional ($D = 1$) phenomena described by the following equation: $f(z) = y(z) + n = \cos(4\pi z) + 2\sin(2\pi z) + n$ where n represents a generic additive noise. Let's now observe a set of N_d acquisitions of the phenomena that form the observation matrix $\mathbf{D}$. In this example the dimensionality of the problem is equal to one, hence the observation matrix is a column vector $\mathbf{d}$ of dimension $N_\mathrm{d} \times 1$. Partitioning this vector in training set, validation set, and test set as described previously, we obtain respectively the training vector $\mathbf{x}$ of dimension $N_\mathrm{x} \times 1$, the validation vector $\mathbf{v}$ of dimension $N_\mathrm{v} \times 1$, and the test vector $\mathbf{y}$ of dimension $N_\mathrm{y} \times 1$. In this example the target vector corresponds to the collection of samples of the function output $\mathbf{t}(\mathbf{d}) = f(\mathbf{d})$, with dimension $N_\mathrm{d} \times 1$. An example of training set for the given problem, with $N_\mathrm{x} = 60$, is reported in Fig 2.3. We can now define a polynomial model, in order to implement the curve fitting. A generic

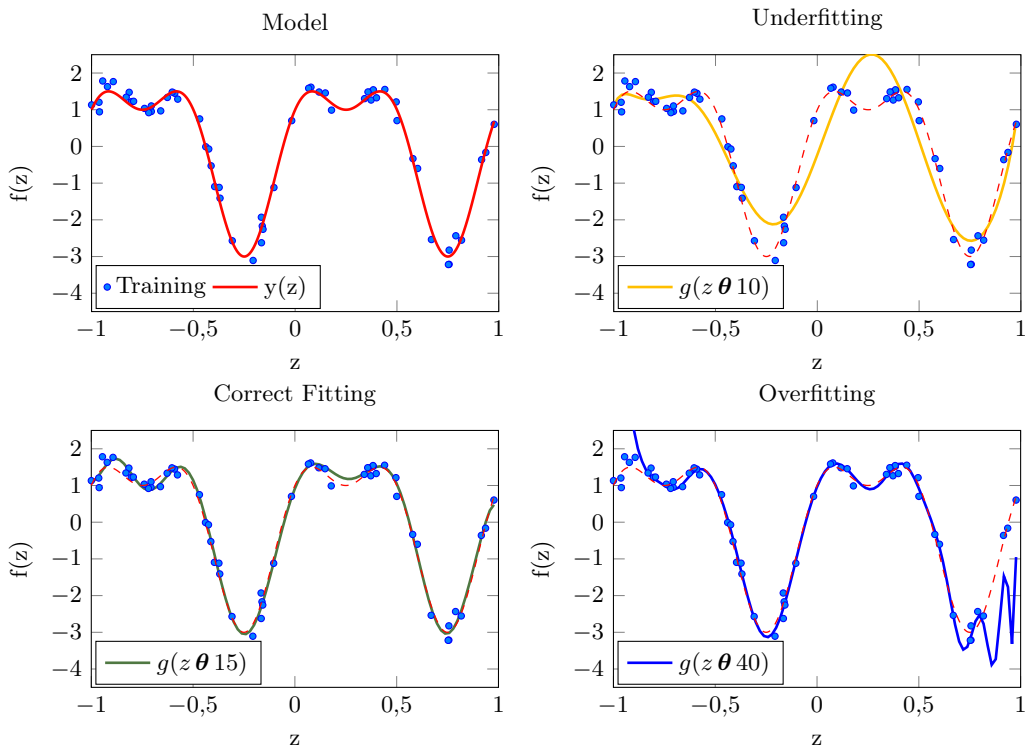

Figure 2.3: Example of polynomial fitting, with underfitting and overfitting phenomena due to different values of ϕ.

one-dimensional polynomial model can be defined as follows:

$$g(z, \boldsymbol{\theta}, \phi) = \theta_0 + \theta_1 z + \theta_2 z^2 + \cdots + \theta_\phi z^\phi = \sum_{j=0}^{\phi} \theta_j z^j$$

where ϕ represents the model order of the polynomial model, the only hyper-parameter for this model, and z is a generic point. The values of the coefficients $\boldsymbol{\theta}$ can be determined by fitting the polynomial to the training data. This can be done by minimizing an error function that measures the misfit between the function $g(\mathbf{x}, \boldsymbol{\theta}, \phi)$, for any given value of $\boldsymbol{\theta}$, and the training set data point targets $\mathbf{t}(\mathbf{x})$. One simple choice of error function, which is widely used in literature, is given by the sum of the squares of the errors between the predictions $g(x_n, \boldsymbol{\theta}, \phi)$ for each data point x_n and the corresponding target

values $t(x_n)$, so that we minimize:

$$E(\boldsymbol{\theta}, \phi) = \sum_{n=1}^{N_X} (g(x_n, \boldsymbol{\theta}, \phi) - t(x_n))^2.$$

It is possible to solve the curve fitting problem by choosing the value of $\boldsymbol{\theta}$ for which $E(\boldsymbol{\theta}, \phi)$ is as small as possible. Because the error function is a quadratic function of the coefficients $\boldsymbol{\theta}$, its derivatives with respect to the coefficients will be linear in the elements of $\boldsymbol{\theta}$, and so the minimization of the error function has a unique solution, denoted by $\boldsymbol{\theta}^*$, which can be found as a direct solution. The resulting polynomial is given by the function $g(\mathbf{x}, \boldsymbol{\theta}^*, \phi^*)$. The optimal ϕ^* must be set in order to prevent overfitting and underfitting, the first one is a situation in witch the error is low over the training set but high over the test set, whereas in the second case the error is high both in the training and the test set. An example of overfitting, underfitting, and optimal configuration is illustrated in Fig 2.3 to clarify these possible situations. In the example shown, the noise presented is a Gaussian noise with zero mean and variance equal to 0.1, i.e., $n \sim \mathcal{N}(0, 0.1)$.

2.2 Neural Networks

NNs represent a powerful and flexible tool belonging to the ML algorithms. This kind of algorithm can be adopted to solve regression, classification, and dimensionality reduction problems with slightly differences in its structure, and without particular limitations [29, 31]. The general structure of a NN is reported in Fig. 2.4. As we can see in the picture the structure provides the following elements:

- Neurons, represented as circles, each one has an activation function used to propagate the input of the neuron to the next layer.
- Weights, represented as arrows, used to connect the neurons giving to them different relevance.

The layers of a neural network can be divided into the following groups:

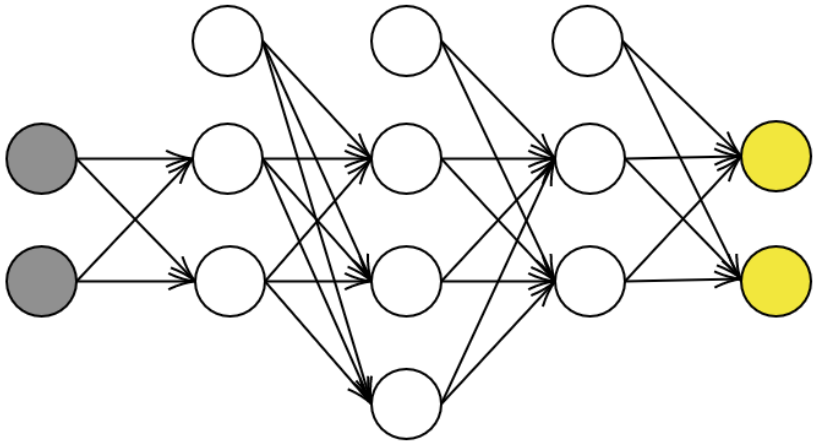

Figure 2.4: General NN structure; grey neurons represent the input layer, white neurons the hidden layers, and yellow neurons the output one.

- Input layer, equal to the number of features of the problem (D) that are used to feed the network with the dataset.
- Hidden layers, that are used to create non linear function able to follow the data function.
- Output layer, that is used to communicate the results.

In the next chapter we will see how to set the activation functions, and the number of neurons and layers for several problems. Generally, in this thesis the number of hidden layers and the relative number of neurons in each hidden layer will be represented with the vector $\mathbf{h}$ of dimension $N_\text{h} \times 1$ where N_h stands for the number of hidden layers and the elements of $\mathbf{h}$ represent the number of neurons in the relative hidden layer.

The NN provide, as all the ML algorithms, a training phase and an online phase. During the training phase the network is fed with the training dataset $\mathbf{X}$ with the training target $\mathbf{T}(\mathbf{X})$ to set the network weights $\boldsymbol{\theta}$ in order to minimize the selected error function $E(\boldsymbol{\theta}, \boldsymbol{\phi})$. To set the hyper-parameters $\boldsymbol{\phi}$ (i.e. λ, ρ, $\mathbf{h}$, and N_e) several trainings can be performed to maximize the performance on the validation set $\mathbf{V}$ and finally the real performance can be evaluated on the test set $\mathbf{Y}$. As we can infer from the previous observations, the training phase can be computationally onerous, and requires a long time

to be performed. In order to reduce the computational cost, the gradient descent algorithm is usually replaced by the backword propagation algorithm that accomplishes the same gradient descent task but with lower complexity. The description of this strategy is beyond the scope of this thesis, but detailed description can be found in [29]. The online phase (after the training phase) for the NN is faster and this represents another remarkable aspect of this kind of algorithms that are prone to be used in real time applications after a time consuming training phase.

2.2.1 Regression

As shown previously, a regression problem is a possible application for ML algorithms and particularly for NNs. In this kind of problem, the goal is to reconstruct a curve starting from a set of measurements. This type of application can be useful for:

- filtering, when the goal is to reduce the noise present on a set of measurements.
- interpolation, when the goal is to reconstruct a curve in order to estimate missing points.
- prediction, when the objective is to predict some values out of the boundaries of the given points.

The generic structure of a NN able to perform regression is reported in Fig. 2.5. For this application the number of input neurons (depicted in blue) is equal to the number of input features D, the neurons present in the hidden layers (in the example $\mathbf{h} = [3, 4, 3]$) exhibit a non linear activation function (the most common is the rectified linear unit (ReLU) function, but many other possible solutions are presented in [29, 30]). In these layers the activation functions are non-linear in order to catch non-linear dependencies among the data, and finally the output neurons (depicted in green) equal to the dimensionality of the output and adopt a linear activation function, able to follow all the possible excursion of the target $t(\mathbf{X})$. The white neurons

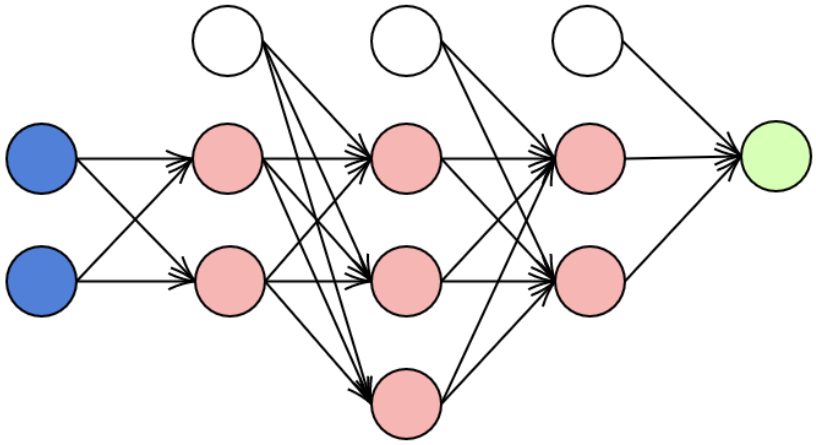

Figure 2.5: NN structure for regression, different neuron colours stays for different activation functions.

represent the bias terms, they are not connected to the previous layer and are able to follow the bias values among the data.

2.2.2 Classification

As said before, a quite similar structure can be used to perform classification on a given dataset. A classification problem is an application of NN where the target function $t(\mathbf{X})$ is discrete, and represents the class to which the points belong. In this case the number of input neurons is still equal to the number of input features D, the neurons in the hidden layers remain the same as the previous application, but the output layer has a number of neurons equal to the number of classes of the problem C. The activation function is a sigmoid with the following expression:

$$S(z) = \frac{e^z}{e^z + 1} \tag{2.4}$$

where z represents the input of the considered neuron and $S(z)$ its output. In this case a sigmoid activation function with excursion between 0 and 1 is preferred to a linear one. That is because the output of each network can be interpreted as a value related to the probability that the input point belongs to the class represented by the considered neuron, hence an excursion

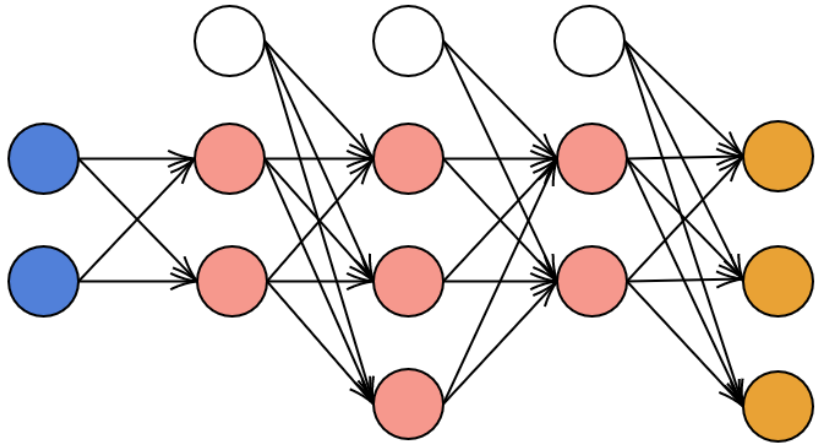

Figure 2.6: NN structure for classification, different neuron colours stands for different activation functions.

between 0 and 1 is preferred. To assign an input to the relative class, the neuron with the highest activation function is selected. An example of NN structure for classification purpose is shown in Fig. 2.6.

2.2.3 Dimensionality Reduction

The last application considered is the dimensionality reduction performed by the ANN [31]. This particular NN structure is composed by the following sections:

- Mapping layers, which consist of one or more hidden layers, with the number of neurons in each layer decreasing progressively till the last one, called bottleneck. In the bottleneck the number of neurons is usually lower than the number of input features;
- Demapping layers, composed of one or more hidden layers where the number of neurons increases progressively.

The input and output layers have the same dimension of the feature space, and the labels during the training phase must be set equal to the input data point. With this structure, the data is mapped in a lower-dimensional feature space (with dimension equal to the number of neurons present in

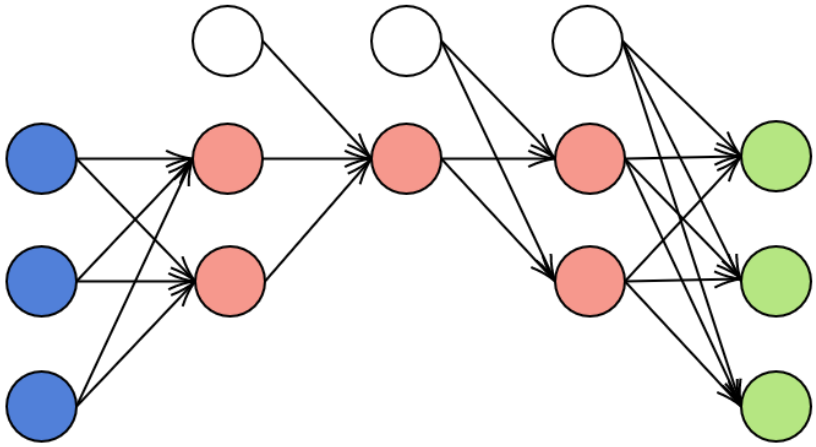

Figure 2.7: ANN structure for dimensionality reduction, different neuron colours stands for different activation functions.

the bottleneck layer), and then reconstructed through the demapping layers minimizing the error with respect to the input data. After the training phase, if we feed the network with a new data point, the activation function in the bottleneck layer outputs a mapped version of the starting data point in a lower dimensional feature space. Afterwards, we can reconstruct the data using the demapping layers with a low reconstruction error. An example of ANN is shown in Fig. 2.7.

Now, focusing on the specific problems we are investigating the following two questions arises:

Q1: *How can we use tools for dimensionality reduction to perform anomaly detection?*

Q2: *What are the most effective algorithms able to perform anomaly detection?*

2.3 Anomaly Detection

In this section we review PCA, KPCA, and GMM, which are often implemented for anomaly detection. Usually, the algorithms that implement dimensionality reduction can also be used as anomaly detectors. One class

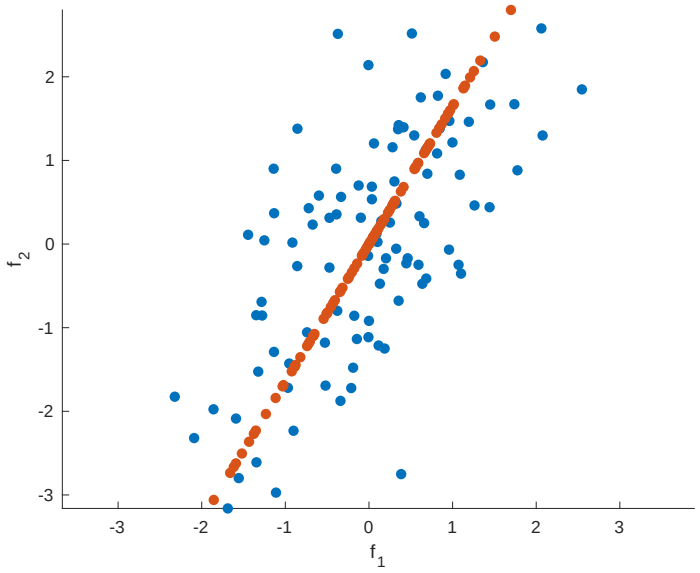

Figure 2.8: Example of PCA; blue points represent the starting points in the original feature space, the projected ones are depicted in red.

classification or anomaly detection is a particular case of the classification problem, where the training data is present only for one class (the standard one), the algorithm is trained only with this class, and the goal is to detect whether a new point belongs to the same class or is an anomaly [33–40]. To use a dimensionality reduction technique as anomaly detector, it is enough to evaluate the reconstruction error between the input data and the reconstructed ones. Then, through a threshold, that can be set in order to ensure a certain false alarm rate on the training set, evaluate if a new point experiments a reconstruction error greater than the threshold: in this case it is considered as anomalous, otherwise it is considered standard. These strategies can be also used with ANN which represents an important and useful tool for anomaly detection.

2.3.1 Principal component analysis

This technique remaps the training data from the feature space $\mathbb{R}^D$ in a subspace $\mathbb{R}^P$ (where $P < D$ is the number of components selected) that

minimizes the error (defined as Euclidean distance) between the data in the feature space and their projection in the selected subspace [41]. More in detail, to find the best subspace over which to project the training data, we need to evaluate the $D \times D$ sample covariance matrix

$$\mathbf{\Sigma}_{\mathrm{x}} = \frac{\mathbf{X}^{\mathrm{T}}\mathbf{X}}{N_{\mathrm{x}} - 1}. \tag{2.5}$$

By eigenvalue decomposition $\mathbf{\Sigma}_{\mathrm{x}}$ can be factorized as $\mathbf{\Sigma}_{\mathrm{x}} = \mathbf{V}_{\mathrm{x}}\mathbf{\Lambda}_{\mathrm{x}}{\mathbf{V}_{\mathrm{x}}}^{\mathrm{T}}$, where $\mathbf{V}_{\mathrm{x}}$ is an orthonormal matrix whose columns are the eigenvectors, while $\mathbf{\Lambda}_{\mathrm{x}}$ is a diagonal matrix that contains the D eigenvalues. The eigenvalues magnitude represents the importance of the direction pointed by the relative eigenvector. The projection into the subspace is obtained by multiplying the data with $\mathbf{V}_{\mathrm{P}}$, i.e., $\mathbf{X}_{\mathrm{P}} = \mathbf{X}\mathbf{V}_{\mathrm{P}}$, $\mathbf{y}_{\mathrm{P}} = \mathbf{Y}\mathbf{V}_{\mathrm{P}}$, and $\mathbf{u}_{\mathrm{P}} = \mathbf{U}\mathbf{V}_{\mathrm{P}}$ where $\mathbf{V}_{\mathrm{P}}$ represent the P greatest eigenvector selected among the D extracted.

To evaluate the error between the projected points and the starting ones it is necessary to reconstruct the data in the original feature space (i.e. $\widetilde{\mathbf{X}} = \mathbf{X}_{\mathrm{P}}\mathbf{V}_{\mathrm{P}}^{\mathrm{T}}$, $\widetilde{\mathbf{Y}} = \mathbf{Y}_{\mathrm{P}}\mathbf{V}_{\mathrm{P}}^{\mathrm{T}}$, and $\widetilde{\mathbf{U}} = \mathbf{U}_{\mathrm{P}}\mathbf{V}_{\mathrm{P}}^{\mathrm{T}}$). After the reconstruction, it is possible to evaluate the error as the Euclidean distance between the original data and the reconstructed data. An example of PCA applied on a set of data is reported in Fig. 2.8. An important limitation of this technique is that it is able to find just linear boundaries in the original feature space.

2.3.2 Kernel principal component analysis

In many cases, the linear boundaries found by PCA represent a severe limitation [42]. KPCA firstly maps the data with a non-linear function, named kernel, then applies the standard PCA to find a linear boundary in the new feature space. Such boundary becomes non-linear in the original feature space. A crucial point in KPCA is the selection of the kernel that leads to linearly separable data in the new feature space. In [43], when data distribution is unknown, the RBF kernel is proposed as good candidate to accomplish this task. Given a generic point $\mathbf{z}$ that corresponds to a $1 \times D$ vector, we can

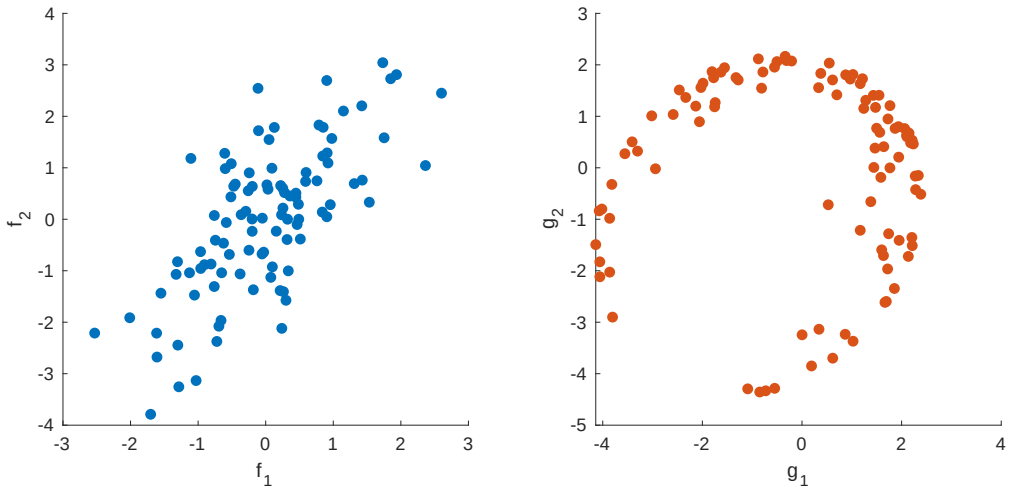

Figure 2.9: Example of KPCA; on the left the points in the original feature space are depicted, on the right the same points are projected in a new feature space through RBF.

apply the RBF as

$$K_n^{(\mathbf{z})} = e^{-\gamma||\mathbf{z}-\mathbf{x}_n||^2} \quad \text{with} \quad n = 1, 2, \ldots, N_\text{x} \tag{2.6}$$

where γ is a kernel parameter (which controls the width of the Gaussian function) that must be set properly, $\mathbf{x}_n$ is the nth row of $\mathbf{X}$, and $K_n^{(\mathbf{z})}$ is the nth component of the point $\mathbf{z}$ in the kernel space. Overall, the vector $\mathbf{z}$ is mapped in the vector $\mathbf{k}^{(\mathbf{z})} = [K_1^{(\mathbf{z})}, K_2^{(\mathbf{z})}, \ldots, K_{N_\text{x}}^{(\mathbf{z})}]$. Remapping all the data in the kernel space, we obtain the subsequent matrices $\mathbf{K}_\text{x}$ of size $N_\text{x} \times N_\text{x}$ for training, $\mathbf{K}_\text{y}$ of size $N_\text{y} \times N_\text{x}$ for validation, and $\mathbf{K}_\text{v}$ of size $N_\text{v} \times N_\text{x}$ for test, respectively. Applying now the PCA to the new data sets, it is possible to find non-linear boundaries in the original feature space. An example of KPCA with RBF kernel applied on a set of data is shown in Fig. 2.9.

2.3.3 Gaussian mixture model

Another important algorithm that can be used to solve OCC problems is the GMM [44]. This approach assumes that data can be represented by a mixture of $\mathcal{M}$ multivariate Gaussian distributions. The outputs of the

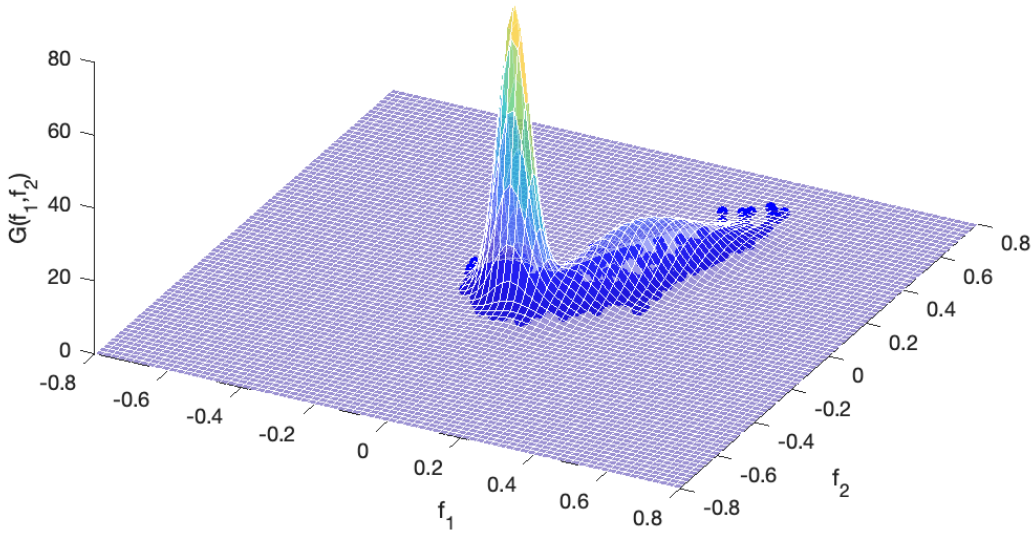

Figure 2.10: Example of GMM; blue points represent the training data, the surface represents the Gaussian mixture of order 2 that fit the data.

algorithm are the covariance matrices, $\boldsymbol{\Sigma}_m$, and the mean values, $\boldsymbol{\mu}_m$, of the Gaussian functions, with $m = 1, 2, \ldots, \mathcal{M}$. The GMM algorithm finds the set of parameters $\boldsymbol{\Sigma}_m$ and $\boldsymbol{\mu}_m$ of a Gaussian mixture that better fit the data distribution. This strategy can also be used to perform clustering on a set of data [29, 30]. An example of GMM applied on a set of data is shown in 2.10.

Now, considering the high versatility of NNs, and the need for effective OCCs, the following question arises:

Q3: *Is it possible to use classical NN structures (instead of the ANNs) to perform anomaly detection?*

2.4 OCCNN

In this section, a novel technique based on NN classifiers is presented. For the sake of clarity, this section is divided in several parts, each one describing a block of the diagram depicted in Fig. 2.11.

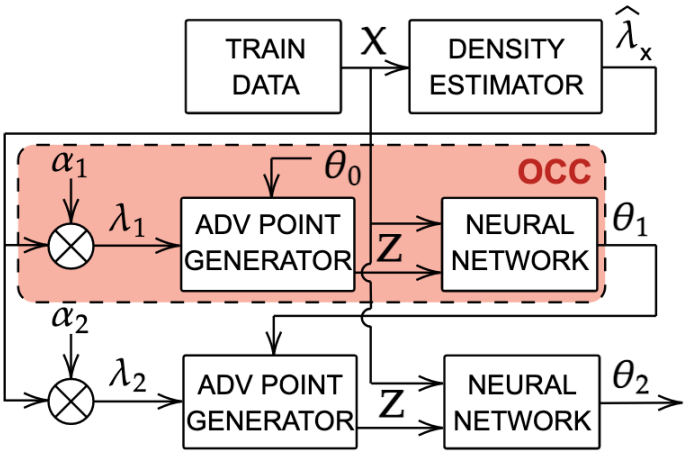

Figure 2.11: Block diagram of the OCCNN algorithm.

2.4.1 Density estimator

The first fundamental step in the classification chain is the density estimation of the training points. As proved in [45] by Pollard, a minimum-variance unbiased estimator of the density exists when the data is distributed according to a Poisson point process. Such estimator has the form [45]

$$\widehat{\lambda}_{\mathrm{x}} = \frac{\left(\sum_{n=1}^{N_{\mathrm{p}}} k_n\right) - 1}{\pi \sum_{n=1}^{N_{\mathrm{p}}} r_n^2} \tag{2.7}$$

where $\widehat{\lambda}_{\mathrm{x}}$ is the estimated density for a dataset $\mathbf{X}$, r_n is the Euclidean distance between the the n-th point and its k_n-th neighbour, and N_{p} is the number of points considered for the estimation. It is possible to show that the mean value and variance of the estimator are, respectively

$$\mathbb{E}\left\{\widehat{\lambda}_{\mathrm{x}}\right\} = \lambda_{\mathrm{x}} \tag{2.8}$$

$$\mathbb{V}\left\{\widehat{\lambda}_{\mathrm{x}}\right\} = \frac{\lambda_{\mathrm{x}}}{\left(\sum_{n=1}^{N_{\mathrm{p}}} k_n\right) - 2}. \tag{2.9}$$

From equation (2.9) it is clear that by increasing the number of points N_{p} and the value of k_n (hence considering farther points) the estimator variance

decreases and accordingly the accuracy increases. However, when the spatial distribution of data points deviates from Poisson, some countermeasures are taken especially to account for the finite boundaries of the normal class:

- the set of points from which evaluating the distances r_n is a subset of the training data $\mathbf{X}$;
- low values of k_n are selected;
- A high number of points N_p is used to increase the estimator accuracy.

More precisely, a subset of $N_\mathrm{p} = 200$ points is selected from the training set (when $\mathbf{X}$ has less points, all the training set is used for the density estimation), and for all these points the second nearest point is considered so $k_n = 2$ for $n = 1, 2, \ldots, N_\mathrm{p}$.

2.4.2 Adversarial points generator

This block solves the task to generate uniformly distributed random points with density $\lambda_i = \widehat{\lambda}_\mathrm{x}\alpha_i$ in a specific portion of the feature space defined by $\boldsymbol{\theta}_{i-1}$ where i is the iteration index. This is fundamental to create an adversarial class $\mathbf{Z}$ from which the NN can learn the boundaries between the training data $\mathbf{X}$ and the adversarial ones. The adversarial points must be generated in the portion of space where the training points are absent, hence where the output layer activation function is greater than 0.5 (we suppose to associate label 1 to the training set points and 0 to the adversarial ones). The function gets as input the estimated training point density multiplied by a factor α_i, that we will discuss in Section 2.4.4, and the NN weights matrix $\boldsymbol{\theta}_{i-1}$ that represents the network state after the previous training iteration and defines the actual estimated boundaries. At the first iteration ($i = 1$) we suppose $\boldsymbol{\theta}_0$ equal to a null matrix; this means that the adversarial points will be generated in all the feature space including in the area filled by the training points.

2.4.3 Network structure

A feed-forward NN with two hidden layers and M neurons in each layer is trained to find the boundaries between the normal class data and the adversarial ones [30]. All the layers are fully connected, and the activation functions are ReLU for the hidden layers and softmax for the output layer. The network is trained with all the training set points for N_{e} epochs with learning rate ρ. At the end of the training phase, the network provides the weight matrix $\boldsymbol{\theta}_i$ that defines the boundaries estimated at the ith iteration of the OCCNN algorithm that will be used in the next iteration by the adversarial points generator block to generate points outside the boundaries defined. Finally, at the last iteration, the boundaries estimated by the NN are the ones used for anomaly detection.

2.4.4 Dimensioning

In this section, a wide set of tests is presented with the aim to find the best configuration of hyper-parameters. The main goals are:

- testing the proposed algorithm with different numbers of training points, N_{x}, varying α_1;
- finding the value of α_2 that maximize the classification accuracy;
- comparing the OCCNN solution with PCA, KPCA, GMM, and ANN.

The OCCNN hyper-parameters are $M = 50$ neurons, $N_{\mathrm{e}} = 5000$ epochs, and the learning rate is $\rho = 0.05$. The error function for a generic training point $\mathbf{x}_n$ is defined as

$$e_{x_n} = \sqrt{\sum_{d=1}^{D} (x_{n,d} - \widetilde{x}_{n,d})^2}. \tag{2.10}$$

The same metric can also be used to evaluate the reconstruction error of all the test set points. A target false alarm rate on the training set is fixed equal to 0.01, consequently a threshold is selected for each algorithm to guarantee such constraint.

N_P \ α_1	0.01	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
1000	0.72	0.89	0.97	0.97	0.96	0.94	0.92	0.91	0.83	0.87	0.78
900	0.69	0.89	0.97	0.98	0.93	0.96	0.93	0.89	0.87	0.85	0.79
800	0.74	0.82	0.96	0.95	0.94	0.93	0.89	0.87	0.87	0.83	0.79
700	0.71	0.91	0.96	0.95	0.95	0.92	0.88	0.86	0.89	0.83	0.85
600	0.67	0.85	0.94	0.93	0.94	0.94	0.89	0.86	0.83	0.81	0.82
500	0.64	0.82	0.91	0.94	0.92	0.89	0.87	0.88	0.76	0.78	0.80
400	0.64	0.84	0.93	0.93	0.91	0.88	0.89	0.85	0.83	0.77	0.74
300	0.63	0.77	0.87	0.90	0.89	0.87	0.88	0.85	0.83	0.78	0.79
200	0.62	0.72	0.79	0.88	0.86	0.82	0.86	0.81	0.81	0.77	0.78

Figure 2.12: Average accuracy calculated varying α_1 and N_P.

Dataset size and test on α_1

The first important step is to find the value of α_1 that maximizes the accuracy of the first iteration evaluated on a validation set and checked on the test set. Validation and test sets have the same distribution, but in order to avoid generalization error it is good practice to define two different datasets [31]. The value of $N_x = N_y = N_v = N_p$ is tested between 200 and 1000, for lower values the density estimator variance becomes too high and the dataset too small to train the NN. Again, the value of α_1 is varied in the range between 0.01 and 1, higher values would generate too many adversarial points in the area of the training set and the NN would classify all the feature space as adversarial. In Fig. 2.12 a heatmap that shows the average accuracy (evaluated over the six distributions shown in Fig. 2.14) of the OCCNN algorithm varying N_p and α_1 is reported. Note that accuracy increases with the number of points in the training set and the best value for α_1 is 0.3, irrespectively of the number of training points. From now on the parameter α_1 is set to 0.3 and the number of training, test and validation points for the next simulations is fixed at 500.

Test on α_2

Another important test is the definition of the parameter α_2 that sets the adversarial point density after the first iteration. This test is executed with

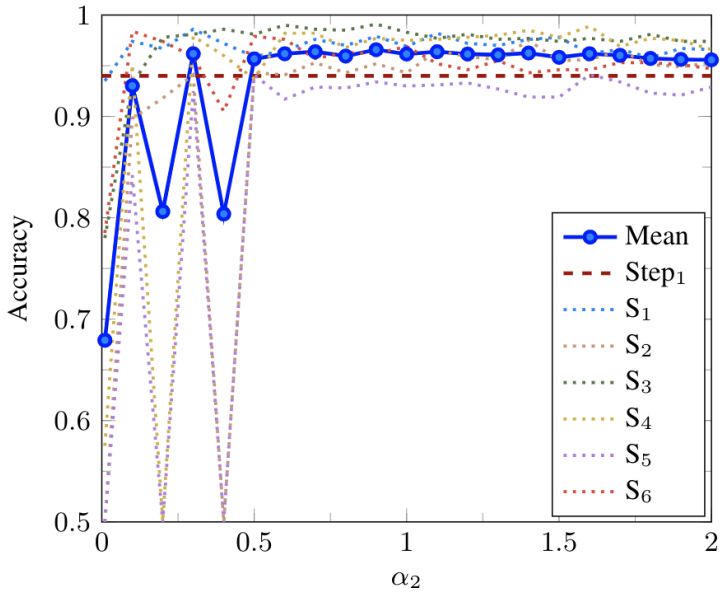

Figure 2.13: Accuracy as a function of α_2 for different distributions.

a fixed number of points in the training set, and the accuracy, when α_2 varies between 0.01 and 2, is shown in Fig. 2.13. It is important to highlight that in this case α_2 can assume values greater than 1 because in this phase of the OCCNN algorithm the boundaries are already defined, so the adversarial points will only be generated outside the boundaries estimated by the NN and defined by the weight matrix $\boldsymbol{\theta}_1$. We can see this step as a fine-tuning of the boundaries roughly estimated in the first iteration. All six curves are shown along with the average accuracy (thick blue curve). It can be noted that for values of α_2 lower than 0.6, the accuracy becomes unpredictable because the low number of adversarial points generated is not enough to define the boundaries correctly. On the other hand, increasing α_2 beyond 1.5 slightly decreases the accuracy because the boundary region tends to be shaped by the adversarial class. The value of $\alpha_2 = 0.8$ is selected to maximize the accuracy and improve the performances with respect to the first step. With these parameters, an image of the boundaries estimated by the OCCNN at the end of the two iterations is shown at the bottom of Fig. 2.14. Comparing these with the points distributions at the top of Fig. 2.14 it is possible to see

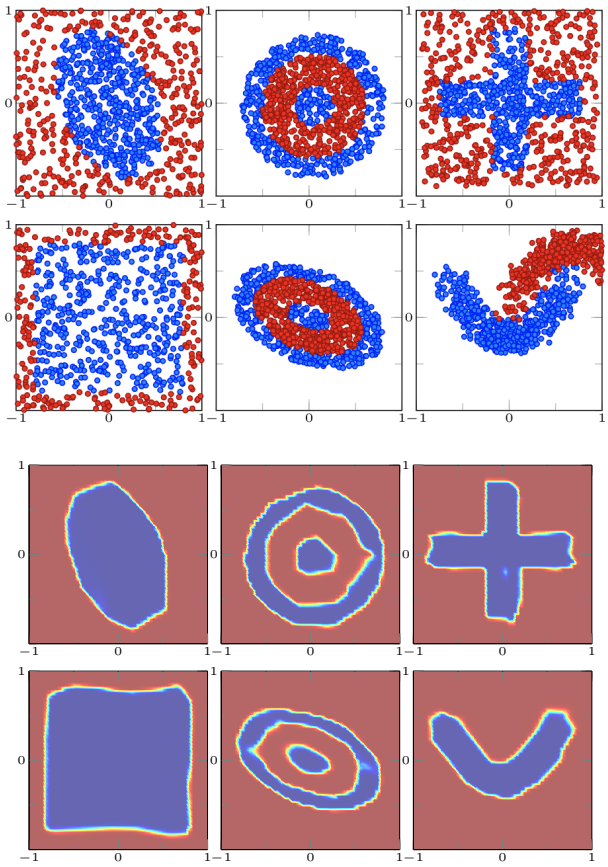

Figure 2.14: At the top some examples of normalised dataset points distribution. Blue circles denote features corresponding to normal conditions, red circles denote features corresponding to anomaly conditions. Estimated boundaries by the OCCNN algorithm are showed at the bottom.

that the estimated boundaries are really close to the real ones. Now that all the parameters are set we are ready to compare the various algorithms.

2.4.5 OCCNN^2

The OCCNN performance is rather sensitive to the density estimation; therefore, depending on the data set, its ability to detect anomalies may be dominated by this first step [36]. Moreover, the Pollard's estimator (2.7) may exhibit accuracy degradation when the data set points distribution deviates from Poisson. Based on these considerations, a possible solution is the

OCCNN^2, an anomaly detector that shares the same strategy of OCCNN only the first NN of the classification chain (the red box in Fig. 2.11) is replaced with another OCC to have a rough estimate of the normal class boundaries. As shown in Section 4.3, a good choice for this step is to use an ANN.

Now that a set of tools for detection of anomalies are available, two questions concerning the goal of this thesis arise:

Q4: *How can we perform anomaly detection based on machine learning tools to detect a damage in a structure?*

Q5: *Is there an effective technique to extract features able to highlight a damage in a structure?*

Chapter 3

Structural Health Monitoring

3.1 Introduction

In this section we present some tools that belong to the so-called OMA able to perform structural monitoring and extract damage sensitive features from some structure characteristics. Over the years several strategies have been developed to extract relevant parameters that can represent the healthy state of a structure (i.e., natural frequencies, fundamental modes, dumping ratios, etc.). These strategies can be divided in two groups:

- Model based methods, that require a simulated model of the structure, hence needing an accurate knowledge of the structure elements and materials.
- Model-free methods, that can be assumed as a blind technique not needing a detailed description of the structure.

Among the model based techniques we can find the finite element method, this strategy provides a simulated model of complex structure as a sum of basic elements that are easier to model with respect to the whole system. This strategy provides an accurate estimation of the modal parameter of the structure as long as its simulation is accurate. The main problem with this strategy is the computational cost and the impossibility to generalize it to different structures. In fact, the need of a model require a targeted

study of the particular infrastructure and cannot be generalized [14]. For this reason we decided to focus our efforts on the model-free methods; to this class belong the peak-picking method, the auto-regressive moving average (ARMA) models, and the SSI approach.

3.2 Operational Modal Analysis

The expression OMA means the class of modal identification methods based on response measurements only. This discipline has been systematized in the last two decades but early applications can be traced back to the beginning of modal testing in the 1960s.

The first applications of OMA were mainly based on the analysis of PSD and the identification of operational deflection shapes (ODSs). An ODS represents the deflection of a structure at a particular frequency under a generic input and it is usually the result of the contribution of different modes.

However, under certain assumptions, ODSs are a close estimate of the actual mode shapes. The mode shapes represent the displacement of a structure subject to a stress with respect to its repose position.

In the 1990s, a number of methods working in time domain were developed and applied in combination with correlation functions, leading to the so-called output-only modal analysis. In the same period the use of ARMA models for modal parameter estimation, first suggested in the late 1970s, started spreading. An increasing number of applications appeared in the literature but output-only modal identification was not fully developed and widely accepted as a reliable source of information, yet.

However, at the end of the 1990s new effective output-only modal identification techniques, such as the SSI, became available, overcoming the limitations of the previous techniques in dealing with closely spaced modes and noise [46].

Nowadays, OMA is a widely accepted tool for modal identification, with several successful applications in civil engineering (bridges, buildings, pedestrian bridges, historical structures, offshore platforms, wind turbines, dams,

stadia), mechanical and industrial engineering (ships, trucks, car bodies, engines, rotating machineries), aerospace engineering (in-flight modal identification of aircrafts and shuttles, studies about flutter).

OMA is based on the following assumptions:

- Linearity (the response of the system to a given combination of inputs is equal to the same combination of the corresponding outputs).
- Stationarity (the dynamic characteristics of the structure do not change over time, so that the coefficients of the differential equations governing the dynamic response of the structure are time independent).
- Observability (the sensor layout has been properly designed to observe the modes of interest, avoiding, for instance, nodal points).

Moreover, unlike the traditional modal testing where the input is controlled, OMA is based on the dynamic response of the structure under test to non-controllable and immeasurable loadings such as environmental and operational loads (traffic, wind, microtremors, and so on). As a consequence, some assumptions about the input are needed. If the structure is excited by white noise, that is to say, the input spectrum is constant, all modes are equally excited and the output spectrum contains full information about the structure [14]. However, this is rarely the case, since the excitation has a spectral distribution of its own. Modes are, therefore, weighted by the spectral distribution of the input and both the properties of the input and the modal parameters of the structure are observed in the response.

Additionally, noise and spurious harmonics due to rotating equipment are observed in the response. Thus, generally, the structure is assumed to be excited by unknown forces that are the output of the so-called excitation system loaded by white noise (see also Fig. 3.1). Under this assumption, the measured response can be interpreted as the output of the combined system, made by the excitation system and the structure under test in series, to a stationary, zero mean, Gaussian white noise.

Since the excitation system and the structure under test are in cascade, the frequency response function of the combined system is the product of

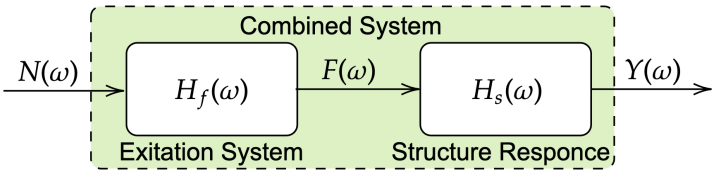

Figure 3.1: Block diagram of the OMA strategy [14].

their respective frequency response functions [14]:

$$H_c(\omega) = H_f(\omega)H_s(\omega)$$

where $H_c(\omega)$, $H_f(\omega)$, and $H_s(\omega)$ are the frequency responses of the combined system, the excitation system, and the structure under test, respectively. In fact, for each subsystem, output and input are related by the following equations [14]:

$$F(\omega) = N(\omega)H_f(\omega)$$

$$Y(\omega) = F(\omega)H_s(\omega) \tag{3.1}$$

where $N(\omega)$, $F(\omega)$, and $Y(\omega)$ denote the Fourier transforms of the white noise input to the excitation system, the excitation system output, and the structure output, respectively. In this context, the measured response includes information about the excitation system and the structure under test, but the modal parameters of the structure are preserved and identifiable, and the characteristics of the excitation system have no influence on the accuracy of modal parameter estimates.

The discrimination between structural modes and properties of the excitation system is possible since the structural system has a narrowband response and time invariant properties, while the excitation system has a broadband response and it may have either time variance or time invariance properties.

The estimation of the modal model of the structure gives the opportunity to estimate also the unknown forces, according to (3.1). The assumption of broadband excitation ensures that all the structural modes in the frequency range of interest are excited. Assuming that the combined system is excited

by a random input, the second order statistics of the response carries all the physical information about the system and plays a fundamental role in output-only modal identification. The focus on second order statistics is justified by the central limit theorem. In fact, the structural response is approximately Gaussian in most cases, regardless of the distributions of the (independent) input loads, which are often not Gaussian.

The spatial distribution of the input also affects the performance of OMA methods, especially in the presence of closely spaced modes. A random distribution of inputs in time and space provides better modal identification results. The presence of measurement noise and spurious harmonics in response to measurements requires appropriate data processing to mitigate their effects and discriminate them from actual structural modes. These strategies take the name of mode selection and will be discussed in the following section.

3.2.1 Peak-Picking

The most undemanding method for output-only modal parameter identification is the basic frequency domain (BFD) method, also known as the peak-picking method. The name of the method comes from the fact that the modes are identified by picking the peaks in the PSD plots. Given a pair of sample records $x(t)$ and $y(t)$ of finite duration T from stationary random processes, their Fourier transforms (which exist as a consequence of the finite duration of the signals) are:

$$X(f) = \int_0^T x(t)e^{-j2\pi ft}dt$$

$$Y(f) = \int_0^T y(t)e^{-j2\pi ft}dt$$

and the auto-spectral and cross-spectral density functions are defined as follows:

$$S_{xx}(f) = \int_{-\infty}^{+\infty} R_{xx}(\tau)e^{-j2\pi f\tau}d\tau$$

$$S_{yy}(f) = \int_{-\infty}^{+\infty} R_{yy}(\tau) e^{-j2\pi f\tau} d\tau$$

$$S_{xy}(f) = S_{yx}(f) = \int_{-\infty}^{+\infty} R_{xy}(\tau) e^{-j2\pi f\tau} d\tau$$

where $R_{xy}(\tau)$ represents the cross-correlation function between the signals $x(t)$ and $y(t)$. The BFD technique is based on the assumption that, around a resonance, only one mode is dominant. A reference sensor for the computation of the cross-spectral densities with all other measurement channels has to be selected to ensure that most of the modes can be observed. As a consequence, sensors close to nodes of the mode shapes cannot be adopted as reference sensors. The ideal choice for the reference sensor makes possible the identification of all the modes through the computation of a single column of the PSD matrix (the column made by the cross-spectral densities between the selected reference sensor and all other sensors).

However, depending on the geometry of the structure and the adopted sensor layout, a single reference sensor could be insufficient to identify all the modes and at least a couple of reference sensors with different orientation have to be adopted. For instance, in the case of a building-like structure characterized by two bending modes in two orthogonal directions x and y and sensors parallel to these directions, the selection of a sensor measuring along x as reference permits the identification of the bending modes in the x direction and eventually torsional modes, but it is inadequate to identify the bending modes in the y direction. This can be observed only through the selection of an additional reference sensor measuring along y [14].

However, the success of the identification process heavily depends on the geometry of the structure and the skill of the analyst. The results of modal identification suffer a certain degree of subjectivity also in the case of noisy measurements, when the peaks in the spectra are not clear. The BFD method provides local estimates of the modal properties. Moreover, the accuracy of the estimated eigenfrequencies depends on the frequency resolution of the spectra. A fine frequency resolution is fundamental to obtain good natural frequency estimates, but this can be increased with well known signal processing techniques (i.e., zero padding) [47].

In principle, the BFD should be applied to evaluate natural frequencies and mode shapes only. However, the BFD technique is effective when damping is low and modes are well separated. If these conditions are violated it may lead to erroneous results. Thus, this technique can be used only in certain circumstances, suffer of subjectivity, and does not provide an exhaustive and accurate estimation of the modal parameters. However it is computationally inexpensive.

3.2.2 Auto-regressive moving average

This technique starts with the equation of motion of a randomly excited linear time-invariant system, that can be written as follows:

$$\mathbf{M}\frac{d^2y(t)}{dt^2} + \mathbf{D}\frac{dy(t)}{dt} + \mathbf{K}y(t) = w(t) \tag{3.2}$$

where $y(t)$ stands for the measured system response, $w(t)$ is a continuous-time, zero mean Gaussian white noise, $\mathbf{M}$ represents the mass matrix, $\mathbf{D}$ the dumping matrix, and $\mathbf{K}$ the stiffness matrix [14]. It is possible to show that this system can be also described by a discrete-time ARMA vector model (the ARMA model is referred as such an ARMA vector model to point out its multivariate character) by approximating the differential operator with finite differences over a finite time step Δt.

Historically, ARMA vector models have been used for the estimation of the modal parameters of civil structures. Due to a number of shortcomings (in particular for systems with many outputs and many modes, where the large set of parameters to be estimated leads to large computational burden and convergence problems), stochastic state-space models have progressively replaced them in the context of modal identification.

In order to explain how modal parameters can be extracted from an ARMA model, assume that a continuous-time system is observed at discrete time instants k with a sampling interval Δt. Since the input on the structure is not available (it is immeasurable), the equivalent discrete-time system can be obtained only by requiring that the covariance function of its response

to a Gaussian white noise input be coincident at all discrete time lags with that of the continuous-time system. This implies that the first and second order moments of the response of the discretized model are equal to the first and second order moments of the response of the continuous-time system at all the considered discrete time instants. Under the assumption that the response of the system is Gaussian distributed, the covariance equivalent model is the most accurate approximated model, since it is exact at all discrete time lags. When the dynamic response of the system is driven by the Gaussian white noise $w(t)$ but there are also some disturbances (process and measurement noise), the latter must also to be taken into account by the equivalent discrete-time model. In the presence of such disturbances, an $ARMA(n\alpha, n\gamma)$ model has the following form:

$$\mathbf{y}_k + \mathbf{\Omega}_1 \mathbf{y}_{k-1} + \cdots + \mathbf{\Omega}_{N_\Omega} \mathbf{y}_{k-N_\Omega} = \mathbf{e}_k + \mathbf{\Gamma}_1 \mathbf{e}_{k-1} + \cdots + \mathbf{\Gamma}_{N_\Gamma} \mathbf{e}_{k-N_\Gamma} \qquad (3.3)$$

where y_k is the vector of the output at the time instant t_k, and e_k is the innovation modeled as a zero mean Gaussian white noise. The left-hand side of (3.3) is the auto-regressive part, while the right-hand side is the moving average part. The matrices $\mathbf{\Omega}_i$ contain the auto-regressive parameters, while the matrices $\mathbf{\Gamma}_j$ contain the moving average parameters; N_Ω and N_Γ represent the auto-regressive and moving average order of the model. It is possible to show that a covariance equivalent ARMA vector model can be converted into a forward innovation state space model, and vice versa. If the order of the state-space model is too large, the model will contain redundant information; on the contrary, if the state-space dimension is too small, a certain amount of information about the modelled system will be lost. This transformation provides a good estimation of the modal parameters but ARMA models usually present stability issues and are computationally expansive.

3.2.3 Stochastic subspace identification

Among all the possibilities, this strategy has been selected as the best candidate to extract modal parameters from the accelerometric measurements present for the *Z-24* bridge. In the following will be explain how this tech-

nique has been applied to the *Z-24* environment. The SSI is a time-domain, parametric, covariance-driven procedure for blind modal analysis (i.e., it can extract the modal frequencies from the accelerometric measurements without any *a priori* knowledge about the structure [14]).

SSI is characterized by the system order $n \in \mathbb{N}$ and the time-lag $i \geq 1$. To apply the algorithm it is enough to satisfy the following constraint, $l \cdot i \geq n$ [14], where l stands for the number of sensors. In the *Z-24* monitoring, the system order n is considered unknown, so it is kept as a parameter varied in the range $n \in [2, 160]$ (with step 2), while the time-lag is $i = 60$ [14].

Let us define the block Toeplitz matrix for a given time-lag i and shift s

$$\mathbf{T}_{s|i}^{(a)} = \begin{bmatrix} \mathbf{R}_i^{(a)} & \mathbf{R}_{i-1}^{(a)} & \dots & \mathbf{R}_{s+1}^{(a)} & \mathbf{R}_s^{(a)} \\ \mathbf{R}_{i+1}^{(a)} & \mathbf{R}_i^{(a)} & \dots & \mathbf{R}_{s+2}^{(a)} & \mathbf{R}_{s+1}^{(a)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{R}_{2i-1}^{(a)} & \mathbf{R}_{2i-2}^{(a)} & \dots & \mathbf{R}_{i+1}^{(a)} & \mathbf{R}_i^{(a)} \end{bmatrix} \tag{3.4}$$

of dimensions $li \times li$ where

$$\mathbf{R}_i^{(a)} = \frac{1}{N-i} \mathbf{D}_{(a,1:N-i,:)} \mathbf{D}_{(a,i:N,:)}^{\mathrm{T}} \tag{3.5}$$

is a correlation matrix of dimension $l \times l$, and the data matrix $\mathbf{D}_{(a,b:c,:)}$ is obtained by the tensor $\boldsymbol{\mathcal{D}}$ selecting a particular acquisition a from all the sensors in the time interval $[b, c]$, where tensor $\boldsymbol{\mathcal{D}}$ of dimensions $N_\mathrm{a} \times l \times N_\mathrm{s}$, contains all the data collected after a pre-processing phase that will be described in the following, with N_a number of acquisitions, l number of sensors, and N_s number of samples. For the sake of this presentation we will drop the acquisition index a, considering that all the subsequent operations are performed for each acquisition. Now, by the singular value decomposition (SVD) we can factorize the block Toeplitz matrix (3.4), with $s = 1$, as

$$\mathbf{T}_{1|i} = \mathbf{U}^{(n)} \boldsymbol{\Sigma}^{(n)} \mathbf{V}^{(n)\mathrm{T}} \tag{3.6}$$

where $\mathbf{U}^{(n)}$ and $\mathbf{V}^{(n)\mathrm{T}}$ are matrices of dimensions respectively $li \times n$ and $n \times li$ and $\boldsymbol{\Sigma}$ is a diagonal matrix, of dimension $n \times n$, that contain the singular

values arranged in descending order. For the sake of this presentation, we will drop the model order index n, considering that all the subsequent observations can be applied for each model order. The SVD allows selecting the correct number of singular values that must be considered, and therefore to split the matrix $\mathbf{T}_{1|i}$ in two parts

$$\mathbf{T}_{1|i} = \mathbf{O}_i \boldsymbol{\Gamma}_i \tag{3.7}$$

$$\mathbf{O}_i = \mathbf{U}\boldsymbol{\Sigma}^{1/2}\mathbf{S} \tag{3.8}$$

$$\boldsymbol{\Gamma}_i = \mathbf{S}^{-1}\boldsymbol{\Sigma}^{1/2}\mathbf{V}^{\mathrm{T}} \tag{3.9}$$

where

$$\mathbf{O}_i = \begin{bmatrix} \mathbf{C} \\ \mathbf{CA} \\ \vdots \\ \mathbf{CA}^{i-1} \end{bmatrix} \quad \text{and} \quad \boldsymbol{\Gamma}_i = \begin{bmatrix} \mathbf{A}^{i-1}\mathbf{G} & \dots & \mathbf{AG} & \mathbf{G} \end{bmatrix} \tag{3.10}$$

represent, respectively, the observability matrix and the reversed controllability matrix. In (3.8) and (3.9) the matrix $\mathbf{S}$ plays the role of a similarity transformation applied to the state-space model, therefore it can be set equal to the identity matrix $\mathbf{I}$. The matrices $\mathbf{A}$, $\mathbf{C}$, and $\mathbf{G}$, in equation (3.10) represent the state matrix, the output influence matrix, and the next state-output covariance matrix, respectively. Note that $\mathbf{C}$ and $\mathbf{G}$ can be easily extracted from the matrices $\mathbf{O}_i$ and $\boldsymbol{\Gamma}_i$, while $\mathbf{A}$ can be calculated by (3.4) as

$$\mathbf{A} = \mathbf{O}_i^{+}\mathbf{T}_{2|i+1}\boldsymbol{\Gamma}_i^{+}. \tag{3.11}$$

Applying the eigenvalues decomposition to $\mathbf{A}$ we get

$$\mathbf{A} = \boldsymbol{\Psi}\boldsymbol{\Omega}\boldsymbol{\Psi}^{\mathrm{T}} \tag{3.12}$$

where $\boldsymbol{\Psi}$ is an orthonormal matrix whose columns are the eigenvectors, while $\boldsymbol{\Omega} = \mathrm{diag}(\widetilde{\lambda}_1, \dots, \widetilde{\lambda}_n)$ is a diagonal matrix that contains the n eigenvalues of the state matrix. Therefore, reintroducing the previously omitted indices, it

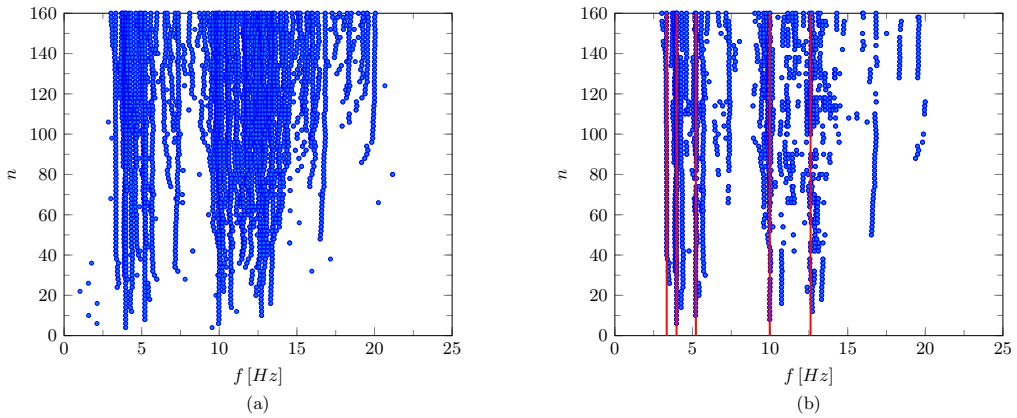

Figure 3.2: Example of stabilization diagram for the first hour monitoring: a) through SSI, b) after mode selection and clustering.

is possible to estimate the continuous-time damage sensitive parameters of the pth mode as follows:

- eigenvalues $\lambda_p^{(a,n)} = f_s \ln(\widetilde{\lambda}_p^{(a,n)})$;
- natural frequencies $\mu_p^{(a,n)} = |\lambda_p^{(a,n)}|/(2\pi)$;
- dumping ratios $\delta_p^{(a,n)} = -\Re\{\lambda_p^{(a,n)}\}/|\lambda_p^{(a,n)}|$;
- mode shapes $\boldsymbol{\phi}_p^{(a,n)} = \mathbf{C}^{(a,n)}\boldsymbol{\psi}_p^{(a,n)}$;

where $\boldsymbol{\phi}_p^{(a,n)}$ is a $l \times 1$ vector, and $\boldsymbol{\psi}_p^{(a,n)}$ is the pth column vector of $\boldsymbol{\Psi}^{(a,n)}$ defined in (3.12). The natural frequencies extracted through this approach in the *Z-24* environment for the first acquisition ($a = 1$) varying the order n is depicted in the stabilization diagram shown in Fig. 3.2a.

3.3 Mode selection

Through SSI, a large number of spurious modes are generated. To sift them out, there are several approaches to distinguish between the real modes and the spurious ones [48]. In this work we used four criteria, which must be satisfied to select a physic mode and reject the spurious ones. The four metrics extracted are: modal assurance criterion (MAC), mean phase deviation (MPD),

dumping ratio check, and complex conjugate poles check. Among several criteria, we choose to use these four metrics as a good compromise between computational cost and accuracy of the spurious modes detection. The metrics selected are widely used and discussed in literature [46, 48–50].

MAC

It is a dimensionless squared correlation coefficient between mode shapes, defined as [46]

$$\mathrm{MAC}(\boldsymbol{\phi}_p^{(a,n)}, \boldsymbol{\phi}_q^{(a,j)}) = \frac{|\boldsymbol{\phi}_p^{(a,n)\mathrm{T}} \boldsymbol{\phi}_q^{(a,j)}|^2}{||\boldsymbol{\phi}_p^{(a,n)}||_2^2 ||\boldsymbol{\phi}_q^{(a,j)}||_2^2} \tag{3.13}$$

with values between 0 and 1. A MAC larger than 0.9 indicates a consistent correspondence between the modes and so, very likely physical modes, whereas small values indicate poor resemblance of the two shapes and so, a spurious mode. A validation criteria based on MAC is the following [51]

$$d_{\mathrm{m}}(a,n,j,p,q) = \frac{|\lambda_p^{(a,n)} - \lambda_q^{(a,j)}|}{\max(|\lambda_p^{(a,n)}|, |\lambda_q^{(a,j)}|)} + 1 - \mathrm{MAC}(\boldsymbol{\phi}_p^{(a,n)}, \boldsymbol{\phi}_q^{(a,j)}) \tag{3.14}$$

where the first term is a measure of the distance between the pth and qth eigenvalues. With (3.14) a mode is considered physical when $d_{\mathrm{m}}(a,n,j,p,q) < 0.1$.

MPD

It measures the deviation of a mode shape components from the mean phase (MP) of all its components. A widely used technique to evaluate the MP is through the SVD [51]

$$\mathbf{P}\boldsymbol{\Lambda}\mathbf{Q}^T = [\Re\{\boldsymbol{\phi}_p^{(a,n)}\} \;\; \Im\{\boldsymbol{\phi}_p^{(a,n)}\}] \tag{3.15}$$

where $\mathbf{P}$ is $l \times 2$, $\mathbf{\Lambda}$ is 2×2, and $\mathbf{Q}$ is 2×2. From this decomposition the MPD can be evaluated as follows

$$\mathrm{MPD}(\boldsymbol{\phi}_p^{(a,n)}) = \frac{\sum_{r=1}^{l} |\phi_{r,p}^{(a,n)}| \arccos \left| \frac{\Re\{\phi_{r,p}^{(a,n)}\} q_{22} - \Im\{\phi_{r,p}^{(a,n)}\} q_{12}}{\sqrt{q_{12}^2 + q_{22}^2} |\phi_{r,p}^{(a,n)}|} \right|}{\sum_{r=1}^{l} \phi_{r,p}^{(a,n)}}. \tag{3.16}$$

When the ratio $\mathrm{MPD}(\boldsymbol{\phi}_p^{(a,n)})/(\pi/2) > 0.75$ the deviation of the mode shape components is stronger than MP and probably the mode is spurious, hence it is rejected.

Damping ratio and complex conjugate poles

For each mode the dumping ratio $\delta_p^{(a,n)}$ in real structures must be positive and lower than 0.2 (otherwise the structure is unstable) hence only modes with $0 < \delta_p^{(a,n)} < 0.2$ are considered. Moreover, if $\Re\{\lambda_p^{(a,n)}\} > 0$ the mode represents an unstable structure hence it is considered spurious.

In Fig. 3.2b, showing the stabilization diagram after mode selection, the remaining modes are represented with $\bar{\mu}_p^{(a,n)}$.

Q6: *Is it possible to perform frequencies tracking automatically after the fundamental frequencies extraction?*

3.4 Features Extraction

After mode selection, we propose clustering and a novel tracking algorithm to extract the time-series of the first two fundamental frequencies. To clarify the process, this section will be presented in the *Z-24* bridge environment, but can be easily extended to other structures.

3.4.1 Clustering

The stabilization diagram obtained after mode selection needs to be further processed to select the natural frequencies. For this purpose, K-means can be used to cluster the data, associating each cluster to a prospective natural

frequency corresponding to its centroid [29, 52]. For each acquisition the algorithm starts with an initial number of centroids (e.g., $K = 4$) initialized at random frequencies within the range of interest, and then their position will be updated until convergence.

Since the number of natural frequencies is unknown, the approach is repeated for different K values, ranging between 2 and 6; a large K tends to produce many spurious modes while small values may discard real fundamental frequencies.

Finally, the centroids configuration that leads to the solution with the lowest error (evaluated as the sum of the Euclidean distances between the centroids and the associated points) is selected. The output of K-means is the number of natural frequencies and their estimation (red lines in Fig. 3.2(b)). Repeating this procedure for each acquisition the blue points depicted in Fig. 3.3a are obtained.

At the end of clustering, the estimated frequencies are no more dependent on the model order n, consequently the notation $\bar{\mu}_p^{(a)}$ is adopted (see Fig. 4.2).

3.4.2 Density-based mode tracking

The tracking phase is the final step in the natural frequencies extraction chain. There are several algorithms able to find the frequency traces starting from the estimation made through the clustering phase [49]. In this thesis, a novel technique, that needs neither the frequency starting points nor their number, is proposed, contrary to several approaches that infer the number and the starting position of the fundamental frequencies through simulation of the physical structure. The proposed tracking algorithm consists of two steps: a starting phase and an online phase.

Initial phase

Without any assumption about the structure, the idea is to analyze the data, $\bar{\mu}_p^{(a)}$, and find some clusters of points that could be the starting ones. To perform this task, the first $N_\mathrm{t} = 200$ acquisitions (i.e. $\bar{\mu}_p^{(a)}$ with $a = 1, 2, \ldots, N_\mathrm{t}$, are considered (see Fig. 3.3a)). From this initial data, the number of points

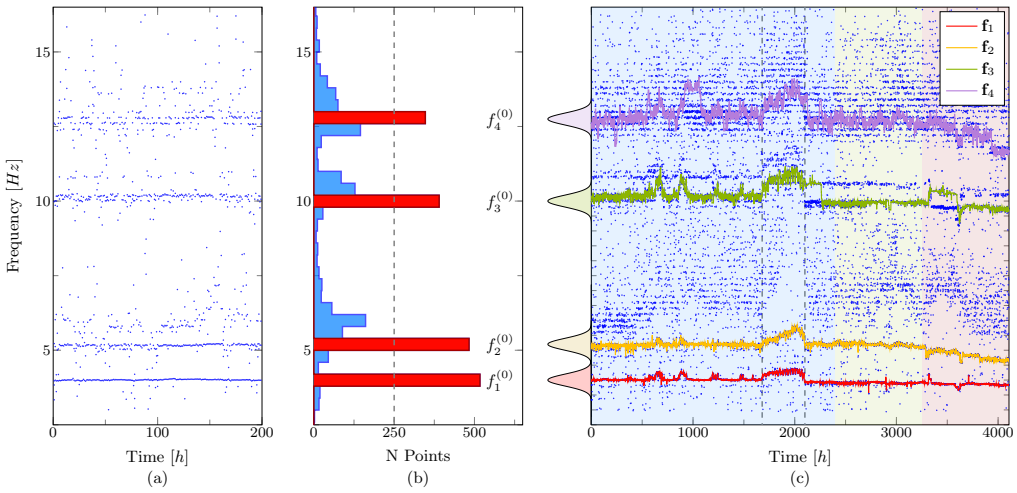

Figure 3.3: a) Residual modes after mode selection over time with one acquisition per hour (first 200 acquisitions). b) Natural frequencies selected after the initial phase of the tracking algorithm. c) First two natural frequencies estimation after the density-based tracking algorithm. Blue points represent the residual modes, red and yellow tracks represent, respectively, the first and second fundamental frequency estimated after the tracking algorithm; vertical dashed lines highlight the period when the average measured temperatures are below 0 °C [15], in particular, it has been demonstrated in [16] that when the temperature goes below 0 °C the natural frequencies of the bridge increase. Blue and green backgrounds highlight the acquisitions made during the normal condition of the bridge, used respectively as training and test sets, while red background stands for damaged condition acquisitions used in the test phase.

that fall into frequency bins of bandwidth $B_\mathrm{f} = 0.4\,\mathrm{Hz}$ are counted. The histogram obtained is depicted in Fig. 3.3b. Selecting the largest values of the histogram, the number of starting points and the corresponding frequencies, $f_s^{(0)}$, are estimated. Specifically, the first estimated frequency is evaluated as the average values of the frequencies that fall into the respective bins. For example, according to Fig. 3.3b the values of the starting points, $f_s^{(0)}$, in this case $s = 1, \ldots, 4$, are estimated and correspond to 4.0, 5.2, 10.1 and 12.8 Hz.

Online phase

In this phase, for each starting point a Gaussian kernel of the form

$$\mathcal{G}\big(\theta; f_s^{(a)}, \sigma_\mathrm{f}\big) = e^{-\frac{1}{2\sigma_\mathrm{f}^2}\left(\theta - f_s^{(a)}\right)^2} \tag{3.17}$$

is used to track the frequencies evolution over time. The parameter σ_f controls the kernel width and has been chosen equal to $\sigma_\mathrm{f} = 0.01\,\mathrm{Hz}$; larger values of σ_f make the system more reactive to fast frequency changes during the tracking but more sensitive to outliers due to the noisy measurements. The tracking algorithm is initialized with $f_s^{(0)}$ and iteratively updated through the following rule

$$f_s^{(a)} = (1-\epsilon) f_s^{(a-1)} + \epsilon \widetilde{f}_s^{(a)} \tag{3.18}$$

where the parameter $\epsilon \in [0,1]$ controls the impact of the new observation. Large values of ϵ reduce smoothness but allow capturing sudden changes of modal frequencies. For the specific data set $\epsilon = 0.7$ is selected. The innovation term $\widetilde{f}_s^{(a)}$ in (3.18) is evaluated through the Gaussian kernel as

$$\widetilde{f}_s^{(a)} = \frac{\sum_p \bar{\mu}_p^{(a)} \mathcal{G}\big(\bar{\mu}_p^{(a)}; f_s^{(a-1)}, \sigma_\mathrm{f}\big)}{\sum_p \mathcal{G}\big(\bar{\mu}_p^{(a)}; f_s^{(a-1)}, \sigma_\mathrm{f}\big)}. \tag{3.19}$$

The four tracks $\mathbf{f}_s = \{f_s^{(a)}\}_{a=1}^{N_\mathrm{a}}$ are shown in Fig. 3.3 and stored in the following matrix

$$\mathbf{F} = \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \mathbf{f}_3 \\ \mathbf{f}_4 \end{bmatrix}^\mathrm{T} = \begin{bmatrix} f_1^{(1)} & f_1^{(2)} & \dots & f_1^{(N_\mathrm{a})} \\ f_2^{(1)} & f_2^{(2)} & \dots & f_2^{(N_\mathrm{a})} \\ f_3^{(1)} & f_3^{(2)} & \dots & f_3^{(N_\mathrm{a})} \\ f_4^{(1)} & f_4^{(2)} & \dots & f_4^{(N_\mathrm{a})} \end{bmatrix}^\mathrm{T}. \tag{3.20}$$

Now that a tool to extract damage sensitive features is introduced and designed the following question arises:

Q7: *How can we test these strategies? Is there in literature a reference structure with accelerometric measurements both in standard and in damaged conditions?*

Chapter 4

Z-24 Bridge

4.1 System Configuration and Data Collection

The *Z-24* bridge was located in the Switzerland canton of Bern. The bridge was part of the road connection between Koppigen and Utzenstorf, overpassing the A1 highway between Bern and Zurich. It was a classical post-tensioned concrete two-cell box girder bridge with a main span of 30 m and two side spans of 14 m. The bridge was built as a freestanding frame with the approaches backfilled later. Both abutments consisted of triple concrete columns connected with concrete hinges to the girder. Both intermediate supports were concrete piers clamped into the girder. An extension of the bridge girder at the approaches provided a sliding slab. All supports were rotated with respect to the longitudinal axis that yielded a skew bridge. The bridge was demolished at the end of 1998 [53]. Before complete demolition, the bridge was subjected to a long-term continuous monitoring test and several progressive damage tests (see also Tab. 4.1):

- A long-term continuous monitoring test took place during the year before demolition. The aim was to quantify the environmental variability of the bridge dynamics.

- Progressive damage tests took place over a month, shortly before com-

Table 4.1: Chronological overview of applied damage scenarios

Date (1998)	Scenario
4 August	Undamaged condition
9 August	Installation of pier settlement system
10 August	Lowering of pier, 20 mm
12 August	Lowering of pier, 40 mm
17 August	Lowering of pier, 80 mm
18 August	Lowering of pier, 95 mm
19 August	Lifting of pier, tilt of foundation
20 August	New reference condition
25 August	Spalling of concrete at soffit, 12 m^2
26 August	Spalling of concrete at soffit, 24 m^2
27 August	Landslide of 1 m at abutment
31 August	Failure of concrete hinge
2 September	Failure of 2 anchor heads
3 September	Failure of 4 anchor heads
7 September	Rupture of 2 out of 16 tendons
8 September	Rupture of 4 out of 16 tendons
9 September	Rupture of 6 out of 16 tendons

plete demolition. The aim was to prove experimentally that realistic damage has a measurable influence on the bridge dynamics. Progressive damage tests were alternated with short-term monitoring tests while the continuous monitoring system was still running.

4.1.1 Data collection

The database of accelerometric measurements is freely available for research purposes on the website [17]. Such a dataset was built within the framework of the European Brite EuRam research project BE-3157, "System Identification to Monitor Civil Engineering Structures" (SIMCES). One of the main objectives of the SIMCES project was to deliver a proof of feasibility for vibration-based structural health monitoring of civil engineering structures by full-scale, long-term tests and progressive failure tests of a representative structure, the *Z-24* bridge.

The accelerometers used are force balance type FBA-11 by Kinemetrics [54]; they are triaxial devices with high sensitivity and are characterized by high reliability and low current consumption. The main specifications

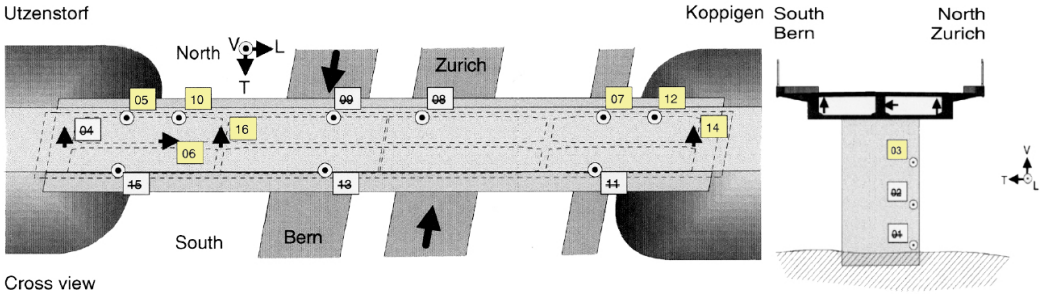

Figure 4.1: Data acquisition setup along the *Z-24* bridge: the selected accelerometers, their positions, and the measured acceleration direction [17].

are: full scale range ± 9.81 m/s^2, output ± 2.5 V, natural frequency critical damping 0.7, and supply voltage ± 12 V DC. The A/D converter used is the IOtech ADC 488/8SA [55], an 8 channel sample and hold with 8 differential inputs, 16 bit, and 100 kHz sampling rate.

The accelerometers position and their measurements axis are shown in Fig. 4.1. In this work, we considered $l = 8$ accelerometers, identified as 03, 05, 06, 07, 10, 12, 14, and 16, which are present in both long-term continuous monitoring phase and in the progressive damage one.[1] Longitudinal acceleration is collected by sensors 03 and 06, transversal acceleration is measured by sensors 14 and 16, and all the remaining sensors gather vertical accelerations.

Every hour $N_\mathrm{s} = 65536$ samples are acquired from each sensor with sampling frequency $f_\mathrm{samp} = 100$ Hz which corresponds to an acquisition time $T_\mathrm{a} = 655.36$ s. Since the measurements are not always available, there are $N_\mathrm{a} = 4107$ acquisitions collected in a period of 44 weeks. All the data collected by the selected sensors are organized in a tensor $\boldsymbol{\mathcal{D}}_\mathrm{raw}$ of dimensions $N_\mathrm{a} \times l \times N_\mathrm{s}$.

During the long-term monitoring, environmental parameters like temperature, wind speed, and traffic were also considered before and after each accelerometer measurement. In this work, only the temperature is considered, consisting of 20 measurements acquired by different thermometers placed

[1] Some accelerometers that experienced failures during the long-term monitoring have been avoided. Moreover, we selected a subset of accelerometers present in both phases to ensure data consistency.

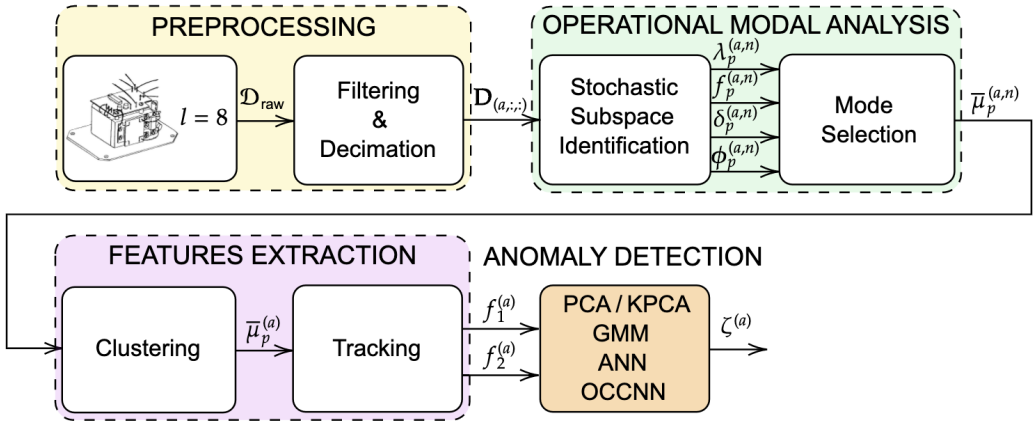

Figure 4.2: The block diagram for signal acquisition, processing, feature extraction, and detection.

along with the structure.

4.1.2 Data pre-processing

The block diagram depicted in Fig. 4.2 represents the sequence of tasks performed for the fully automatic anomaly detection approach presented in this work.

To reduce the computational cost and memory needs of the subsequent elaborations, some pre-processing steps have been applied to the data $\boldsymbol{\mathcal{D}}_{\text{raw}}$. First, a decimation by a factor 2 is applied to each acquisition; thus the sampling frequency is scaled to $f_{\text{samp}} = 50\,\text{Hz}$. Such sampling frequency is deemed sufficient because the *Z-24* fundamental frequencies fall in the $[2.5, 20]\,\text{Hz}$ frequency range [53]. After decimation, the data are processed with a finite impulse response (FIR) band-pass filter of order 30 with band $[2.5, 20]\,\text{Hz}$, to remove disturbances. The pre-processed data are then stored in a tensor $\boldsymbol{\mathcal{D}}$ of size $N_{\text{a}} \times l \times N$, where $N = 32000 \simeq N_{\text{s}}/2$. Regarding the environmetal parameters, the temperatures collected are averaged over time and among sensors to have one estimate for the whole bridge each hour. The main matrices, vectors and scalars introduced are summarised in Table 4.2.

Table 4.2: Data symbol, dimension, unit of measure, and description

Variable	Dimension	Unit	Description
$\boldsymbol{\mathcal{D}}_{\text{raw}}$	$N_a \times l \times N_s$	m/s^2	Raw accelerometer data
$\boldsymbol{\mathcal{D}}_{(a,:,:)}$	$N_a \times l \times N$	m/s^2	Preprocessed accelerometer data
$\lambda_p^{(a,n)}$	1×1	Hz	Eigenvalues
$\mu_p^{(a,n)}$	1×1	Hz	Natural frequencies
$\delta_p^{(a,n)}$	1×1		Dumping ratios
$\boldsymbol{\phi}_p^{(a,n)}$	$l \times 1$		Mode shapes
$\bar{\mu}_p^{(a,n)}$	1×1	Hz	Natural freq. after mode selection
$\bar{\mu}_p^{(a)}$	1×1	Hz	Natural frequencies after clustering
$f_1^{(a)}$	1×1	Hz	First natural freq. after tracking
$f_2^{(a)}$	1×1	Hz	Second natural freq. after tracking
$\widehat{\zeta}^{(a)}$	1×1		Decision labels

4.2 Algorithmic Complexity and Processing Time

In this section, we discuss the computational complexity of the algorithms adopted and the corresponding processing time when they are applied on the *Z-24* dataset. The most computationally expensive blocks are SSI and the anomaly detectors during the training phase: PCA, KPCA, GMM, ANN, OCCNN, OCCNN^2. Their complexity is evaluated as follows:

- SSI requires the SVD of matrix $\mathbf{T}_{1|i}$ (3.6), of size $li \times li$, and eigenvalues decomposition of matrix $\mathbf{A}$ (3.12), of size $n \times n$. SVD complexity is $\mathcal{O}(k(li)^2 + k'(li)^3)$ (where k and k' are constants which are 4 and 22, respectively, for the R-SVD algorithm) [56]; the complexity of eigenvalues decomposition is $\mathcal{O}(n^3)$. All these procedures are applied for each system order n considered in the range $n \in [2, 160]$ (with step 2, for a total of $n_\text{s} = 80$), and repeated for all the N_a acquisitions. Thus, the overall complexity is $\mathcal{O}(N_\text{a} \sum_n ((k(li)^2 + k'(li)^3) + n^3))$, and can be rewritten as $\mathcal{O}(N_\text{a} n_\text{s}(k(li)^2 + k'(li)^3 + 2n_\text{s}^3))$.

- PCA needs the evaluation of the covariance matrix of the training points with complexity $\mathcal{O}(N_{\mathrm{x}}s^2)$, and its eigenvalues decomposition, $\mathcal{O}(s^3)$, where s is the number of features and N_{x} the number of training points. The total complexity of PCA results to be $\mathcal{O}(N_{\mathrm{x}}s^2 + s^3)$.

- KPCA requires the evaluation of the kernel with complexity $\mathcal{O}(N_{\mathrm{x}}^2)$, its covariance matrix evaluation, $\mathcal{O}(N_{\mathrm{x}}^3)$, and its eigenvalues decomposition, $\mathcal{O}(N_{\mathrm{x}}^3)$. The overall complexity of KPCA is $\mathcal{O}(N_{\mathrm{x}}^2 + 2N_{\mathrm{x}}^3)$.

- GMM has complexity $\mathcal{O}(N_{\mathrm{x}}\mathcal{M}^2)$ where $\mathcal{M}$ is the order of the model, according to [57].

- ANN is a particular kind of NN with 5 layers of L_1, L_2, L_3, L_4 and L_5 neurons each, respectively. The computational cost for training the ANN using the backpropagation algorithm is $\mathcal{O}(N_{\mathrm{x}}N_{\mathrm{e}}(L_1L_2 + L_2L_3 + L_3L_4 + L_4L_5))$, where N_{e} is the number of epochs.

- OCCNN requires a two-step training with a different number of training points in each phase, N_{x1} and N_{x2} respectively, with $N_{\mathrm{x1}} > N_{\mathrm{x2}}$ as explained in the previous section. Considering a NN of 4 layers with L_1, L_2, L_3, and L_4 neurons, respectively, the complexity is $\mathcal{O}((N_{\mathrm{x1}} + N_{\mathrm{x2}})N_{\mathrm{e}}(L_1L_2 + L_2L_3 + L_3L_4))$.

- OCCNN^2 exploits the combination of ANN and OCCNN. Therefore, $\mathcal{O}(N_{\mathrm{x}}N_{\mathrm{e}}(L_1L_2 + L_2L_3 + L_3L_4 + L_4L_5) + N_{\mathrm{x2}}N_{\mathrm{e}}(L_6L_7 + L_7L_8 + L_8L_9))$ is the overall complexity, where L_1, L_2, L_3, L_4 and L_5 are the neurons in the ANN layers, and L_6, L_7, L_8, and L_9 are the ones in the OCCNN layers.

In Table 4.3 a summary of the complexity of all the algorithms is reported, along with the processing times experienced on a computer with $2,4\,\mathrm{GHz}$ Intel Core i5 processor and $2133\,\mathrm{MHz}$ LPDDR3 RAM.

Table 4.3: Computational complexity and processing time of the algorithms

Algorithm	Complexity	Time [s]
SSI	$\mathcal{O}(N_{\mathrm{a}} n_{\mathrm{s}}(k(li)^2 + k'(li)^3 + 2n_{\mathrm{s}}^3))$	12 334.29
PCA	$\mathcal{O}(N_{\mathrm{x}} s^2 + s^3)$	3.55
KPCA	$\mathcal{O}(N_{\mathrm{x}}^2 + 2N_{\mathrm{x}}^3)$	8.71
GMM	$\mathcal{O}(N_{\mathrm{x}} \mathcal{M}^2)$	2.50
ANN	$\mathcal{O}(N_{\mathrm{x}} N_{\mathrm{e}}(L_1 L_2 + L_2 L_3 + L_3 L_4 + L_4 L_5))$	26.26
OCCNN	$\mathcal{O}((N_{\mathrm{x1}} + N_{\mathrm{x2}}) N_{\mathrm{e}}(L_1 L_2 + L_2 L_3 + L_3 L_4))$	135.65
OCCNN^2	$\mathcal{O}(N_{\mathrm{x}} N_e(L_1 L_2 + L_2 L_3 + L_3 L_4 + L_4 L_5) + N_{x2} N_{\mathrm{e}}(L_6 L_7 + L_7 L_8 + L_8 L_9))$	54.86

4.3 Performance

In this section, the proposed algorithms are applied to the *Z-24* bridge data set to detect anomaly based on the fundamental frequencies estimation [14, 58, 59]. The performance is evaluated through the following metrics, described previously, considering only the test set through accuracy, precision, recall and F_1 score. In these numerical results, the dataset is divided into a training set and a test set. For this comparison, we decided not to use a validation phase because it is not needed in OCCNN and OCCNN^2 as there are no hyper-parameters to set. For the other algorithms, we set their parameters to ensure the maximum accuracy on the test set. With this methodology, we slightly overestimate the accuracy of PCA, KPCA, and GMM. Despite this setup favor PCA, KPCA and GMM, in the following paragraph it is shown that the proposed solution, OCCNN^2, overcame their performance.

The feature space has dimension $D = 2$ because only the first two fundamental frequencies are considered (this decision will be widely discussed in the next chapter), and unless otherwise specified the three data sets used for training, test in normal condition, and damaged condition, have cardinality $N_{\mathrm{x}} = 2399$, $N_{\mathrm{y}} = 854$, and $N_{\mathrm{u}} = 854$, respectively. For PCA, the number of components selected is $P = 1$. For KPCA, after several tests the values of P and γ that ensure the minimum reconstruction error are $P = 3$ and $\gamma = 8$.

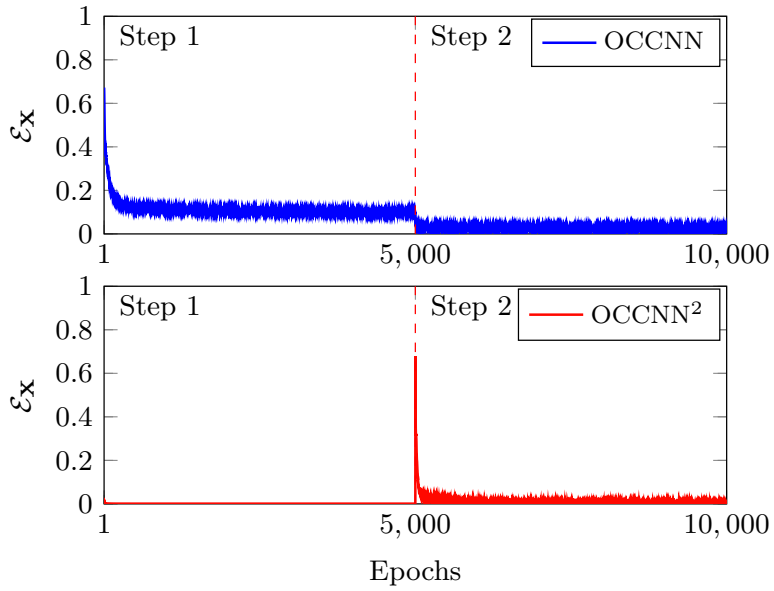

Figure 4.3: Error function evolution over the epochs during training.

For GMM the order of the model that maximize performance is $\mathcal{M} = 10$. Regarding the ANN, we adopted a fully connected network with 5 layers of, respectively, 100, 50, 1, 50 and 100 neurons, with ReLU activation functions. Note that, the same ANN is also used for the first step of the OCCNN^2 algorithm. The parameters of the OCCNN and OCCNN^2 are set accordingly to [36], resulting in $\alpha_1 = 0.3$, $\alpha_2 = 0.8$, and are largely independent on the spatial distribution of the feature points; this allows to presume that such OCCs can work on different structures and bridges. The NN has 2 hidden layers with $L = 50$ neurons each. All the NNs are trained for a number of epochs $N_\mathrm{e} = 5000$ with a learning rate $\rho = 0.05$. The error function adopted for a training set $\mathbf{X}$ is

$$\mathcal{E}_{\mathbf{X}} = -\sum_{n=1}^{N_\mathrm{x}} \sum_{c=1}^{C} t_{n,c} \ln \widetilde{t}_{n,c} \tag{4.1}$$

where N_x is the number of points in the training set, C is the number of classes ($C = 2$), $t_{n,c} = 1$ if the nth acquisition belongs to the cth class and zero otherwise, and $\widetilde{t}_{n,c}$ is the activation function value for point n of the cth

Table 4.4: Algorithm, parameter, value, and description

Algorithm	Parameter	Value	Description
SSI	n	$[2, 160]$	System order
Clustering	K	$[2, 6]$	Number of clusters
Tracking	N_{t}	200	Window size
Tracking	B_{f}	0.4 Hz	Frequency bin bandwidth
Tracking	ϕ_{f}	0.01 Hz	Kernel width
Tracking	ϵ	0.7	Innovation
PCA	P	1	Number of selected components
KPCA	γ	8	Gaussian kernel width
KPCA	P	3	Number of selected components
GMM	$\mathcal{M}$	10	Gaussian mixture model order
ANN	layers	5	Number of hidden layers
ANN	neurons	$\{100, 50, 1, 50, 100\}$	Number of neurons
NN	layers	2	Number of hidden layers
NN	neurons	$\{50, 50\}$	Number of neurons
Training	N_e	5000	Number of training epochs
Training	ρ	0.05	Learning rate
OCCNN	α_1	0.3	Density factor first iteration
OCCNN	α_2	0.8	Density factor second iteration
OCCNN^2	α_2	0.8	Density factor second iteration

output neuron [30]. As can be seen in Fig. 4.3 for each step of the algorithms the networks are trained for $N_{\mathrm{e}} = 5000$ epochs, quite enough for the error function to reach the minimum. A target false alarm rate on the training set is fixed equal to 0.01, from which a threshold is selected for each algorithm to guarantee such constraint. In Table 4.4 the principal hyper-parameters are summarized.

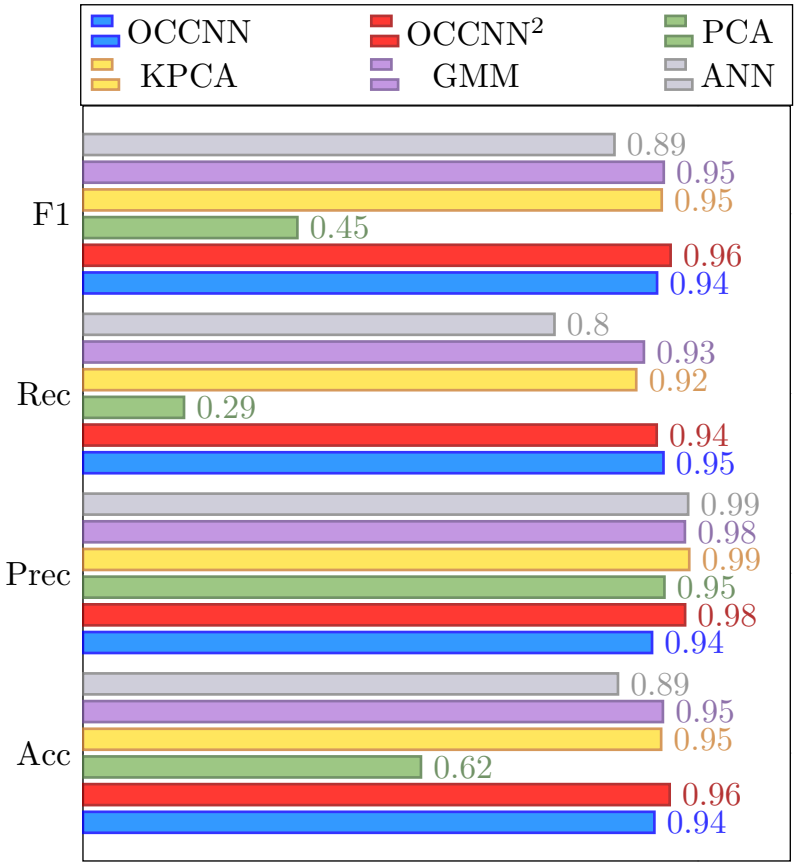

Figure 4.4: Comparison of the classification algorithms in terms of F_1 score, recall, precision, and accuracy.

4.3.1 Algorithm Comparison

The performance comparison of the algorithms is reported in Fig. 4.4. As we can see, considering the F_1 score OCCNN outperforms PCA and ANN but is overtaken by KPCA and GMM. This happens because the non-uniform data distribution of the training set influences Pollard's estimator which overestimates the density. As a result, there are too many adversarial points generated, which lead to an underestimation of the normal class boundary. The OCCNN^2 overcomes this limitation and shows the best F_1 score and accuracy. This represents a remarkable result considering the low number of hyper-parameters to be tuned.

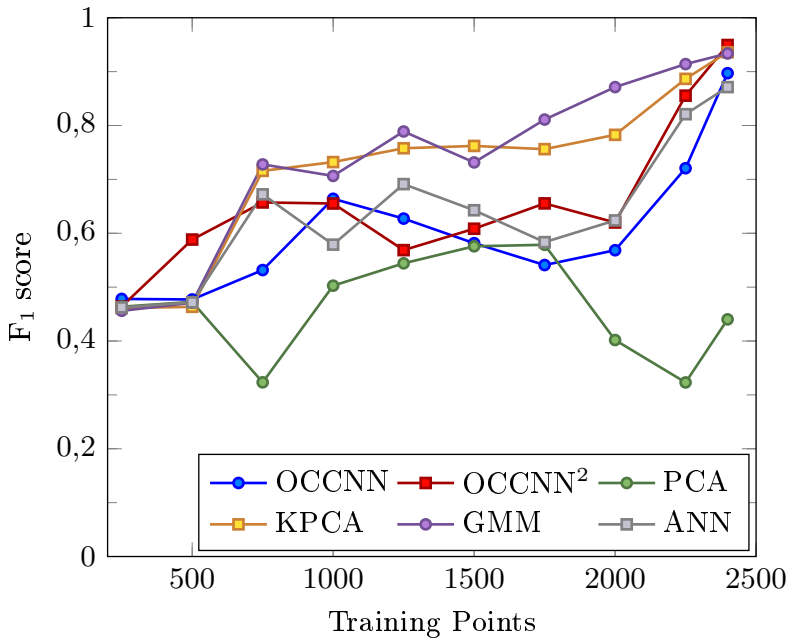

Figure 4.5: F_1 score varying the number of points, N_x, used for training, with $N_y = 854$ and $N_u = 854$.

4.3.2 Impact of the training set and responsiveness

In Fig. 4.5 and Fig. 4.6, the F_1 score dependence on the number of points used to train the algorithms, N_x, and the number of anomaly points, N_u, used to test the algorithms, are depicted.

The number of training points, N_x, in Fig. 4.5 varies from 250 to 2399 with steps of 250 points (except for the last step, equal to 149). This plot shows how many points are necessary for the correct estimation of the normal class boundaries. In these results, all the numbers of points used for test are kept constant (i.e. $N_y = N_u = 854$) to maintain the same boundaries between the classes. As it can be seen in the figure, all the algorithms require a high number of data points corresponding to long-lasting monitoring (almost one year [14]) to provide the highest F_1 score. The algorithms that show a sharper increase in their performance, hence requiring less training points, are KPCA and GMM.

In Fig. 4.6, the number of training points is kept constant, $N_x = 2399$,

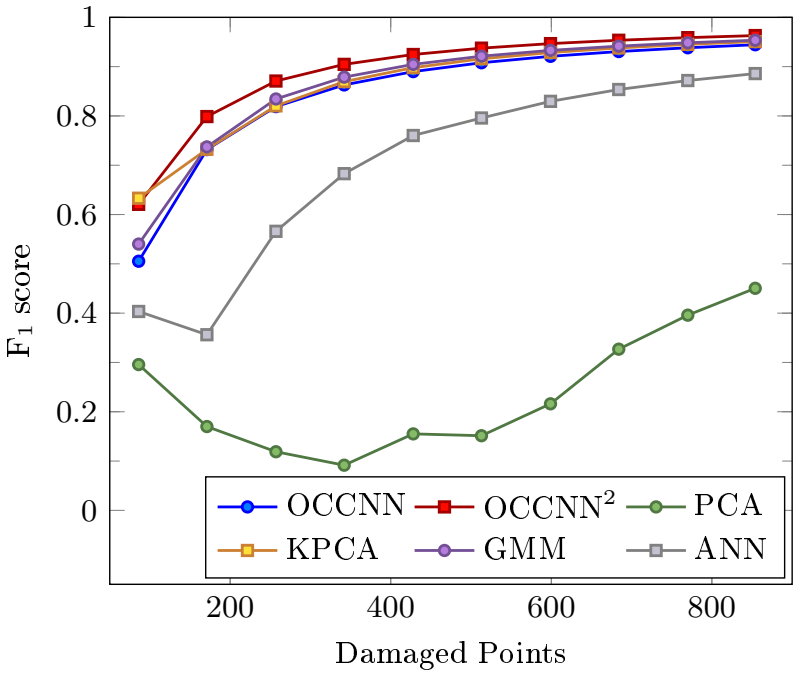

Figure 4.6: F_1 score varying the number of points during damage, N_u, used to test the algorithms, with $N_x = 2399$ and $N_y = 854$.

and the number of anomaly test points, N_u, varies from 86 to 854 with 10 points steps. The plot shows that OCCNN^2 exhibits the best responsiveness (i.e. it requires the lowest number of damaged points to achieve higher values of the F_1 score).

Now that the performance has been analyzed in detail, the following question arises:

Q8: *Is it possible to automatically select the most reliable fundamental frequencies, among all the ones extracted, in order to reduce the problem dimensionality without affecting the performance?*

Chapter 5

Dimensionality Reduction

In this section, we introduce two possible strategies to reduce the dimensionality of the damage-sensitive features extracted, with the intention of reducing the amount of data stored and transmitted through the sensor network and increase the anomaly detection performance or at least do not deteriorate it. Moreover, in case of wireless monitoring network, several strategies can be adopted to manage the sensors and increase the network life time [60–63], but firstly, a good practice can be to define the minimum number of sensors necessary to accomplish the anomaly detection task.

5.0.1 Feature extraction

This technique consists of mapping a set of data in a low-dimensional feature space, trying to reduce an error function that represents the distance between the original data and the remapped data obtained from the mapping subspace. In this work, we decided to use the ANN as auto-encoders to accomplish this task [29, 31]. In this case, the normalized feature matrix $\mathbf{X}$ is fed to an ANN with the classic bottleneck structure that provides a mapping set of layers and a consequent set of demapping layers. The input and output layers have the same dimension of the feature space, and the labels during the training phase must be set equals to the input data point. With this structure, the data are mapped in lower-dimensional feature space (with dimension equal to the number of neurons present in the bottleneck layer)

and then reconstructed through the demapping layers minimizing the error with respect to the input data.

5.0.2 Feature selection

This approach proposes to select the most reliable features among all the available ones with some metrics to eliminate noisy components that can deteriorate the damage detection capability of the OCC algorithms. In this thesis, we decided to evaluate some statistical features (i.e., variance, kurtosis, and entropy) to identify the most reliable fundamental frequencies among the four extracted, which are widely discussed in the next section.

5.1 Performance

The performance is evaluated through the following metrics, described previously, considering only the test set:

$$\text{Accuracy} = \frac{T_\text{P} + T_\text{N}}{T_\text{P} + T_\text{N} + F_\text{P} + F_\text{N}}$$

$$\text{Precision} = \frac{T_\text{P}}{T_\text{P} + F_\text{P}}$$

$$\text{Recall} = \frac{T_\text{P}}{T_\text{P} + F_\text{N}}$$

$$\text{F}_1 \text{ score} = 2 \cdot \frac{\text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}}$$

where T_P, T_N, F_P, and F_N, represent, respectively, true positive, true negative, false positive, and false negative predictions.

The feature space has dimension $D = 4$, the three data sets used for training, test in normal condition, and damaged condition, have cardinality $N_\text{x} = 2399$, $N_\text{y} = 854$, and $N_\text{u} = 854$. For PCA, the number of components selected is $P = 1$. For KPCA, after several tests the values of P and γ that ensure the minimum reconstruction error are $P = 3$ and $\gamma = 8$. Regarding the ANN we adopted a fully connected network with 7 layers of, respectively,

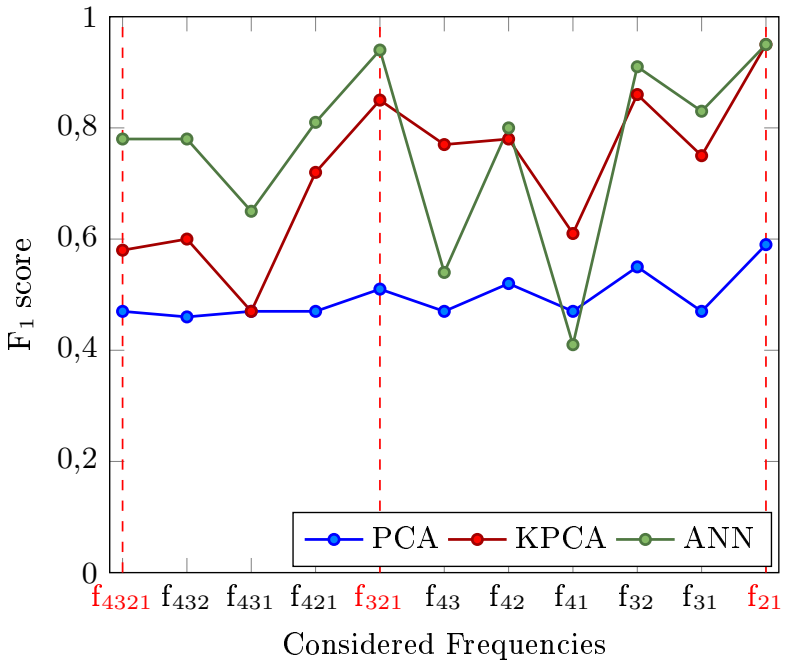

Figure 5.1: F_1 score varying the considered fundamental frequencies, vertical dashed red lines indicate the best configuration with 4, 3, and 2 frequencies.

50, 20, 10, k, 10, 20 and 50 neurons with k number of features extracted, with ReLU activation functions for the feature extraction task, and a fully connected network with 5 layers of, respectively, 100, 50, 1, 50 and 100 for the anomaly detection. All the NNs are trained for a number of epochs $N_e = 5000$ with a learning rate $\rho = 0.05$. The error function adopted for a training set $\mathbf{X}$ is

$$\mathcal{E}_{\mathbf{X}} = -\sum_{n=1}^{N_x}\sum_{c=1}^{C} t_{n,c} \ln \widetilde{t}_{n,c}, \tag{5.1}$$

where N_x is the number of points in the training set, C is the number of classes ($C = 2$), $t_{n,c} = 1$ if the nth acquisition belongs to the cth class and zero otherwise, and $\widetilde{t}_{n,c}$ is the activation function value for point n of the cth output neuron [30].

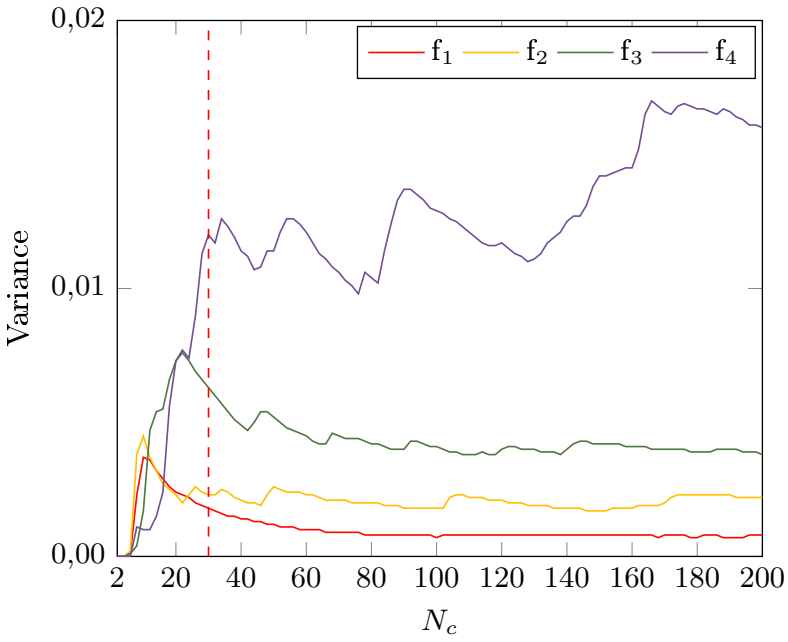

Figure 5.2: Modal frequencies variance for different points; vertical dashed red lines indicate the minimum number of points for the correct frequency sorting.

5.1.1 Frequencies selection

First of all, a brute force approach is implemented to evaluate the fundamental frequencies that provide the best performance of the anomaly detection algorithms in terms of F_1 score with all the possible combinations of features. As reported in Fig. 5.1, the best performance is achieved with algorithms with the same feature configuration, highlighted by the red vertical dashed lines, that allows sorting the features with increasing importance as f_4, f_3, f_2, and f_1. This is why a good metric to describe the fundamental frequencies reliability must sort the frequencies in the same order. With this aim, three statistical metrics are reported as a good candidate to accomplish this task.

The first one is the sample variance of the frequencies extracted defined as

$$\text{Variance} = \frac{\sum_{i=1}^{N_c}(f_s^{(i)} - \bar{f}_s)^2}{N_c - 1}$$

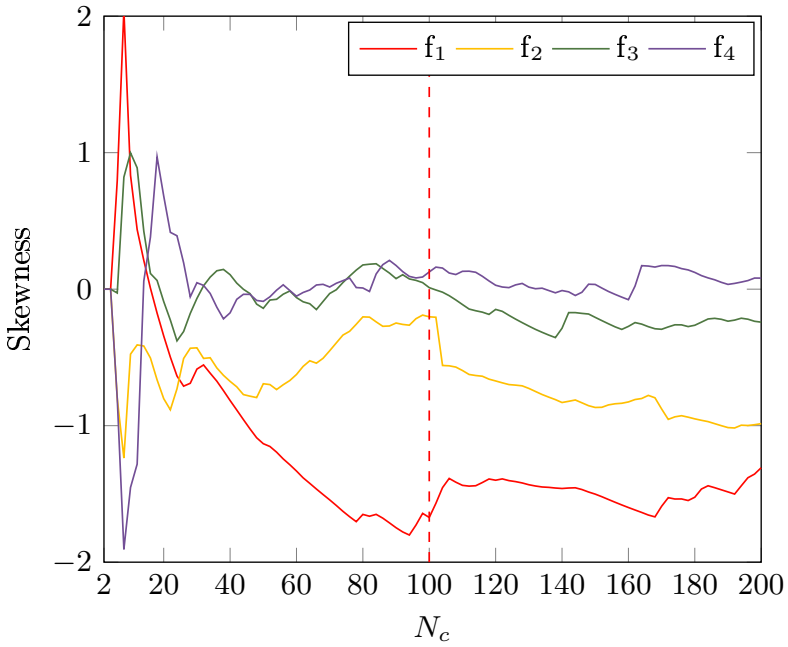

Figure 5.3: Modal frequencies skewness varying the number of points; vertical dashed red lines indicate the minimum number of points for the correct frequency sorting.

where N_c is the number of acquisition considered, $f_s^{(i)}$ is the ith acquisition of the sth fundamental frequency, and $\bar{f}_s$ stands for the mean value of the sth frequency evaluated in the interval $\{1, \dots, N_c\}$. As shown in Fig. 5.2, this feature works well after $N_c = 30$ observations; as depicted, the variance of reliable features is lower with respect to the noisy ones; so, this method can be successfully used to sort the frequencies in the correct order.

The second metric proposed is the skewness that measures the asymmetry of the probability distribution, i.e.

$$\text{Skewness} = \frac{\frac{1}{N_c}\sum_{i=1}^{N_c}(f_s^{(i)} - \bar{f}_s)^3}{\left(\sqrt{\frac{1}{N_c}\sum_{i=1}^{N_c}(f_s^{(i)} - \bar{f}_s)^2}\right)^3}.$$

As shown in Fig. 5.3, this metric can also be used to sort the frequencies correctly; the method becomes reliable after around $N_c = 100$ observations.

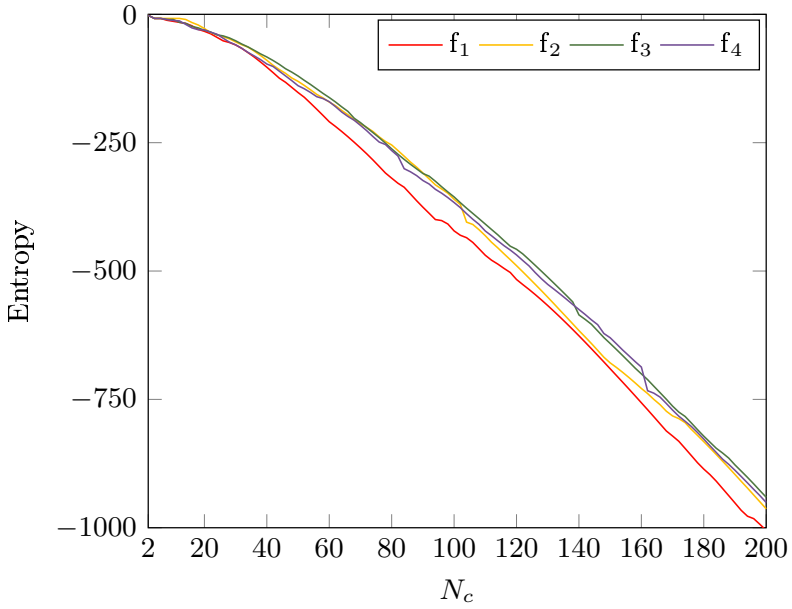

Figure 5.4: Modal frequencies entropy varying the number of points.

Finally, in Fig. 5.4 the entropy is evaluated as a further metric

$$\text{Entropy} = -\sum_{i=1}^{N_c} P(f_s^{(i)}) \log_{10} P(f_s^{(i)})$$

where the probability density function $P(f_s^{(i)})$ is evaluated numerically implementing data binning. In this case, the trend is descendent because the information introduced by new measurements decreases by increasing the number of observations. This metric can also be used to sort the frequencies, but in this case, f_3 and f_4 are inverted for some values of N_c and f_2 is very close to the previous two and can be missorted.

5.1.2 Algorithm Comparison

This paragraph provides a performance comparison of the anomaly detection algorithms and dimensionality reduction techniques. In Fig. 5.5 the performance of PCA, KPCA, and ANN is evaluated in term of F_1 score, varying the dimensionality of the features considered. Dashed curves are referred to

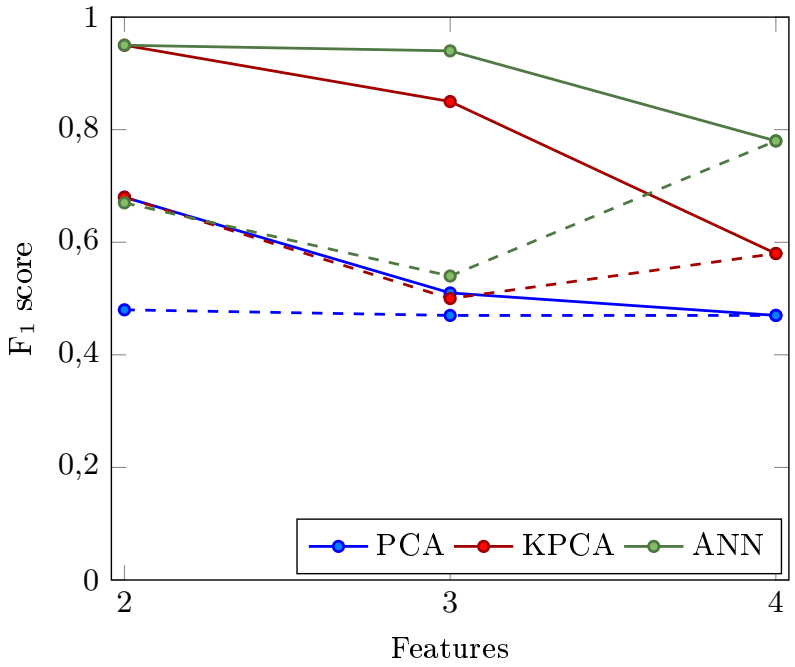

Figure 5.5: F_1 score varying the dimensionality of the feature space, dashed lines correspond to frequency extraction technique, and continuous lines are referred to as the frequency extraction approach.

features extracted with auto-encoder implemented with ANN. As we can observe from the plot, the dimensionality reduction slightly decreases the algorithms' performance; hence, for this application, feature extraction does not improve the system's anomaly detection capability. On the contrary, continuous lines are referred to feature selected through the metrics described in the previous paragraph. In this case, selecting the correct fundamental frequencies tends to increase the detection performance, so it is strongly recommended for this application because both reduce the dimensionality of the problem and increase the anomaly detection capability. In Fig. 5.6 the performance of the algorithms in terms of F_1 score, recall, precision, and accuracy is reported in the best configuration, hence considering only the first two fundamental frequencies selected by the described metrics (f_1 and f_2). As we can see, the F_1 score of both KPCA and ANN are around 95%, and the accuracy for ANN is greater than 96% that represents a remarkable

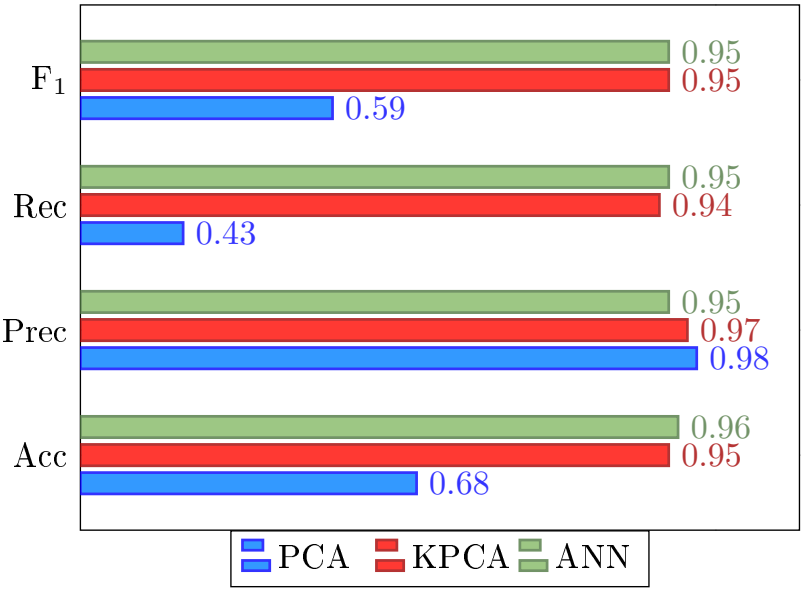

Figure 5.6: Comparison of the classification algorithms in terms of F_1 score, recall, precision, and accuracy with the best feature configuration.

result in this anomaly detection application.

The performance assessment of the automatic detection of anomalies revealed promising results. However, a widespread use of such techniques is a large-scale monitoring scenario poses the following research questions:

Q9: *Is it possible to further reduce the amount of data that must be stored by the network?*

Q10: *Is there a possibility to reduce the network infrastructure costs, reducing the amount of sensors necessary to detect a damage or using low-cost sensors?*

Chapter 6

Data Management

This chapter analyzes the minimum amount of data that must be stored to perform anomaly detection on the vibrational waveforms, and some strategies that can be implemented to reduce such volume of data, both representing important topics often studied in literature [6, 64].

Considering a network of $l = 8$ synchronized sensors interconnected to a coordinator that store the accelerometric measurements, where each sensor acquire $N_\mathrm{s} = 65536$ samples each acquisition with $N_\mathrm{b} = 16$ resolution bits for $N_\mathrm{a} = 4107$ acquisition, it is trivial to observe that the total amount of data stored by the coordinator is equal to $M_\mathrm{t} = N_\mathrm{s} N_\mathrm{b} N_\mathrm{a} l \simeq 32\,\mathrm{Gbit} = 4\,\mathrm{GB}$. This considerable amount of data has been stored in a year of non continuous measurements. In fact the effective acquisition time can be estimated as $T_\mathrm{t} = T_\mathrm{a} N_\mathrm{a} \simeq 44860\,\mathrm{m} \simeq 448\,\mathrm{h}$, that shows how huge could be the volume of data in a continuous measurement system; in a year it is around 47 GB. To reduce the amount of data the first step is the decimation approach. In this application the fundamental frequency of the bridge falls in the interval $[0, 20]\,\mathrm{Hz}$, so to respect the Shannon's theorem with a guard band of 5 Hz a sampling frequency $f_\mathrm{samp} = 50\,\mathrm{Hz}$ it is enough to capture the bridge oscillations, since the measurements are acquired by accelerometers with $f_\mathrm{samp} = 100\,\mathrm{Hz}$, a decimation by factor 2 can be adopted and the total amount of data is halved: $M_\mathrm{d} = M_\mathrm{t}/2 \simeq 2\,\mathrm{GB}$. Moreover, starting from the decimated acquisitions, three other parameters can be modified to find some possible configuration

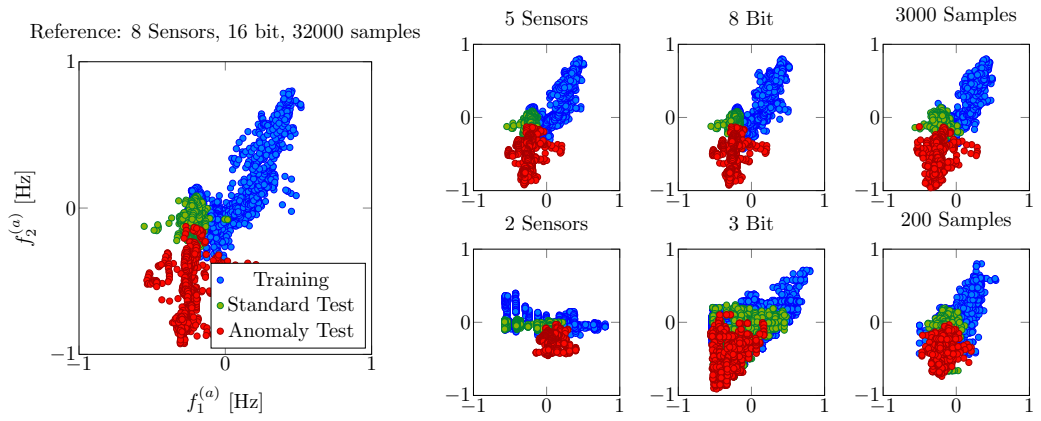

Figure 6.1: Examples of feature transformation due to the effect of low number of sensors, low number of bits, and low number of samples with respect to the standard measurement condition reported on the left.

that does not deteriorate the performance of the OCC algorithms, but can reduce the volume of data:

- The number of sensors l can be reduced in order to contain the network costs and the amount of data produced by the sensor network.
- The number of samples N_s or equivalently the acquisition time T_a can be reduced to limit the network's operative time, the energy consumption, and the data produced.
- The number of bits N_b can be reduced to drop the storage occupation of the single acquisition and contain the accelerometer cost.

All these possibilities will be analyzed and widely discussed in the next section. In Fig. 6.1 some working points of the system are reported and compared to the reference working condition after decimation ($l = 8$, $N_\mathrm{d} = 32768$, $N_\mathrm{b} = 16$).

6.1 Performance

The performance is evaluated through the accuracy, considering only the test set. The feature space has dimension $D = 2$, and the three data sets used for

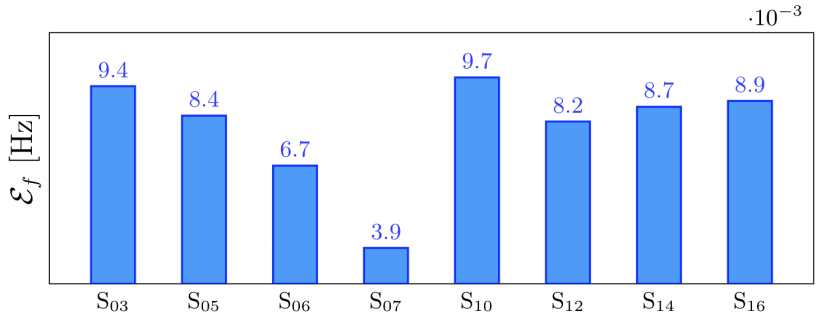

Figure 6.2: Error produced removing the selected sensor.

training, test in normal condition, and damaged condition, have cardinality $N_{\mathrm{x}} = 2399$, $N_{\mathrm{y}} = 854$, and $N_{\mathrm{u}} = 854$. For PCA, the number of components selected is $P = 1$. For KPCA, after several tests, the values of P and γ that ensure the minimum reconstruction error are $P = 3$ and $\gamma = 8$. For GMM the order of the model that maximizes performance is $\mathcal{M} = 10$. Regarding the OCCNN^2 the first step boundary estimation is made by a fully connected ANN with 7 layers of, respectively, 50, 20, 10, 1, 10, 20 and 50 neurons, with ReLU activation functions, and a fully connected NN with 2 hidden layers with $L = 50$ neurons, each one for the second step. All the NNs are trained for a number of epochs $N_{\mathrm{e}} = 5000$ with a learning rate $\rho = 0.05$. The error adopted to evaluate the points displacement in the feature space from the original position due to the different configurations is the RMSE, defined as

$$\mathcal{E}_f = \frac{1}{\sqrt{N_{\mathrm{a}} N_{\mathrm{s}}}} \sqrt{\sum_{s=1}^{N_{\mathrm{s}}} \sum_{n=1}^{N_{\mathrm{a}}} (f_s^{(n)} - \bar{f}_s^{(n)})^2} \tag{6.1}$$

where N_{s} is the number of features ($N_{\mathrm{s}} = 2$), $f_s^{(n)}$ is the sth feature of the nth acquisition in the initial configuration, and $\bar{f}_s^{(n)}$ is the relative data point in the modified configuration.

6.1.1 Sensors Relevance

Before evaluating the error introduced by reducing the number of sensors, it is fundamental to evaluate the importance that each one has in the fun-

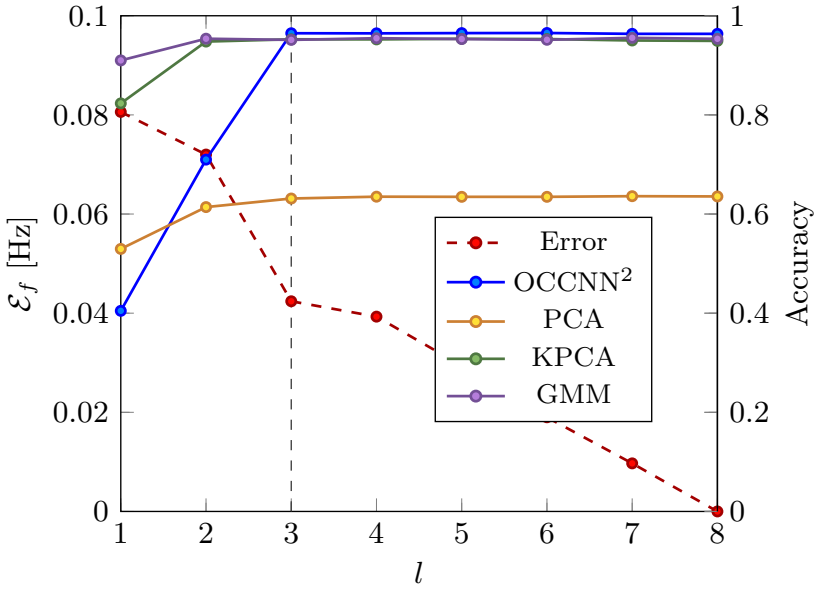

Figure 6.3: Error varying the number of sensors available.

damental frequencies estimation. It is widely known in literature that the sensor position strongly affects the mode estimation [14, 65, 66]. To verify the sensors' relevance, we removed them one-by-one and evaluated the error resulting in the feature space points with respect to the standard condition. The error is calculated as the RMSE defined previously. As it can be seen in Fig. 6.2, sensor S_{10} generates the most significant error in the fundamental frequencies extraction when removed. With this technique it is possible to sort the sensors from the most relevant to the least one as follows: S_{10}, S_{03}, S_{16}, S_{14}, S_{05}, S_{12}, S_{06}, S_{07}. To evaluate the performance with respect to the number of sensors used to extract fundamental frequencies, the sensors will be removed in the same order, to always consider the worst condition with the given number of sensors.

6.1.2 Number of Sensors

Now that the sensor relevance is defined, we are able to verify the performance by varying the number of sensors used on the structure to derive the fundamental frequencies with low error. As we can see in Fig. 6.3, the ac-

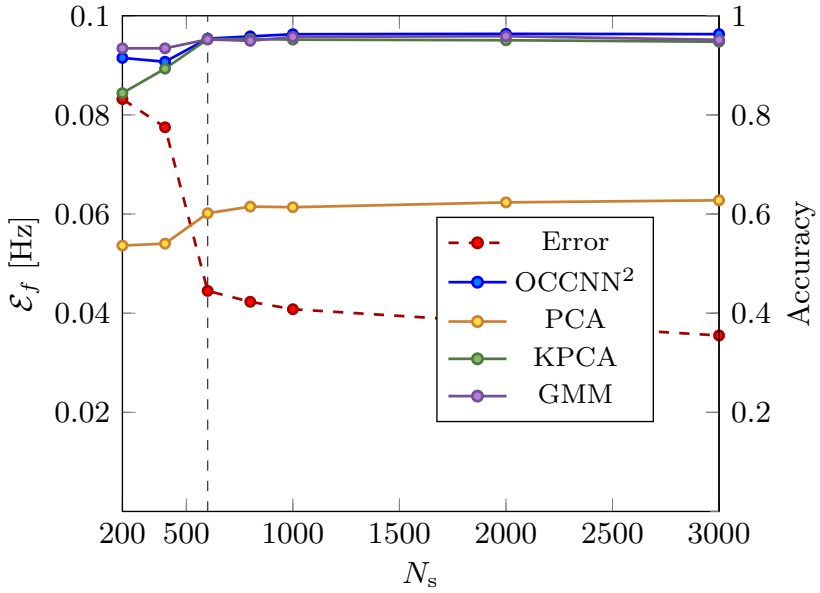

Figure 6.4: Error varying the number of samples.

curacy of the algorithms remains almost the same as long as the number of sensors available is greater than 2, as the error presents a significant increase in correspondence of the gap between 2 and 3 sensors. Thus we can deduce that the minimum number of sensors that can be used to monitor the *Z-24* bridge is equal to 3. In this configuration it is easy to notice that the amount of data stored is reduced to $M_{\text{sen}} = N_{\text{s}} N_{\text{b}} N_{\text{a}} 3 \simeq 0.8\,\text{GB}$.

6.1.3 Number of Samples

To evaluate the effect of the acquisition time on the algorithms' anomaly detection performance, we progressively reduced the number of samples used to extract the structure's fundamental frequencies. As we can see in Fig. 6.4, the performance of the algorithms remains almost constant as long as the number of samples N_{s} is greater than 600, which corresponds to an acquisition time of 12 s with a sampling frequency $f_{\text{samp}} = 50\,\text{Hz}$. By reducing the acquisition time drastically without relevant loss of performance, we achieve a strong reduction of data occupation, that in this configuration is equal to $M_{\text{sam}} = 600 N_{\text{b}} N_{\text{a}} l \simeq 0.04\,\text{GB}$ without performance degradation.

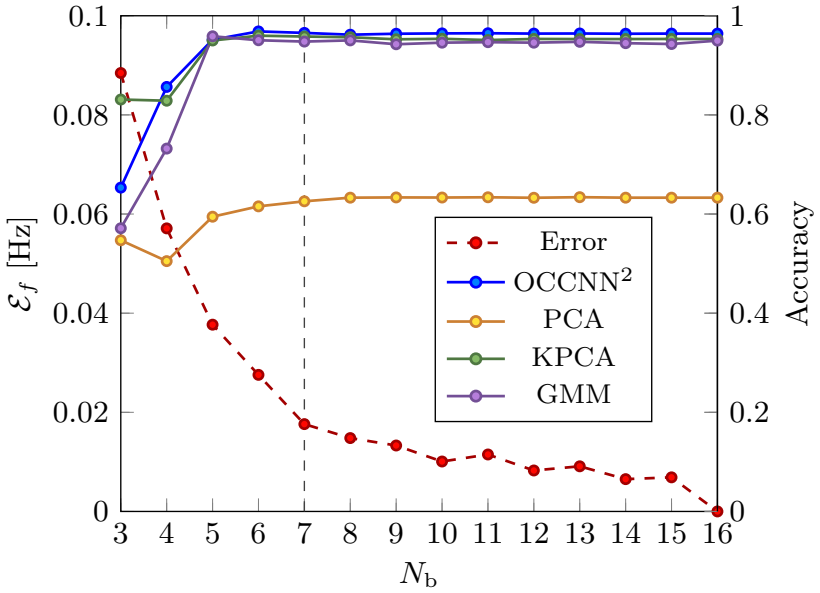

Figure 6.5: Error varying the number of bits.

6.1.4 Number of Bits

The number of bits can also be dropped to reduce the volume of data stored and the single accelerometer cost. To test their impact on the performance, we progressively reduced the number of bits used to encode the waveforms extracted from the accelerometers as reported in Fig. 6.5. As we can see, the error remains contained as far as the number of bits used to encode the samples is greater than 6; likewise, the accuracy of the algorithms remains high as long as the error is contained. Several low-cost accelerometers are available on the market with a resolution $N_\mathrm{b} = 8$, and these results show that this type of sensor could accomplish the anomaly detection task. In this case, the data occupation is $M_\mathrm{bit} = 8N_\mathrm{s}N_\mathrm{a}l \simeq 1\,\mathrm{GB}$.

6.2 Observations

Three different approaches are proposed to reduce the volume of data stored and limit the costs of sensors and network infrastructure necessary to monitor the structure. In this sense, when the goal is reducing the amount of

data stored, it is good practice cut down the observation time using a number of accurate sensors; when the target is to minimize the sensor cost, a good practice is to adopt low-cost sensors (with low resolution or number of bits per sample), combined with long observation time with a network of several sensors; when the objective is to contain the network infrastructure cost, a low number of accurate sensors and long observation time can be considered. To evaluate the error introduced from these strategies and the performance of the algorithms, the RMSE and the accuracy are used as metrics. The results show that these strategies can be adopted without significant loss of performance; in fact, all the algorithms except the PCA, ensure accuracy greater than 94% in all the proposed configurations, with the maximum performance reached by OCCNN^2 whose accuracy never goes down below 95%.

The effectiveness of this approach led us to poses the following question:

Q11: *Is it possible to exploit accelerometric measurements acquired by senors installed on a structure also to infer human activities?*

Chapter 7

Human Activities Classification Using Biaxial Seismic Sensors

7.1 Introduction

The problem of identifying and classifying the presence of a target in a particular environment with low-cost sensors remains a key issue for outdoor security applications [35, 36, 67, 68]. The variability of the ground and environment characteristics (i.e., weather conditions, humidity, temperature, wind speed) makes the target detection more complicated than a controlled indoor environment.

In literature, many works propose to use networks of geophones (which present weak dependencies from the environment) to capture the ground vibration to detect the presence of persons in indoor scenarios [69] or to classify several vehicles with different weights in a well defined outdoor area [70]. In [71], a method for detecting intruders and predicting their activities outdoor using a seismic sensor is presented. Similarly, in [72], the objective is to detect and classify different targets (e.g., humans, vehicles, and animals led by a human) using seismic and passive infrared sensors. Other solutions exploit cooperative sensors (i.e., microphones and geophones) and data fusion techniques to improve vehicle classification accuracy and estimate their velocity [73, 74]. In [75], a system architecture for the classification of moving

objects using both scalar and multimedia sensors is proposed.

In this work, we aim to distinguish between four different human activities, ride a bike, drive a car, walk, run, and investigate the possibility of doing it using only a biaxial geophone (i.e., with two channels, horizontal and vertical). In particular, the intuition behind this work is that forces involved in the activities based on sliding contact with the ground solicit mainly horizontal ground vibrations, while activities characterized by footsteps are responsible for vertical vibrations. The consequences of this consideration on the solution proposed are explained more in-depth in Section 7.3. The presented solution is based on the dimensionality reduction of frequency-domain features of the data, based on PCA, to ensure good classification performance at low computational cost [76]. Then, two different classification methods, using two distinct classifiers for each one, are presented.

The main contributions of the letter are the following. *i)* We propose a human activities classification system that uses only a biaxial geophone; *ii)* We propose the PSD as a sensitive feature for the activity identification, and PCA to remap the data in an advantageous feature space; *iii)* We propose two different classification methods: classification made with a single classifier or using a cascade of two classifiers; *iv)* For both methods, we compare two distinct classifiers, support vector machine (SVM) and k-NN, and we measure the impact of vertical and horizontal components on classification performance. Moreover, we present a comparison with conventional algorithms that exploit cross-correlation, named template matching, and an alternative classification method based on linear discriminant analysis (LDA).

7.2 System Model

As depicted in Fig. 7.1, a passive geophone with two channels (vertical and horizontal) captures ground vibrations in an outdoor environment. The acquired data are then processed to classify between four human activities: ride a bike, drive a car, walk, and run.

At first, the analog signals produced by the horizontal and vertical channels of the geophone, $x_{\mathrm{h}}(t)$ and $x_{\mathrm{v}}(t)$, pass through an analog to digital

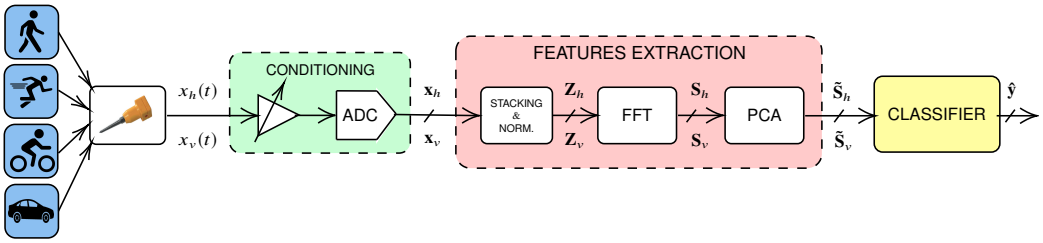

Figure 7.1: Illustration of the processing data chain to extract features from the geophone signals and perform classification.

converter (ADC) with sampling frequency f_s and resolution N_bit. The output of the conversion consists in two time series $\mathbf{x}_\mathrm{h} = \{x_\mathrm{h}(k/f_\mathrm{s})\}_{k=0}^{K-1}$ and $\mathbf{x}_\mathrm{v} = \{x_\mathrm{v}(k/f_\mathrm{s})\}_{k=0}^{K-1}$ of K samples. Subsequently, the time series are firstly split in N_w row vectors, obtained through a partially overlapped sliding window of length W samples, with $W \leq K$, and a sliding step Δ_w, and then rearranged into the matrices $\mathbf{Z}_\mathrm{h}$ and $\mathbf{Z}_\mathrm{v}$, of size $N_\mathrm{w} \times W$, by stacking them. From now on, since the processing stages apply to $\mathbf{Z}_\mathrm{h}$ and $\mathbf{Z}_\mathrm{v}$ separately, we indicate both with $\mathbf{Z}$ for the sake of conciseness. In this phase, the samples of each observation window are normalized column-wise such that the results are zero mean row vectors with $\max_j |z_{n,j}| = 1, n = 1, \ldots, N_\mathrm{w}$. Finally, the data are labeled by activity for the classification during the training phase.

7.2.1 PSD estimation

For the identification of relevant features for human activity classification, we analyze the signal in the frequency domain [34]. In particular, we estimate the PSD of the acquired samples using the weighted overlapped segment averaging (WOSA) for each row of $\mathbf{Z}$, with Hanning window and 50% overlap. The number of points, D, of the fast Fourier transform (FFT) regulates the trade-off between the frequency resolution and the PSD estimation accuracy [77]. The estimated PSDs $\{\mathbf{s}_j\}_{j=1}^{N_\mathrm{w}}$ are then organized in the matrix $\mathbf{S}$ of size $N_\mathrm{w} \times D$, generically used at this stage to indicate both $\mathbf{S}_\mathrm{h}$ and $\mathbf{S}_\mathrm{v}$. Afterward, PCA is applied to $\mathbf{S}$ to reduce the dimensionality of the data and extract the features. From now on, we split N_w into two subsets N_o and N_t, for the training and the test phases, respectively.

7.2.2 PCA

Principal component analysis (PCA) distills the essential information from the dataset, which is then represented as a set of new orthogonal variables called principal components obtained from a linear combination of the original data [78]. For the calculation of the principal components, we consider only the N_o points of the training subset. After centering the matrix $\mathbf{S}$ by subtracting its column-wise sample mean, we evaluate the sample covariance matrix $\mathbf{\Sigma} = \frac{1}{N_o}\mathbf{S}^T\mathbf{S}$. Then, $\mathbf{\Sigma}$ is factorized by eigenvalues decomposition (EVD) $\mathbf{\Sigma} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$, where $\mathbf{\Lambda}$ is the diagonal matrix of eigenvalues, ordered from the largest to the smallest, and $\mathbf{Q}$ is the matrix of eigenvectors [79].

In order to perform dimensionality reduction, we only keep the first D_h or D_v eigenvalues of $\mathbf{\Lambda}$ and the corresponding eigenvectors $\tilde{\mathbf{Q}}$ (i.e., selected columns of $\mathbf{Q}$). The projections $\tilde{\mathbf{S}}$ of the observations in the components subspace through the new projection matrix $\tilde{\mathbf{Q}}$ are $\tilde{\mathbf{S}} = \mathbf{S}\tilde{\mathbf{Q}}$, where $\tilde{\mathbf{S}}$ is a $N_w \times D_h$ or $N_w \times D_v$ matrix, and D_h, $D_v \leq D$ are the number of principal components considered for $\tilde{\mathbf{S}}_h$ and $\tilde{\mathbf{S}}_v$. Note that, while the principal components are calculated solely over the training points N_o, all the N_w points are projected in the components subspace. These two matrices represent the selected features used to train and test the classifiers described in Section 7.3.

7.3 Classification Techniques

After dimensionality reduction, the features $\tilde{\mathbf{S}}_h$, $\tilde{\mathbf{S}}_v$ are used by a classifier to determine the type of human activity. In particular, two classification methods are proposed.

- **Single classifier.** For each observation window, the data of the two channels are jointly processed. The $N_w \times (D_h + D_v)$ matrix $\tilde{\mathbf{S}}_{hv}$ is built concatenating the matrices $\tilde{\mathbf{S}}_h$ and $\tilde{\mathbf{S}}_v$ as $\tilde{\mathbf{S}}_{hv} = [\tilde{\mathbf{S}}_h\ \tilde{\mathbf{S}}_v]$. The monoaxial (i.e., horizontal or vertical) geophone configuration is obtained setting $D_v = 0$ or $D_h = 0$, respectively.

- **Cascade classifier.** The classification is carried out in two steps. Firstly, a three-class classifier uses the data acquired from the horizontal channel $\tilde{\mathbf{S}}_{\mathrm{h}}$ to classify between bike, car, and footsteps (i.e., run and walk are treated as if they are the same class). Then, if the first classifier does not choose for the bike or car classes, a second one uses the $\tilde{\mathbf{S}}_{\mathrm{v}}$ features to discriminate between run and walk. The rationale behind this approach is that based on the dominant forces during the interaction between the target and the ground, all the activities based on sliding contact (i.e., car and bike) tend to stimulate horizontal ground vibrations. In contrast, activities characterized by footsteps tend to excite vertical vibrations.

Hereafter, the two classifiers used in this work are briefly described. Besides, ordinary cross-correlation based classification method used both in time and frequency domain is reviewed. For the sake of clarity, $\mathbf{y}$ is the vector of actual classification labels of length N_{w}, $\hat{\mathbf{y}}$ is the vector of classification labels estimated by the algorithms, $\tilde{\mathbf{s}}_n$ is the n-th row of $\tilde{\mathbf{S}}_{\mathrm{h}}$ (either $\tilde{\mathbf{S}}_{\mathrm{v}}$ or $\tilde{\mathbf{S}}_{\mathrm{hv}}$), and $\mathcal{S}$ is the feature space of dimension D (either D_{h} or D_{v}). Thus, each point, both for training and test, is represented by the pair $(\tilde{\mathbf{s}}_n, y_n)$.

7.3.1 Support vector machine

The SVM constructs a set of hyperplanes in high-dimensional space that can be used for tasks like classification or regression [29]. Hence, it is a parametric learning algorithm whose error function includes a regularization term as follows:

$$g(\tilde{\mathbf{w}}) = \sum_{n=1}^{N_{\mathrm{o}}} \ln\left(1 + e^{-y_n(\tilde{\mathbf{s}}_n^{\mathrm{T}}\tilde{\mathbf{w}})}\right) + \lambda||\tilde{\mathbf{w}}||_2^2$$

where $\tilde{\mathbf{w}}$ are the weights of the parametrical model, and λ is the regularization parameter [29, 30]. The hyperplane that performs a good separation is the one that maximizes its distance from the nearest training points of each class, and it is identified by the set of weights $\tilde{\mathbf{w}}$ that minimizes the error function. Given a test point $(\tilde{\mathbf{s}}_m, y_m)$, the estimated label $\hat{y}_m$ is given by $\hat{y}_m = \tilde{\mathbf{s}}_m\tilde{\mathbf{w}}^{\mathrm{T}}$.

7.3.2 k-NN

In k-NN a set of N_o pairs $\{(\tilde{\mathbf{s}}_n, y_n)\}_{n=1}^{N_\mathrm{o}}$ is given as training set, where $\tilde{\mathbf{s}}_n$ takes values in the feature space $\mathcal{S}$ upon which is defined the metric $d(\tilde{\mathbf{s}}_n, \tilde{\mathbf{s}}_m)$; the Euclidean distance in this work. Given a test point $(\tilde{\mathbf{s}}_m, y_m)$, the estimate of y_m is given by the nearest neighbor training point with respect to the test point as

$$\hat{y}_m = \big\{ y_k : \tilde{\mathbf{s}}_k = \arg\min_{\tilde{\mathbf{s}}_n} d(\tilde{\mathbf{s}}_n, \tilde{\mathbf{s}}_m) \big\}. \tag{7.1}$$

With (7.1) we can assign $\tilde{\mathbf{s}}_m$ to the same class of the nearest $\tilde{\mathbf{s}}_n$. If the number of training points is large enough, it makes sense to use the majority rule of the nearest k neighbors, instead of the single nearest neighbors [80]. In the case of binary classification, k is chosen to be odd to avoid that a point is assigned to two different classes.

7.3.3 Template matching

Classification methods that use cross-correlation as a similarity metric, often called *template matching*, are extensively used in signal processing [81, 82]. Similarity can be searched both in time and frequency domain. In the frequency-domain case, at first the estimated PSDs, $\{\mathbf{s}_j\}_{j=1}^{N_\mathrm{w}}$, of both horizontal and vertical channel, are normalized to have zero mean and unitary standard deviation. Then, given a set of N_o pairs $(\{\mathbf{s}_n, y_n)\}_{n=1}^{N_\mathrm{o}}$ as training set, and a test point $(\mathbf{s}_m, y_m)$, we define the vector $\mathbf{r}$ such that its n-th element is

$$r_n = \max\{\mathrm{corr}(\mathbf{s}_n, \mathbf{s}_m)\} \quad n = 1, \ldots, N_\mathrm{o} \tag{7.2}$$

where $\mathrm{corr}(\mathbf{s}_n, \mathbf{s}_m)$ is the cross-correlation vector, of length $2D-1$, between the sequences $\mathbf{s}_n$ and $\mathbf{s}_m$ [81, 82]. Carrying out the same operations for both the horizontal and the vertical channel, we obtain the vectors $\mathbf{r}_\mathrm{h}$ and $\mathbf{r}_\mathrm{v}$, respectively, which are then concatenated to obtain $\mathbf{r}_\mathrm{hv} = [\mathbf{r}_\mathrm{h}\ \mathbf{r}_\mathrm{v}]$. The estimated label $\hat{y}_m$, is thus

$$\hat{y}_m = \big\{ y_{(k-1) \bmod N_\mathrm{o}+1} : k = \arg\max_n (\mathbf{r}_\mathrm{hv}) \big\}. \tag{7.3}$$

Similar operations are performed in the time-domain case.

7.4 Numerical Results

In this section, we present several tests to compare the performance of the classifiers in different settings. For the measurements, a biaxial two-channel geophone, with natural frequency of 4.5 Hz, frequency bandwidth of $0.2 - 240$ Hz, and sensitivity of 78 V/m/s has been used. The geophone has been placed in a flowerbed next to a car park. All the activities have been performed within a distance of 6 m from the geophone. The seismic sensor was interfaced with the PC using an ADC with sampling frequency $f_\mathrm{s} = 400$ Hz and resolution $N_\mathrm{bit} = 14$ bit. Furthermore, for each channel a LM 386 amplifier has been adopted to increase the signal-to-quantization noise ratio.

Each activity has been repeated several times for a whole period equal to $T_\mathrm{a} = 20$ min, using different vehicles and involving people with various height and gait. In particular, six people took part in the run and walk measurements, while two bikes and a car, driven by two persons, were used. Then, the acquisition period has been split into several observation windows of duration $T_\mathrm{ob} = W/f_\mathrm{s}$ ranging between 5 s to 30 s, depending on the test, and using a sliding step of $\Delta_\mathrm{w} = 5$ s; the PSDs was computed with a $D = 64$ points FFT.

To properly study the classifiers' performance, the acquisitions have been randomly split in *training* and *test* sets as:

- **Training and validation**: 60% of all PSDs of each activity are randomly chosen to calculate the projection matrix of PCA, $\tilde{\mathbf{Q}}$. Then, the projected points are used to train the classifiers. To set the hyper-parameters of SVM and k-NN, 10% of these points are randomly chosen and then used to perform cross-validation. The resulting optimal values are $k = 3$ and $\lambda = 0.1$.

- **Test**: the other 40% of PSDs are used to test the performance of the algorithms. In this case, the projection in the principal components

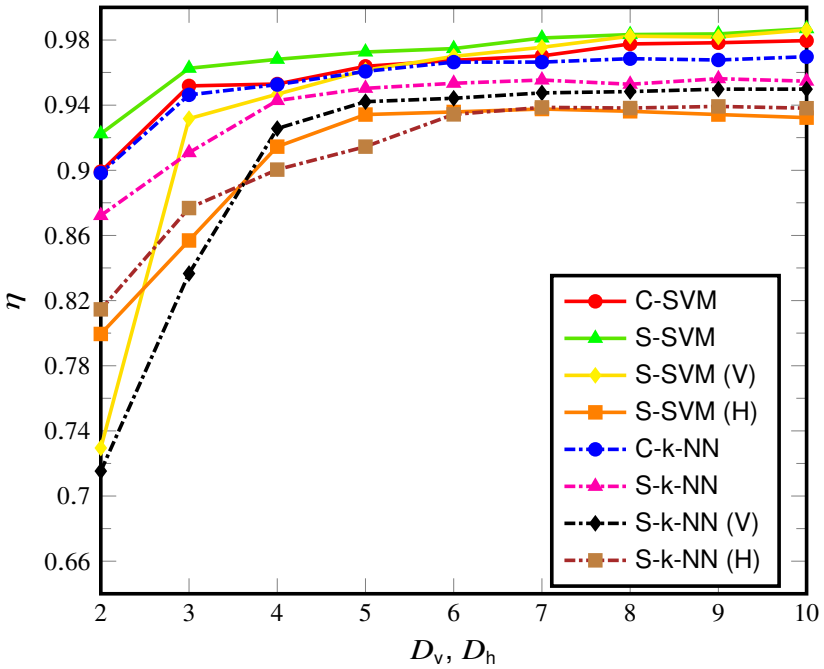

Figure 7.2: Accuracy varying PCA components of the vertical and horizontal channels, with $T_{ob} = 20\,s$.

subspace through PCA is made using the projection matrix computed during the training phase.

The same ratio of training and test points has also been used for the template matching based approaches and the LDA. However, no cross-validation is required in these cases. As a figure of merit, to evaluate the performance of the classifiers, we consider the *accuracy*, defined as

$$\eta = \frac{\text{number of points correctly classified}}{\text{number of total points}}. \tag{7.4}$$

7.4.1 Accuracy vs. number of PCA components

For both the horizontal and the vertical channel, the number of selected components D_h and D_v have been varied and the effect on the accuracy of the algorithms has been studied. In this test the observation window has been set to $W = 8000$ samples, corresponding to an observation time $T_{ob} = 20\,s$.

As shown in Fig. 7.2, the single SVM (S-SVM) classifier provides always the best performance, for each value of D_{h} and D_{v}; in particular, with only $D_{\mathrm{h}} = D_{\mathrm{v}} = 3$ components, it reaches an accuracy greater than 96%, while to achieve the same accuracy with the cascade SVM (C-SVM) configuration, it is necessary to use a higher number of PCA components (D_{h}, $D_{\mathrm{v}} \geq 5$). On the contrary, the single k-NN (S-k-NN) converges to an accuracy roughly equal to 93%, much lower than that of the cascade k-NN (C-k-NN), which stands at almost 97%. These results prove that, in the case of k-NN and for this experiment setup, it is better to adopt the cascade solution and use the samples coming from the channels separately. With regard to the monoaxial scenario, we can see that for the SVM, the performance in case we use only the samples of the vertical channel is comparable with those of C-SVM and S-SVM solutions, when $D_{\mathrm{v}} = D_{\mathrm{h}} = 10$. On the contrary, if we use only the horizontal channel, the accuracy is always worse. Differently, the performance of k-NN is always better when using a biaxial geophone.

7.4.2 Accuracy vs. observation duration

In this test, the observation window duration varies between 5 s and 30 s. The selected principal components are $D_{\mathrm{h}} = D_{\mathrm{v}} = 5$. As shown in Fig. 7.3, here too, the performance of the single k-NN is worse than the cascade solution. On the contrary, the accuracy S-SVM is always better than the cascade solution, with a higher gap when $T_{\mathrm{ob}} = 5\,\mathrm{s}$. Moreover, we can notice that for $T_{\mathrm{ob}} > 20\,\mathrm{s}$ the accuracy of the cascade k-NN and of the single SVM are comparable, while that of the cascade SVM decreases. The results for the monoaxial geophone confirm what has been experimented in Section 7.4.1.

The performance of template matching based approaches, both in time and frequency -domain, are included in Fig. 7.3 for comparison. As can be noted, the traditional techniques which do not exploit data structure are outperformed by the proposed solutions. In addition, the accuracy of LDA in both cascade (C-LDA) and single (S-LDA) configurations, are also included [83]. While for very long ($> 30\,\mathrm{s}$) observation windows, the S-LDA classifier reaches the same performance as the proposed solution based

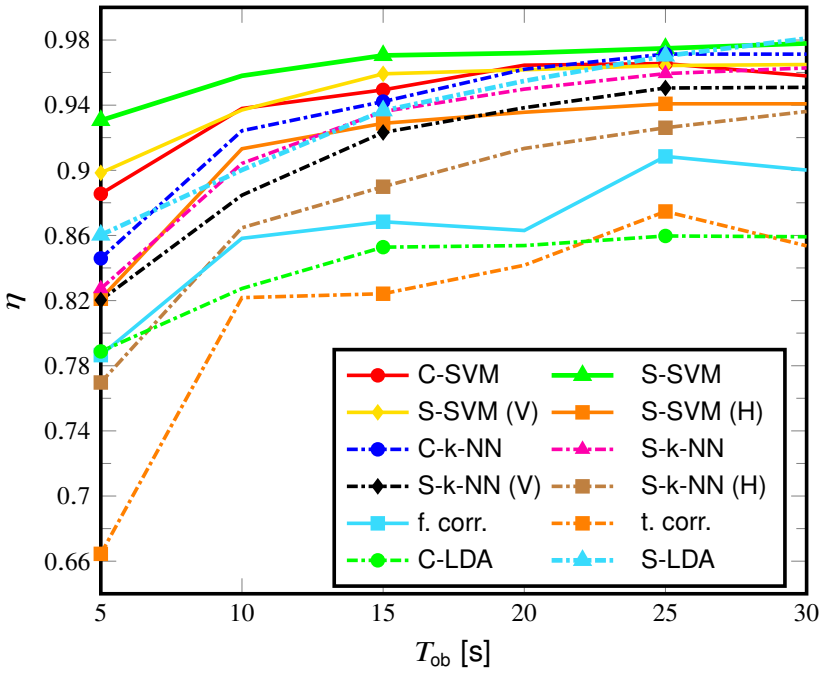

Figure 7.3: Accuracy varying the observation window duration, T_{ob}. For PCA-based classifiers $D_{\text{h}} = D_{\text{v}} = 5$.

on PCA, followed by SVM, for short observations some of the proposed solutions (C-SVM, S-SVM, and S-SVM (V)) outperform LDA. Note that, C-LDA classification always exhibit poor performance almost equivalent to template matching.

7.5 Observations

A passive human activity classification method exploiting the ground vibrations observed by a two-channel geophone is proposed. Data collected by the two channels are processed by a PCA-based dimensionality reduction to extract significant features in the frequency domain. The classification step is performed by either a single classifier or a cascade of two classifiers. The analysis considered the most important parameters (observation window and the number of PCA components) and setups (joint processing or cascade processing) to provide a complete set of results which may assist the system

designer. Based on performance assessment on real waveforms, the extensive numerical results show that the number of selected PCA components and the observation window duration strongly impact the performance of the algorithms. For example, for large observation windows, the best solution is represented by the single SVM classifier that jointly processes both vertical and horizontal data. However, as complexity may have a role in designing autonomous low-power devices, if the k-NN classifier is preferred, the cascade solution performs the best.

The flexibility of these tools led us to poses the following question:

Q12: *Is it possible to apply similar techniques to solve other problems?*

Chapter 8

Anomaly Detection Using WiFi Signals of Opportunity

8.1 Introduction

With the advent of the technological revolution named Internet of things (IoT), increasingly pervasive and context-adaptive communication systems are conquering the radio-frequency (RF) spectrum [10]. Since spectrum population may represent an issue in some frequency bands, e.g., the over-crowded industrial, scientific and medical (ISM) ones, there is an increasing interest in exploiting existing over-the-air signals, devised for some specific purpose, to perform other tasks thus avoiding dedicated radio emissions. WiFi routers, broadcast stations, and mobile cellular networks are only a few examples of such signals of opportunity (SoOp) [84–87].

Security in homes, industrial environments, and facilities is becoming a critical aspect of modern society, and for such reasons, ambient intelligence is gaining attention recently [37]. Video-based surveillance systems using, for example, cameras are the dominant technology in such scenarios. However, the personal privacy issue is still a reason for deterring users. The ambient intelligence paradigm is not only beneficial for security purposes but more generally as an enabler for context-aware applications like smart homes, to name one example.

The capability to extract information from the effects of the propagation on RF signals opens up a way to acquire knowledge about an environment by the observation of SoOp. In this context, there are two main characteristics of the observed signal used for detection, one is the RSS, and the other is channel-state information (CSI). The RSS is very easy to get with simple hardware, so it gained considerable attention in the last decade. A human motion localization method that exploits standard deviation of RSS is presented in [88], [89], and the detection and tracking of multiple persons in an indoor environment is proposed in [90]. However, techniques that exploit received power are susceptible to multipath propagation and need a multitude of devices to be effective, even when confined in indoor environments [91], [92].

Channel estimation allows greater precision when used in motion detection compared to RSS measurements. In [93], fine-grained subcarrier information (i.e., channel frequency response) is exploited to design a device-free passive human detection. In [94], a scheme for adaptive indoor passive detection is proposed, where the CSI amplitude measured in an indoor environment is shown to vary in the presence of human motion. In [95] the authors propose a device-free RF environmental vision system based both on RSS and CSI, while in [96] a crowd counting system that uses SoOp is presented.

Target/change detection can also be performed with radar techniques, either using dedicated sources with large bandwidth [97], or using sources of opportunities [98] with smaller bandwidth but in large environments that ensure target/anomaly spatial resolution. However, our goal here is to avoid dedicated signals with very large bandwidth and perform anomaly detection with SoOp even when the target/anomaly is not spatially resolvable.

This work proposes a ML approach for anomaly detection in an indoor environment using WiFi SoOp. In particular, the main contributions are the following.

- We use inexpensive RF sensors to collect SoOp [35]. Environmental changes are detected through RF channel modifications without demodulating the signal.

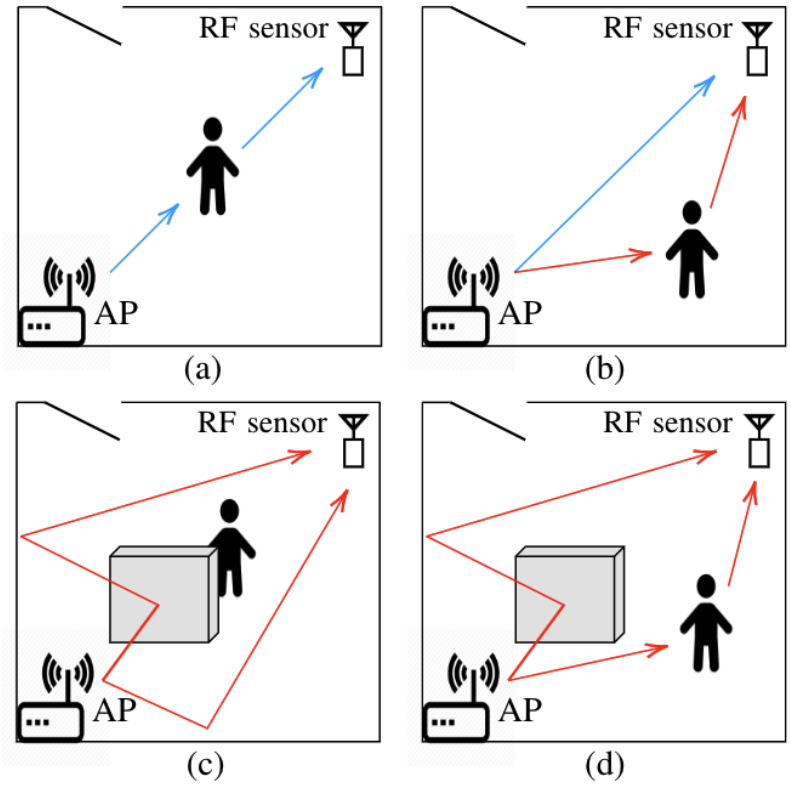

Figure 8.1: The four scenarios considered with the anomaly represented by a human being.

- In particular, we record and analyze samples that belong to beacon packets transmitted by an access point (AP).
- We compare the performance of two ML classifiers such as PCA and KPCA, as a function of the number of beacon packets collected [36].
- The tests have been performed in both line-of-sight (LOS) and non-line-of-sight (NLOS) conditions.
- Finally, we show that the proposed approach exhibits superior performance than a well-known RSS-based solution in terms of accuracy, even using just a single sensor.

Table 8.1: Scenarios considered for anomaly detection.

Condition	Scenario	Target/Anomaly
LOS	(a)	Static in the middle of the direct path
	(b)	Moving in the room
NLOS	(c)	Static in the middle of the direct path
	(d)	Moving in the room

8.2 System Overview and Problem Setup

Let us consider a scenario of the ones depicted in Fig. 8.1, where a RF sensor performs down-conversion followed by an analog-to-digital conversion to capture samples of over-the-air WiFi signals. In particular, without loss of generality, let us focus on the case where there is an AP operating in the 2.4 GHz band. While, nowadays, there are several high-performance devices for spectrum monitoring, it is interesting to investigate the possibility of using very low-cost devices [35]. Using inexpensive devices the measurement quality is usually affected by front-end impairments, low sampling frequency, low resolution, and low sensitivity. However, as is shown in Section 8.4, the samples have been acquired with enough accuracy to allow the ML algorithms to achieve high accuracy.

As case studies, let us focus on the four different scenarios depicted in Fig. 8.1, where the visibility of the AP and the position of the target are varying. In Table 8.1, a synthetic description of the scenarios is presented. For the sake of conciseness, we refer to such scenarios as (a), (b), (c), and (d), respectively.

8.2.1 Data pre-processing

The samples of the complex envelope of the received signal at the RF sensor, r_i, $i \in \mathbb{N}$, are pre-processed to detect the time of arrival (ToA) of the beacon

packets emitted by the AP. In the RF sensor adopted, the sampling rate, f_s, can be varied according to the needs. On the one hand, a high sampling frequency can guarantee a larger signal bandwidth at the cost of an increasing computational rate required to process the samples. On the other hand, a low sampling rate may alleviate the computational burden but result in poor detection performance.

To detect changes in the propagation environment, we need to compare the received packet samples (or their frequency representation) in the presence and absence of the anomaly. To facilitate this task, in principle, we should have the same transmitted SoOp in both situations so that any change in the received signal characteristics can be ascribed to a change in the channel. In the IEEE 802.11 standard, the AP sends beacons at regular intervals. Beacons are special packets repeated over time, which contain the same information, e.g., the network service set identifier (SSID). Our idea is to exploit beacon packets as sources of opportunity for our anomaly detector. In literature, the most common approaches for intrusion detection are based on RSS estimation. A non-parametric kernel-function based anomaly detection has been used in several applications, such as [99], [92]. These methods do not require to have the same transmitted SoOp because RSS estimation reflects the energy superposition of multiple paths of the signal. In section 8.4, the performance of the RSS-based method (named RASID) proposed in [99] is compared to our novel solution based on PCA.

As shown in Fig. 8.2, the first step in the proposed approach is to extract beacon packets from the observed samples. In order to keep the computational burden low, it is possible to use non-coherent detection of beacons starting from the envelope of the received samples, $|r_i|$, $i \in \mathbb{N}$. After beacon detection from the ultra-dense over-the-air packet flow, N_{P} samples are extracted. In particular, denoting with N_{B} the number of beacons detected, we have

$$\mathbf{b}_j = \{r_i\}_{i=\mathrm{ToA}_j}^{\mathrm{ToA}_j+N_{\mathrm{P}}} \quad \text{with} \quad j = 1, \ldots, N_{\mathrm{B}} \tag{8.1}$$

where $\mathbf{b}_j$ is a vector containing the first N_{P} samples of the j-th beacon while ToA_j is the $j-$th beacon time of arrival. We want to emphasize that precise

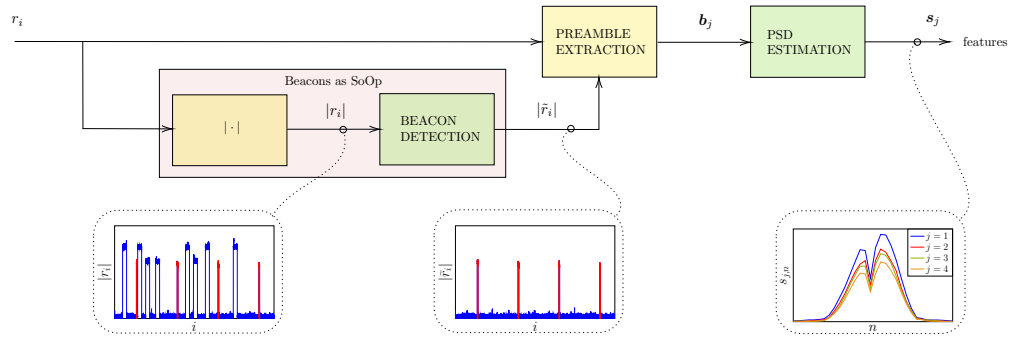

Figure 8.2: Illustration of the pre-processing data chain to extract features from beacon packets.

ToA estimation is not necessary as the detection algorithm proposed is rather insensitive to time shifts and therefore to ToA uncertainty. Note that when N_{P} is relatively small, only a fraction of the beacon is captured, e.g., its preamble.

8.2.2 Features extraction

Starting from the basic idea that any change in the environment reflects on the radio channel response, we estimate the PSD of the received beacon signals and train two well-known ML algorithms for change detection.

The PSD is estimated from the received samples $\{\mathbf{b}_j\}_{j=1}^{N_{\mathrm{B}}}$ using WOSA with Hanning window and 50% overlap. The segment length $D < N_{\mathrm{P}}$ used in WOSA is a parameter that can be tuned to trade-off between frequency resolution and PSD estimation accuracy (D is the FFT length). The estimated PSDs $\{\mathbf{s}_j\}_{j=1}^{N_{\mathrm{B}}}$ are then organized in a matrix $\mathbf{S}$ of size $N_{\mathrm{B}} \times D$. To further mitigate the impact of outliers in training the algorithms, PSDs have been averaged over M points and organized in vectors $\mathbf{x}_k$, as

$$\mathbf{x}_k = \frac{1}{M} \sum_{j=(k-1)M+1}^{kM} \mathbf{s}_j \quad \text{with} \quad k = 1, \ldots, N \tag{8.2}$$

where N is the number of points after the averaging operation, i.e., $N = N_{\mathrm{B}}/M$. For a given M it is possible to estimate the time needed for target

detection as

$$T_{\mathrm{D}} = \frac{M}{B_{\mathrm{R}} E_{\mathrm{C}}} \tag{8.3}$$

where B_{R} is the beacon rate (number of beacons per second), and E_{C} represents the number of beacons extracted over the number of beacons transmitted.[1] Finally, the features are organized in the matrix

$$\mathbf{X} = [\mathbf{x}_1^{\mathrm{T}}, \mathbf{x}_2^{\mathrm{T}}, \ldots, \mathbf{x}_N^{\mathrm{T}}]^{\mathrm{T}} \tag{8.4}$$

of size $N \times D$, where, from now on, N is the number of input points and D is their dimension. Note that if $M = 1$ then $\mathbf{X} = \mathbf{S}$.

8.3 Survey of ML Techniques

In this section, we briefly review PCA and KPCA which are usually adopted for classification problems, especially for pattern recognition and anomaly detection. Both of them need a zero-mean training set $\mathbf{X}_{\mathrm{m}}$, that must be obtained subtracting the mean value $\mathbf{m} = \mathbf{1}_N \otimes [m_1, m_2, ..., m_D]$, calculated as follows

$$m_d = \frac{1}{N} \sum_{n=1}^{N} x_{n,d} \quad \text{with} \quad d = 1, 2, ..., D \tag{8.5}$$

and therefore evaluated as $\mathbf{X}_{\mathrm{m}} = \mathbf{X} - \mathbf{m}$.

8.3.1 Principal Component Analysis

PCA is a widely known algorithm in exploratory data analysis. Given the centered training set $\mathbf{X}_{\mathrm{m}}$, the algorithm remaps the training data from the feature space $\mathbb{R}^D$ in a subspace $\mathbb{R}^P$ (where $P < D$ is the number of principal components selected) that minimize the information loss between the projected data and the original ones. The best subspace over which to project the data depends on the training set distribution and the number of components selected P. The algorithm starts evaluating the sample covariance

[1] Beacon extraction may fail because of collisions so the actual fraction of beacons selected can be less than one.

matrix $\mathbf{C}$ of the training set as

$$\mathbf{C} = \frac{\mathbf{X}_{\mathrm{m}}^{\mathrm{T}}\mathbf{X}_{\mathrm{m}}}{N-1}. \tag{8.6}$$

The matrix $\mathbf{C}$ can be factorised by eigenvalue decomposition as

$$\mathbf{C} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{\mathrm{T}} \tag{8.7}$$

where $\mathbf{V}$ is an orthonormal matrix whose columns are the eigenvectors, and $\mathbf{\Lambda}$ is a diagonal matrix that contains the D eigenvalues. The eigenvalues magnitude represent the importance of a particular component, hence selecting the P eigenvectors corresponding to the P highest eigenvalues, we obtain the best P-dimensional linear subspace over which to project the data. To do this, we multiply the portion of eigenvector matrix selected

$$\mathbf{V}_P = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_P] \tag{8.8}$$

by the data that we want to project

$$\mathbf{X}_P = \mathbf{X}\mathbf{V}_P. \tag{8.9}$$

An illustration of the data projected in a two-dimensional subspace is reported in Fig. 8.3. For each scenario, the blue points represent the features extracted in the empty room case, while the red points refer to features observed in the presence of a target. It is easy to see that as the two groups of points are overlapped, anomaly detection can be challenging.

8.3.2 Kernel Principal Component Analysis

This approach takes inspiration by the standard PCA and overcomes the limitation of the linear mapping that corresponds to finding linear boundaries in the original feature space. In many applications, this constraint represents a severe limitation and can sharply decrease the classification accuracy. KPCA firstly maps the data with a non-linear function, after which applies

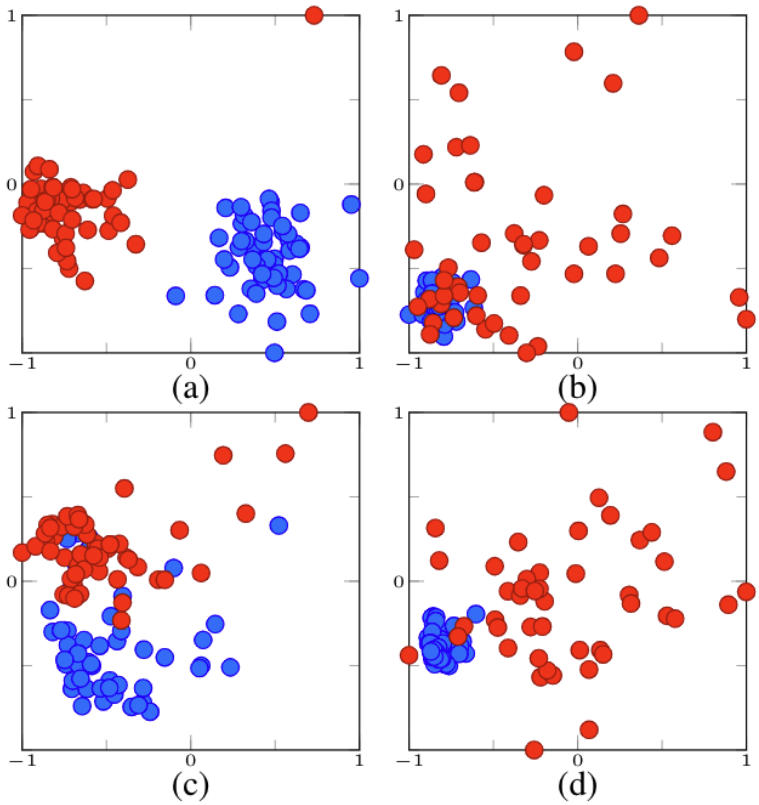

Figure 8.3: Examples of PCA outputs, in the four scenarios, when $P = 2$ principal components are selected. Blue circles refer to empty room points while red circles refer to the presence of a target in the room.

the standard PCA to find a linear boundary in the new feature space. Such boundary becomes non-linear, going back to the original feature space. A crucial point in KPCA is the selection of a non-linear function that leads to linearly separable data in the new feature space. In literature, when the data distribution is unknown, the RBF kernel is often proposed as the right candidate to accomplish this task [43]. Suppose we have a generic point $\mathbf{z}$ that corresponds to a vector of length D, we can apply the RBF as follows

$$K_{\mathbf{z}j} = e^{-\gamma||\mathbf{z}-\mathbf{x}_j||^2} \quad \text{with} \quad j = 1, 2, \ldots, N \tag{8.10}$$

where γ is a kernel parameter (inversely proportional to the width of the Gaussian function) that must be appropriately set, and $K_{\mathbf{z}j}$ is the j-th com-

ponent of the point $\mathbf{z}$ in the kernel space. Overall the starting vector $\mathbf{z}$ is mapped in a vector $\mathbf{K_z}$ of length N. Applying now the PCA described in section 8.3.1 to the new data set obtained remapping all the training points, it is possible to find non-linear boundaries in the starting feature space for a better classification. It is good practice to center the points mapped with the RBF because the mapping in the new feature space could be non-zero mean.

8.4 Numerical Results

In this section, we present several tests intending to select the best parameters setting in different working conditions and compare the performance of the algorithms.

The RF sensor is represented by the software defined radio (SDR) device *HackRF One* operating in receiving mode in the 2.4 GHz ISM band with a bandwidth of 20 MHz which corresponds to $f_s = 20\,\text{MS/s}$. The sensor output is composed by the in-phase and the in-quadrature baseband signals, each one represented with 8 bit/sample. In the setting considered $B_R = 10\,\text{packets/s}$ and $E_C = 0.83$. The scenario is represented by a room of size $4.8\times4.8\times2.8$ m^3 and the target considered is a person of 1.78 m height and 65 kg weight.

The next subsections present the accuracy varying the number of averaged points M, used to make the detection in two different working conditions and compare the performance also with the RSS-based technique [99], in two cases, $M = 2$ and $M = 8$, respectively.[2] The number of features is set to $D = 32$, which corresponds to the number of frequency bins. In all the experiments, the data are partitioned in the following sets:

- **Training:** composed of 50% of the data extracted in the empty room case;
- **Validation:** composed of 25% of the empty room points and with the same number of adversarial points (the adversarial class depends on

[2]The accuracy is defined as the number of correct classifications over the number of points classified.

the working condition and the target type);

- **Test:** composed of the remaining empty room points and the same number of adversarial points.

In this configuration, training, validation, and test sets have the same number of points, $N = 400$, when averaging is not performed ($M = 1$). For each simulation point, 100 Monte Carlo iterations are performed, and the data partitioning is repeated, selecting random points in each iteration. Given a generic point $\mathbf{z}$ and its reconstructed version $\widetilde{\mathbf{z}} = \mathbf{z}_P \mathbf{V}_P^{\mathrm{T}}$ from its projection $\mathbf{z}_P$, the error function e_z is defined as

$$e_z = ||\mathbf{z} - \widetilde{\mathbf{z}}||. \tag{8.11}$$

Evaluating the error function over the validation set, we selected a threshold to ensure a false alarm (FA) probability lower than 0.01 on the training set.

8.4.1 AP-sensor link in line-of-sight

Firstly, the scenarios showed in Fig. 8.1a and Fig. 8.1b, where the AP and the RF sensor are in LOS, are analyzed. In these scenarios, the PCA, KPCA, and RSS-based algorithms are trained varying the number of beacons M considered for the classification of one point. For each value of M the number of components P of the PCA algorithm is varied from 1 to $D - 1$, and the accuracy is tested on the validation set. Hereafter, the value of P that maximizes the validation accuracy with a particular value of M is used to evaluate the performance on the test set. The same procedure is taken for the KPCA to select the best configuration of the parameters P and γ (the kernel parameter is varied from 5 to 40 with steps of 5). The parameters used are listed in Table 8.2. A Gaussian kernel for RSS-based method has been chosen, and the normal distribution approximation for the bandwidth has been adopted [100]. The RSS has been calculated over the acquired beacon samples. As shown in Fig. 8.4 both PCA and KPCA provide comparable

Table 8.2: PCA and KPCA parameters for the considered scenarios.

			M											
			1	2	3	4	5	6	7	8	9	10	11	12
(a)	PCA	P	4	2	1	1	1	1	1	1	1	1	1	1
	KPCA	P	6	2	1	1	2	2	2	2	2	2	2	2
		γ	30	10	5	5	5	5	5	5	5	5	5	5
(b)	PCA	P	4	2	1	1	1	1	1	1	1	1	1	1
	KPCA	P	10	2	1	1	2	2	2	2	2	2	2	2
		γ	25	20	15	15	10	10	5	5	5	5	5	5
(c)	PCA	P	2	1	1	1	1	1	1	1	1	1	1	1
	KPCA	P	4	4	3	3	2	2	2	2	2	2	2	1
		γ	10	10	10	20	10	10	15	5	15	5	25	5
(d)	PCA	P	2	1	1	1	1	1	1	1	1	1	1	1
	KPCA	P	4	3	3	3	2	2	2	2	2	2	1	2
		γ	10	15	10	10	5	10	10	5	10	5	20	10

performance for values of M greater than 3, that corresponds to data acquisition of 0.4 s and overcome the accuracy of the RSS-based technique both in scenarios (a) and (b). For lower values of M, the KPCA presents higher accuracy with respect to PCA.

8.4.2 AP-sensor link in non-line-of-sight

In the NLOS case depicted in Fig. 8.1c and Fig. 8.1d, an obstacle between the AP and the RF sensor is present. Even in this scenario PCA, KPCA, and RSS-based method are trained and optimized with the same procedure described in the previous subsection. As we can see from Fig. 8.5, the performances are different from the LOS case. In this configuration, the PCA

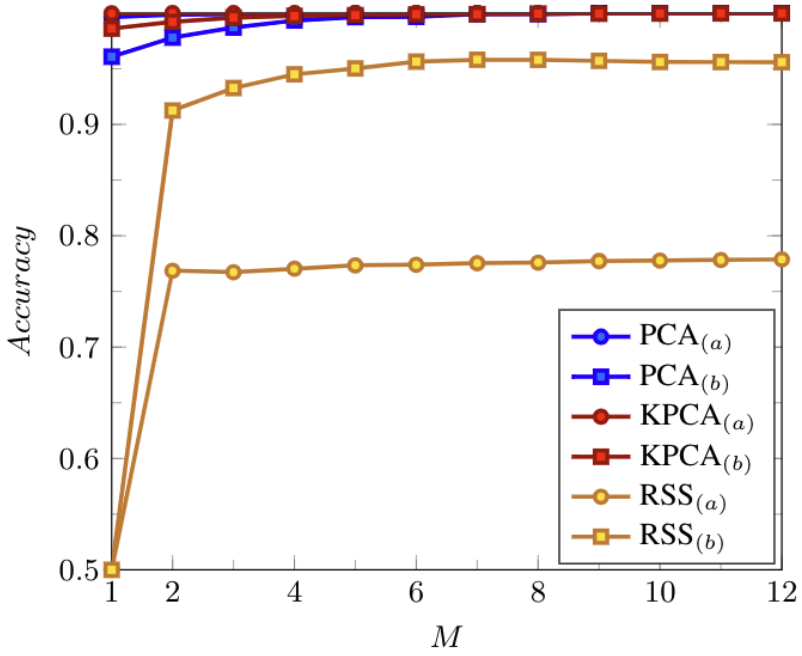

Figure 8.4: Accuracy in scenarios (a) and (b), varying M.

accuracy with static target slightly decrease, instead KPCA works well also for low M. For $M = 2$ in the case of a static object, we experience a remarkable performance degradation for PCA while KPCA preserves high accuracy. This effect is not present in the case of a moving target. This happens because the absence of a direct path between the transmitter and the RF sensor makes the channel transfer function strongly influenced by the target that intersects the reflected paths. Therefore, the detection of a moving target, for low values of M, is favored by a NLOS scenario: presumably, the intersection of a reflected path is more likely to happen than the intersection of the direct path in the LOS case. Even in these scenarios, the RSS-based technique accuracy is overwhelmed by the proposed approaches, particularly for low values of M.

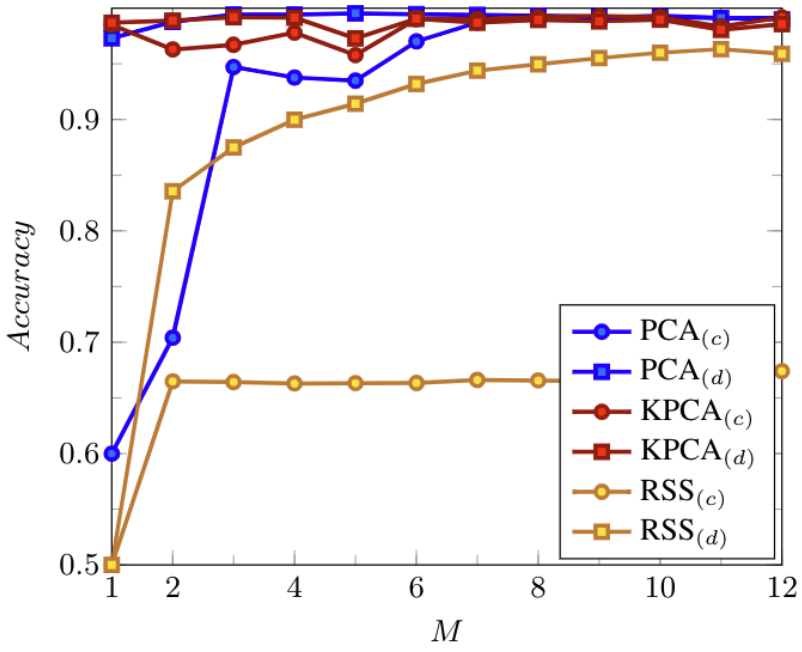

Figure 8.5: Accuracy in scenarios (c) and (d), varying M.

8.4.3 Beacon average

For the sake of selecting the best solution in each scenario, in this subsection, we compare the performance of the algorithms, with $M = 2$ when the main goal is a fast detection of the target and $M = 8$ when the objective is the maximization of the classifier accuracy. As we can see in Fig. 8.6, the KPCA algorithm provides a better solution when a timely detection is needed (blue curve), in this case, the accuracy is always greater than 95%. The worst-case for the PCA algorithm is the NLOS scenario with a static target where the accuracy goes down to 75%. In the case of $M = 8$, the accuracy generally increases in all the scenarios for all the algorithms, because the average over a higher number of points reduces the outliers and hence increases the distance between points in different classes, particularly PCA sharply increase its performance becoming comparable with the KPCA, instead the RSS-based solution experiences a saturation effect with no increment in the performance. With large values of M, the PCA performance is comparable with that of

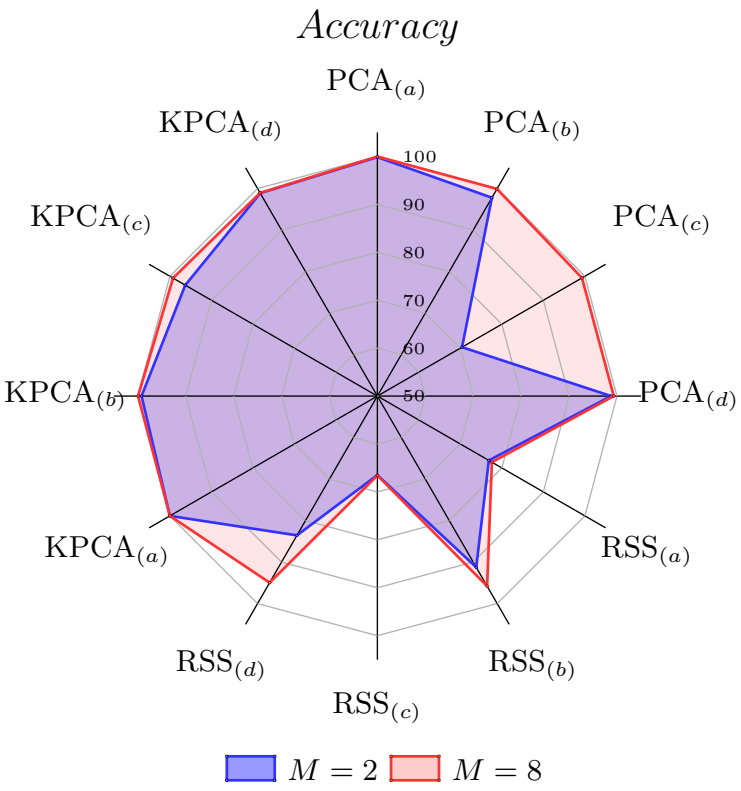

Figure 8.6: Accuracy of PCA, KPCA, and RSS-based method varying M, in the four scenarios.

KPCA in all the scenarios, suggesting the use of PCA that is less complex and computationally faster. Finally, it is also important to highlight that the PCA presents lower accuracy in the case of NLOS with respect to the LOS one when M is low. This happens because the interception of the direct path strongly changes the channel transfer function when the target is present.

8.5 Observations

In this work, we proposed and studied an RF-based automatic indoor anomaly detector exploiting SoOp from WiFi devices and machine learning techniques. The numerical results based on real waveforms, collected by a RF sensor,

demonstrated that the detection of changes in the environment is possible and that it does not require expensive devices. Moreover, the accuracy of PCA reaches 90% at relatively low observation intervals (i.e., 0.4 s), while for KPCA the observation of one beacon is enough to ensure an accuracy greater than 95%. Finally, numerical results show that the proposed approaches exhibit superior performance than the RSS-based solution in terms of accuracy, even using only one RF sensor.

Chapter 9

Conclusion

With this thesis, we proposed an automated approach to perform anomaly detection for SHM. Several ML algorithms have been tested and compared on a widely known dataset, the *Z-24* accelerometric measurements.

More in detail, in chapter 2, we provided an overview of several machine learning algorithms and their general applications (regression, classification, dimensionality reduction) with a particular focus on anomaly detection. In this branch of ML, we contributed with two new anomaly detectors named OCCNN and OCCNN^2 that in low dimensional feature spaces provide better performance compared to the classic solutions available in the literature. Both approaches exploit the flexibility of the NN classifiers in the anomaly detection setting. Still, the second one overcame the limitation of OCCNN related to the density estimation of the points in the feature space.

In chapter 3, we presented some important SHM algorithms and strategies able to extract modal parameters from the input data. We focused our efforts on the output-only model-free approaches to extend our strategy to different kinds of structures without any assumption about materials or models. The goal was to develop a blind system that does not need prior knowledge about the structure, except the accelerometer data.

In chapter 4, we presented how to apply the ML algorithms described in chapter 2 in the context of SHM. To demonstrate the proposed solutions' effectiveness, we considered a widely known benchmark, the dataset

of accelerometric measurements taken from the *Z-24* bridge. In particular, we applied the SSI algorithm to the *Z-24* bridge measurements to extract the structure's modal parameters. Then, we proposed a new density-based mode tracking able to extract the fundamental frequencies from the modal parameter via a Gaussian kernel. Finally, we applied the ML algorithms on the extracted frequencies to perform anomaly detection, and we proved that OCCNN^2 provides the best performance among all the algorithms tested, in terms of accuracy and F_1 score.

In chapter 5, we proposed some strategies to select the most reliable features among the ones extracted. More specifically, we compared the feature extraction technique performed by an ANN with the feature selection paradigm. We proved that the second option is the most appropriate in the SHM environment, where the fundamental frequencies are already significant features. Moreover, we demonstrated that noisy modal frequencies could deteriorate the anomaly detector performance; hence using only the most reliable features is recommended in this scenario.

In chapter 6, we investigated the minimum operational condition that can be adopted to ensure some target performance, with the aim to reduce the amount of data stored by the network during monitoring. We showed that it is possible to reduce the number of bits used to encode the accelerometric measurements, the number of sensors used to monitor the structure, and the acquisition time for each measurement. In particular, the third solution is the one that reduces the amount of data that must be stored more significantly. Still, when the aim is to reduce the network complexity or the sensors cost, the other two solutions can be a good compromise between complexity and data volume.

A related work that exploits some of the signal processing tools developed during the research is presented in chapter 7. Such work consists of classifying the activities performed by a human being from accelerometric data collected through a biaxial geophone located in its vicinity. Finally, in chapter 8, we propose anomaly detection tools in the radio-frequency domain to exploit over-the-air SoOp to assess the presence of a non-collaborative intruder in an indoor scenario. While deviating a little bit from the primary research

area, these two last chapters present original results that originated from it.

This thesis presents the endeavor to explore the use of ML algorithms in the area of SHM. While most of the results are based on the *Z-24* database, the extension of the proposed solutions to a wide class of structures appears feasible and supported by the analysis of their robustness. Therefore, an interesting direction to pursue is applying and validating the designed methodologies to other structures. Moreover, since deep learning is a growing area capturing the attention of many researchers worldwide, the application of deep learning in this context seems a thriving research area that could lead to attractive solutions.

Questions Recall

Q1: *How can we use tools for dimensionality reduction to perform anomaly detection?*

A1: Dimensionality reduction techniques can always be used to accomplish anomaly detection tasks. The strategy maps the data in a low dimensional feature space and remaps them in the original one. This procedure introduces a reconstruction error; depending on its magnitude, a point can be defined as standard or anomalous.

Q2: *What are the most effective algorithms able to perform anomaly detection?*

A2: It is not possible to define the best algorithm to perform anomaly detection. The performance of the algorithms depends on the data structure; for instance, some algorithms work well with a large number of points in a low dimensional feature space. On the contrary, other strategies achieve good performances with a low number of points and high dimensionality. In general, there are algorithms more flexible than others, which can provide good results in a wide set of problems.

Q3: *Is it possible to use classical NN structures (instead of the ANNs) to perform anomaly detection?*

A3: Yes, it is. In this thesis, we propose the OCCNN and OCCNN^2 algorithms that exploit the NN flexibility in the anomaly detection task. These algorithms generate adversarial dummy points to train the anomaly detector as a two-class NN classifier.

Q4: *How can we perform anomaly detection based on machine learning tools to detect damage in a structure?*

A4: Anomaly detection can be applied in several scenarios, one of this is SHM. To achieve good performance, it is necessary to effectively extract damage-sensitive features from the sensors' measurements installed on the structure.

Q5: *Is there an effective technique to extract features able to highlight damage in a structure?*

A5: Yes, there is. In this thesis, we focused our efforts on processing accelerometric measurements gathered by sensors installed on bridges. In order to maintain generality, we decide not to use strategies that require a model of the monitored structure; hence we only exploited information extracted by the data (model-free or data-driven strategies). Among several approaches offered by the literature, we decide to use SSI that satisfies all the requirements described above.

Q6: *Is it possible to perform frequencies tracking automatically after the fundamental frequencies extraction?*

A6: Yes, it is. In this thesis, we propose several strategies to accomplish this task. We proposed a novel density-based time-domain tracking algorithm to exploit the time correlation between consecutive measurements, filtering new measurements with different kernels (linear and Gaussian). These strategies provide satisfactory and reliable tracking results.

Q7: *How can we test these strategies? Is there in literature a reference structure with accelerometric measurements both in standard and in damaged conditions?*

A7: The crucial point is to find an actual structure with an extensive database of measurements both in standard and in damaged condition to test the algorithm performance. Fortunately, the *Z-24* bridge, a known benchmark in literature, is a real structure whose accelerometric and environmental measurements are available online both in standard and damaged conditions. We decided to test our algorithms on this structure.

Q8: *Is it possible to automatically select the most reliable fundamental frequencies, among all the ones extracted, to reduce the problem dimensionality without affecting the performance?*

A8: Yes, it is. We proposed several metrics to accomplish this task, and we showed that variance and skewness of the extracted frequencies could be effectively used to determine the most reliable frequencies among all the extracted ones.

Q9: *Is it possible to further reduce the amount of data that the network must store?*

A9: Yes, it is. In the thesis, we proposed three different strategies to reduce the amount of data stored without effect the anomaly detectors performance: *i*) reduce the number of sensors used to monitor the structure; *ii*) reduce the number of resolution bits of the sensors; *iii*) reduce the acquisition time for each measurement.

Q10: *Is there a possibility to reduce the network infrastructure costs, reducing the number of sensors necessary to detect damage, or using low-cost sensors?*

A10: Yes, it is. Experimental results showed that when the objective is to reduce the network infrastructure costs, the best strategy is to reduce the number of sensors using a low number of accurate accelerometers to achieve good performance. Instead, when the goal is to reduce sensors cost, a large network of low-cost sensors should be the best solution to maintain high anomaly detection capability.

Q11: *Is it possible to exploit accelerometric measurements acquired by sensors installed on a structure also to infer human activities?*

A11: Yes, it is. As shown in chapter 7, measurements gathered by seismic sensors installed on the ground can be used to infer human activities with high accuracy.

Q12: *Is it possible to apply similar techniques to solve other problems?*

A12: Yes, it is. As said before, ML and anomaly detection techniques are flexible strategies that can be used in different fields. For instance, in chapter 8, we propose an application that exploits WiFi signals of opportunity to detect the presence of a target in an environment through anomaly detection algorithms. Experimental results show that this approach works pretty well and provides very good results in terms of accuracy.

Publications

1 E. Favarelli, A. Giorgetti, "Machine Learning for Automatic Processing of Modal Analysis in Damage Detection of Bridges," *IEEE Transactions on Instrumentation and Measurement (TIM)*, 2020.

2 E. Favarelli, E. Testi, A. Giorgetti, "Data Management in Structural Health Monitoring," *8th Civil Structural Health Monitoring Workshop (CSHM-8)*, Online, Mar. 29 - 31, 2021.

3 E. Favarelli, E. Testi, A. Giorgetti, "Dimensionality Reduction in Modal Analysis for Structural Health Monitoring," *International Conference on Smart Urban Infrastructures and Artificial Intelligence Technologies (ICSUIAIT)*, Rome, Italy, Feb. 18-19, 2021.

4 E. Testi, L. Pucci, E. Favarelli, A. Giorgetti, "Blind Traffic Classification in Wireless Networks," *European Signal Processing Conference (EUSIPCO)*, Amsterdam, Jan. 18-21, 2021.

5 E. Testi, E. Favarelli, A. Giorgetti, "Blind Source Separation for Wireless Networks: a Tool for Topology Sensing," *EAI International Conference on Cognitive Radio Oriented Wireless Networks (CROWNCOM)*, Online, Nov. 25-26, 2020.

6 E. Testi, E. Favarelli, A. Giorgetti, "Reinforcement Learning for Connected Autonomous Vehicle Localization via UAVs," *IEEE International Workshop on Metrology for Agriculture and Forestry*, Online, Nov. 4-6, 2020.

7 L. Pucci, E. Testi, E. Favarelli, A. Giorgetti, "Human Activities Classification Using Biaxial Seismic Sensors," *IEEE Sensors Letters*, vol. 4, no. 10, pp. 1-4, Oct. 2020.

8 E. Favarelli, E. Testi, A. Giorgetti, "One Class Classifier Neural Network for Anomaly Detection in Low Dimensional Feature Spaces," *International Conference on Signal Processing and Communication Sys-*

tems (ICSPCS), Surfers Paradise, Gold Coast, Australia, Dec. 16-18, 2019.

9 E. Favarelli, E. Testi, L. Pucci, M. Chiani, A. Giorgetti, "Anomaly Detection Using WiFi Signals of Opportunity," *International Conference on Signal Processing and Communication Systems (ICSPCS)*, Surfers Paradise, Gold Coast, Australia, Dec. 16-18, 2019.

10 E. Testi, E. Favarelli, L. Pucci, A. Giorgetti, "Machine Learning for Wireless Network Topology Inference," *International Conference on Signal Processing and Communication Systems (ICSPCS)*, Surfers Paradise, Gold Coast, Australia, Dec. 16-18, 2019.

11 E. Testi; E. Favarelli; A. Giorgetti, "Machine Learning for User Traffic Classification in Wireless Systems," in *European Signal Processing Conference (EUSIPCO)*, pp. 1-5, Rome, Italy, Sep. 3-7, 2018.

Bibliography

[1] D. Brunelli, C. Moser, L. Thiele, and L. Benini, "Design of a solar-harvesting circuit for batteryless embedded systems," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 56, no. 11, pp. 2519–2528, 2009.

[2] D. Brunelli, L. Benini, C. Moser, and L. Thiele, "An efficient solar energy harvester for wireless sensor nodes," in *2008 Design, Automation and Test in Europe*, 2008, pp. 104–109.

[3] A. S. Weddell, M. Magno, G. V. Merrett, D. Brunelli, B. M. Al-Hashimi, and L. Benini, "A survey of multi-source energy harvesting systems," in *2013 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2013, pp. 905–908.

[4] D. Carli, D. Brunelli, D. Bertozzi, and L. Benini, "A high-efficiency wind-flow energy harvester using micro turbine," in *SPEEDAM 2010*, 2010, pp. 778–783.

[5] D. Pianini, A. Elzanaty, A. Giorgetti, and M. Chiani, "Emerging distributed programming paradigm for cyber-physical systems over lorawans," in *IEEE Globecom Workshops (GC Wkshps)*, 2018, pp. 1–6.

[6] A. Elzanaty, A. Giorgetti, and M. Chiani, "Lossy compression of noisy sparse sources based on syndrome encoding," *IEEE Transactions on Communications*, vol. 67, no. 10, pp. 7073–7087, 2019.

[7] S. Sindaco, S. Nanni, C. Aguzzi, L. Roffia, and T. Salmon Cinotti, "Enabling context aware tuning of low power sensors for smart agriculture,"

in *2020 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor)*, 2020, pp. 114–118.

[8] F. G. Brundu, E. Patti, A. Osello, M. D. Giudice, N. Rapetti, A. Krylovskiy, M. Jahn, V. Verda, E. Guelpa, L. Rietto, and A. Acquaviva, "IoT software infrastructure for energy management and simulation in smart cities," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 2, pp. 832–840, 2017.

[9] E. Lattanzi, E. Regini, A. Acquaviva, and A. Bogliolo, "Energetic sustainability of routing algorithms for energy-harvesting wireless sensor networks," *Computer Communications*, vol. 30, no. 14, pp. 2976 – 2986, 2007.

[10] M. Chiani and A. Elzanaty, "On the LoRa modulation for IoT: Waveform properties and spectral analysis," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8463–8470, 2019.

[11] B. Al Homssi, A. Al-Hourani, S. Chandrasekharan, K. M. Gomez, and S. Kandeepan, "On the bound of energy consumption in cellular IoT networks," *IEEE Transactions on Green Communications and Networking*, vol. 4, no. 2, pp. 355–364, 2020.

[12] B. Al Homssi, A. Al-Hourani, K. G. Chavez, S. Chandrasekharan, and S. Kandeepan, "Energy-efficient IoT for 5G: A framework for adaptive power and rate control," in *12th International Conference on Signal Processing and Communication Systems (ICSPCS)*, 2018, pp. 1–6.

[13] L. Sciullo, C. Aguzzi, M. Di Felice, and T. S. Cinotti, "WoT store: Enabling things and applications discovery for the W3C web of things," in *16th IEEE Annual Consumer Communications Networking Conference (CCNC)*, 2019, pp. 1–8.

[14] G. Fabbrocino and C. Rainieri, *Operational modal analysis of civil engineering structures*. New York: Springer-verlag, May 2014.

[15] R. M. Azzara, G. D. Roeck, M. Girardi, C. Padovani, D. Pellegrini, and E. Reynders, "The influence of environmental parameters on the dynamic behaviour of the San Frediano bell tower in Lucca," *Engineering Structures*, vol. 156, pp. 175–187, 2018.

[16] E. Reynders, G. Wursten, and G. D. Roeck, "Output-only structural health monitoring in changing environmental conditions by means of nonlinear system identification," *Structural Health Monitoring*, vol. 13, no. 1, pp. 82–93, 2014.

[17] Z-24 bridge data. [Online]. Available: https://bwk.kuleuven.be/bwm/z24

[18] A. Al-Hourani, S. Chandrasekharan, and S. Kandeepan, "Path loss study for millimeter wave device-to-device communications in urban environment," in *IEEE international conference on communications workshops (ICC)*, 2014, pp. 102–107.

[19] K. Sithamparanathan and A. Giorgetti, *Cognitive radio techniques: spectrum sensing, interference mitigation, and localization.* Artech house, 2012.

[20] W. Ejaz, M. Naeem, M. Basharat, A. Anpalagan, and S. Kandeepan, "Efficient wireless power transfer in software-defined wireless sensor networks," *IEEE Sensors Journal*, vol. 16, no. 20, pp. 7409–7420, 2016.

[21] S. Chandrasekharan, K. Gomez, A. Al-Hourani, S. Kandeepan, T. Rasheed, L. Goratti, L. Reynaud, D. Grace, I. Bucaille, T. Wirth, and S. Allsopp, "Designing and implementing future aerial communication networks," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 26–34, 2016.

[22] B. Al Homssi, A. Al-Hourani, R. J. Evans, K. G. Chavez, S. Kandeepan, W. S. T. Rowe, and M. Loney, "Free spectrum for IoT: How much can it take?" in *IEEE International Conference on Communications Workshops (ICC Workshops)*, 2018, pp. 1–6.

[23] V. Yadav, L. Kumar, and P. Kumar, "Evolution and development of wireless communication system," in *International Conference on Computing, Power and Communication Technologies (GUCON)*, 2019, pp. 53–57.

[24] G. D. Roeck, "The state-of-the-art of damage detection by vibration monitoring: the SIMCES experience," *Journal of Structural Control*, vol. 10, no. 2, pp. 127–134, May 2003.

[25] K. Worden, C. Farrar, J. Haywood, and M. Todd, "A review of nonlinear dynamics applications to structural health monitoring," *Structural Control and Health Monitoring*, vol. 15, no. 4, pp. 540–567, Jul. 2008.

[26] F. Zonzini, C. Aguzzi, L. Gigli, L. Sciullo, N. Testoni, L. De Marchi, M. Di Felice, T. S. Cinotti, C. Mennuti, and A. Marzani, "Structural health monitoring and prognostic of industrial plants and civil structures: A sensor to cloud architecture," *IEEE Instrumentation Measurement Magazine*, vol. 23, no. 9, pp. 21–27, 2020.

[27] F. Zonzini, A. Girolami, L. De Marchi, A. Marzani, and D. Brunelli, "Cluster-based vibration analysis of structures with GSP," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 4, pp. 3465–3474, 2021.

[28] F. Zonzini, M. M. Malatesta, D. Bogomolov, N. Testoni, A. Marzani, and L. De Marchi, "Vibration-based SHM with upscalable and low-cost sensor networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 10, pp. 7990–7998, 2020.

[29] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer Verlag, Aug. 2006.

[30] J. Watt, R. Borhani, and A. K. Katsaggelos, *Machine Learning Refined*. Cambridge University Press, 2016.

[31] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[32] D. Y. Harvey and M. D. Todd, "Structural health monitoring feature design by genetic programming," *Smart Materials and Structures*, vol. 23, no. 9, p. 095002, jul 2014.

[33] A. Santos, M. Silva, C. Sales, J. Costa, and E. Figueiredo, "Applicability of linear and nonlinear principal component analysis for damage detection," in *IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings*, Pisa, Italy, May 2015, pp. 869–874.

[34] E. Favarelli, E. Testi, L. Pucci, and A. Giorgetti, "Anomaly detection using WiFi signals of opportunity," in *13th International Conference on Signal Processing and Communication Systems (ICSPCS)*, Surfers Paradise, Gold Coast, Australia, Dec. 2019, pp. 1–7.

[35] E. Testi, E. Favarelli, and A. Giorgetti, "Machine learning for user traffic classification in wireless systems," in *26th European Signal Processing Conference (EUSIPCO)*, Rome, Italy, Sep. 2018, pp. 2040–2044.

[36] E. Favarelli, E. Testi, and A. Giorgetti, "One class classifier neural network for anomaly detection in low dimensional feature spaces," in *13th International Conference on Signal Processing and Communication Systems (ICSPCS)*, Surfers Paradise, Gold Coast, Australia, Dec. 2019, pp. 1–7.

[37] E. Testi, E. Favarelli, L. Pucci, and A. Giorgetti, "Machine learning for wireless network topology inference," in *13th International Conference on Signal Processing and Communication Systems (ICSPCS)*, Gold Coast, Australia, Dec. 2019.

[38] P. Perera and V. M. Patel, "Learning deep features for one-class classification," *IEEE Transactions on Image Processing*, vol. 28, no. 11, pp. 5450–5463, Nov. 2019.

[39] R. Chalapathy, A. K. Menon, and S. Chawla, "Anomaly detection using one-class neural networks," *Computing Research Repository (CoRR)*, Aug. 2018.

[40] P. Schlachter and B. Yang, "Active learning for one-class classification using two one-class classifiers," in *26th European Signal Processing Conference (EUSIPCO)*, Rome, Italy, Sep. 2018, pp. 1197–1201.

[41] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[42] B. Schölkopf, A. Smola, and K.-R. Müller, "Kernel principal component analysis," in *Proceedings International Conference on Artificial Neural Networks*, vol. 1327, no. 6. Lausanne, Switzerland: Springer, Oct. 1997, pp. 583–588.

[43] B. Schölkopf, A. Smola, E. Smola, and K.-R. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Computation*, vol. 10, pp. 1299–1319, Jul. 1998.

[44] A. Santos, E. Figueiredo, M. Silva, R. Santos, C. Sales, and J. C. W. A. Costa, "Genetic-based EM algorithm to improve the robustness of gaussian mixture models for damage detection in bridges," *Structural Control and Health Monitoring*, vol. 24, no. 3, p. e1886, May 2017.

[45] J. H. Pollard, "On distance estimators of density in randomly distributed forests," *Biometrics*, vol. 27, no. 4, pp. 991–1002, Dec. 1971.

[46] D. Brigante, C. Rainieri, and G. Fabbrocino, "The role of the modal assurance criterion in the interpretation and validation of models for seismic analysis of architectural complexes," in *Proceedings International Conference on Structural Dynamics (Eurodin)*, vol. 199, Rome, Italy, Sep. 2017, pp. 3404–3409.

[47] L. Calandrino and M. Chiani, *Lezioni di Comunicazioni Elettriche*. Pitagora, 2013.

[48] C. Wu, H. Liu, X. Qin, and J. Wang, "Stabilization diagrams to distinguish physical modes and spurious modes for structural parameter

identification," *Journal of Vibroeng.*, vol. 19, no. 4, pp. 2777–2794, Jun. 2017.

[49] A. Cabboi, F. Magalhães, C. Gentile, and Á. Cunha, "Automated modal identification and tracking: Application to an iron arch bridge," *Structural Control and Health Monitoring*, vol. 24, no. 1, p. e1854, Feb. 2017.

[50] M. Pastor, M. Binda, and T. Harčarik, "Modal assurance criterion," *Procedia Engineering*, vol. 48, pp. 543–548, 2012.

[51] E. Reynders, J. Houbrechts, and G. D. Roeck", "Fully automated (operational) modal analysis," *Mechanical Systems and Signal Processing*, vol. 29, pp. 228–250, May 2012.

[52] E. P. Carden and J. M. W. Brownjohn, "Fuzzy clustering of stability diagrams for vibration-based structural health monitoring," *Computer-Aided Civil and Infrastructure Eng.*, vol. 23, no. 5, pp. 360–372, May 2008.

[53] E. Reynders and G. D. Roeck, "Continuous vibration monitoring and progressive damage testing on the z 24 bridge," *Encyclopedia of structural health monitoring*, vol. 10, pp. 127–134, May 2009.

[54] (2001) FBA11 datasheet. [Online]. Available: ftp://aplica.at/Manuals/Sensors/FBA11.pdf

[55] (1998) ADC488 datasheet. [Online]. Available: https://www.artisantg.com/info/IOtech_ADC488_Manual.pdf

[56] G. H. Golub and C. F. V. Loan, *Matrix computations*. JHU press, 2012, vol. 3.

[57] J. J. Verbeek, N. Vlassis, and B. Kröse, "Efficient greedy learning of gaussian mixture models," *Neural computation*, vol. 15, no. 2, pp. 469–485, 2003.

[58] M. Silva, A. Santos, E. Figueiredo, R. Santos, C. Sales, and J. Costa, "A novel unsupervised approach based on a genetic algorithm for structural damage detection in bridges," *Engineering Applications of Artificial Intelligence*, vol. 52, pp. 168–180, Jun. 2016.

[59] M. Silva, A. Santos, R. Santos, E. Figueiredo, C. Sales, and J. C. Costa, "Agglomerative concentric hypersphere clustering applied to structural damage detection," *Mechanical Systems and Signal Processing*, vol. 92, pp. 196–212, Feb. 2017.

[60] G. Park, T. Rosing, M. D. Todd, C. R. Farrar, and W. Hodgkiss, "Energy harvesting for structural health monitoring sensor networks," *Journal of Infrastructure Systems*, vol. 14, no. 1, pp. 64–79, 2008.

[61] S. Taylor, K. Farinholt, E. Flynn, E. Figueiredo, D. L. Mascarenas, E. A. Moro, G. Park, M. Todd, and C. Farrar, "A mobile-agent-based wireless sensing network for structural monitoring applications," *Measurement Science and Technology*, vol. 20, no. 4, pp. 1–14, Jan. 2009.

[62] D. Mascarenas, E. Flynn, C. Farrar, G. Park, and M. Todd, "A mobile host approach for wireless powering and interrogation of structural health monitoring sensor networks," *IEEE Sensors Journal*, vol. 9, no. 12, pp. 1719–1726, 2009.

[63] A. Girolami, F. Zonzini, L. De Marchi, D. Brunelli, and L. Benini, "Modal analysis of structures with low-cost embedded systems," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2018, pp. 1–4.

[64] A. Elzanaty, A. Giorgetti, and M. Chiani, "Weak RIC analysis of finite gaussian matrices for joint sparse recovery," *IEEE Signal Processing Letters*, vol. 24, no. 10, pp. 1473–1477, 2017.

[65] E. B. Flynn and M. D. Todd, "A bayesian approach to optimal sensor placement for structural health monitoring with application to active sensing," *Mechanical Systems and Signal Processing*, vol. 24, no. 4, pp. 891 – 903, 2010.

[66] E. B. Flynn and M. Todd, "Optimal placement of piezoelectric actuators and sensors for detecting damage in plate structures," *Journal of Intelligent Material Systems and Structures*, vol. 21, no. 3, pp. 265–274, 2010.

[67] Q. Zhou, G. Tong, D. Xie, B. Li, and X. Yuan, "A seismic-based feature extraction algorithm for robust ground target classification," *IEEE Signal Processing Letters*, vol. 19, no. 10, pp. 639–642, Oct. 2012.

[68] J. Huang, Q. Zhou, X. Zhang, E. Song, B. Li, and X. Yuan, "Seismic target classification using a wavelet packet manifold in unattended ground sensors systems," *Sensors*, vol. 13, no. 7, pp. 8534–8550, Jul. 2013.

[69] S. Pan, K. Lyons, M. Mirshekari, H. Y. Noh, and P. Zhang, "Multiple pedestrian tracking through ambient structural vibration sensing," in *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems*, New York, NY, USA, Nov. 2016, pp. 366–367.

[70] J. Jackowski and R. Wantoch-Rekowski, "Classification of wheeled military vehicles using neural networks," in *18th International Conference on Systems Engineering (ICSEng)*, Las Vegas, NV, USA, Aug. 2005, pp. 212–217.

[71] B. Mukhopadhyay, S. Anchal, and S. Kar, "Detection of an intruder and prediction of his state of motion by using seismic sensor," *IEEE Sensors Journal*, vol. 18, no. 2, pp. 703–712, Jan. 2018.

[72] X. Jin, S. Sarkar, A. Ray, S. Gupta, and T. Damarla, "Target detection and classification using seismic and PIR sensors," *IEEE Sensors Journal*, vol. 12, no. 6, pp. 1709–1718, Jun. 2012.

[73] M. Zubair and K. Hartmann, "Target classification based on sensor fusion in multi-channel seismic network," in *IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, Bilbao, Spain, Dec. 2011, pp. 438–443.

[74] A. Sutin, H. Salloum, A. Sedunov, and N. Sedunov, "Acoustic detection, tracking and classification of low flying aircraft," in *IEEE International Conference on Technologies for Homeland Security (HST)*, Waltham, MA, USA, Nov. 2013, pp. 141–146.

[75] M. Civelek and A. Yazici, "Automated moving object classification in wireless multimedia sensor networks," *IEEE Sensors Journal*, vol. 17, no. 4, pp. 1116–1131, Feb. 2017.

[76] W. D. Fisher, T. K. Camp, and V. V. Krzhizhanovskaya, "Anomaly detection in earth dam and levee passive seismic data using support vector machines and automatic feature selection," *Journal of Computational Science*, vol. 20, pp. 143–153, May 2017.

[77] P. Welch, "The use of fast Fourier transform for the estimation of power spectra: A method based on time averaging over short, modified periodograms," *IEEE Transactions on audio and electroacoustics*, vol. 15, no. 2, pp. 70–73, Jun. 1967.

[78] I. Jolliffe, *Principal Component Analysis*. Springer Verlag, 1986.

[79] H. Abdi and L. J. Williams, "Principal component analysis," *WIREs Comput. Stat.*, vol. 2, no. 4, pp. 433–459, Jul. 2010.

[80] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967.

[81] R. C. Gonzalez, R. E. Woods, and S. L. Eddins, *Digital image processing using MATLAB*. Pearson Education India, 2004.

[82] K. Briechle and U. D. Hanebeck, "Template matching using fast normalized cross correlation," in *Optical Pattern Recognition XII*, vol. 4387, 2001, pp. 95–102.

[83] Y. Guo, T. Hastie, and R. Tibshirani, "Regularized linear discriminant analysis and its application in microarrays," *Biostatistics*, vol. 8, no. 1, pp. 86–100, 2007.

[84] S. Bartoletti, A. Conti, and M. Z. Win, "Device-free counting via wideband signals," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 5, pp. 1163–1174, May 2017.

[85] S. Bartoletti, A. Conti, and M. Z. Win, "Passive radar via LTE signals of opportunity," *IEEE International Conference on Communication Workshops (ICC)*, pp. 181–185, Jun. 2014.

[86] S. Bartoletti, A. Conti, and M. Z. Win, "Device-free counting via OFDM signals of opportunity," *IEEE International Conference on Communication Workshops (ICC)*, pp. 1–5, May 2018.

[87] M. Leng, W. P. Tay, C. M. S. See, S. Gulam Razul, and M. Z. Win, "Modified CRLB for cooperative geolocation of two devices using signals of opportunity," *IEEE Transactions on Wireless Communications*, vol. 13, no. 7, pp. 3636–3649, Jul. 2014.

[88] A. E. Kosba, A. Abdelkader, and M. Youssef, "Analysis of a device-free passive tracking system in typical wireless environments," in *3rd International Conference on New Technologies, Mobility and Security*, Cairo, Egypt, Dec. 2009, pp. 1–5.

[89] Y. Jin, Z. Tian, M. Zhou, Z. Li, and Z. Zhang, "A whole-home level intrusion detection system using WiFi-enabled IoT," in *14th International Wireless Communications Mobile Computing Conference (IWCMC)*, Limassol, Cyprus, Jun. 2018, pp. 494–499.

[90] I. Sabek and M. Youssef, "Multi-entity device-free WLAN localization," in *IEEE Global Communications Conference (GLOBECOM)*, Anaheim, California, Dec. 2012, pp. 2018–2023.

[91] J. Yang, Y. Ge, H. Xiong, Y. Chen, and H. Liu, "Performing joint learning for passive intrusion detection in pervasive wireless environments," in *Proceedings IEEE International Conference on Computer Communications (INFOCOM)*, San Diego, California, Mar. 2010, pp. 1–9.

[92] Y. Jin, Z. Tian, M. Zhou, Z. Li, and Z. Zhang, "An adaptive and robust device-free intrusion detection using ubiquitous WiFi signals," in *IEEE International Conference on Digital Signal Processing (DSP)*, Shanghai, China, Nov. 2018, pp. 1–5.

[93] Z. Zhou, Z. Yang, C. Wu, L. Shangguan, and Y. Liu, "Omnidirectional coverage for device-free passive human detection," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 7, pp. 1819–1829, Jul. 2014.

[94] Z. Tian, Y. Li, M. Zhou, and Z. Li, "WiFi-based adaptive indoor passive intrusion detection," in *IEEE International Conference on Digital Signal Processing (DSP)*, Shanghai, China, Nov. 2018, pp. 1–5.

[95] S. Savazzi, S. Sigg, M. Nicoli, V. Rampa, S. Kianoush, and U. Spagnolini, "Device-free radio vision for assisted living: Leveraging wireless channel quality information for human sensing," *IEEE Signal Processing Magazine*, vol. 33, no. 2, pp. 45–58, Mar. 2016.

[96] E. Cianca, M. De Sanctis, and S. Di Domenico, "Radios as sensors," *IEEE Internet of Things Journal*, vol. 4, no. 2, pp. 363–373, Apr. 2017.

[97] M. Chiani, A. Giorgetti, and E. Paolini, "Sensor radar for object tracking," *Proceedings of the IEEE*, vol. 106, no. 6, pp. 1022–1041, Jun. 2018.

[98] F. Colone, D. Pastina, P. Falcone, and P. Lombardo, "WiFi-based passive ISAR for high-resolution cross-range profiling of moving targets," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 6, pp. 3486–3501, Jun. 2014.

[99] A. E. Kosba, A. Saeed, and M. Youssef, "Rasid: A robust wlan device-free passive motion detection system," in *IEEE International Conference on Pervasive Computing and Communications*, Lugano,Swiss, Mar. 2012, pp. 180–189.

[100] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. Chapman and Hall/CRC, Apr. 1986.

Acknowledgements

There are several people I should thanks for where I am today, several people who gave me support when I wanted to give up, several people who showed me the light when the only thing I could see was the darkness.

First of all Enrico Testi, a friend, a colleague, a teammate. He was always by my side cheering in the best moments and standing next to me in the worst ones. Enrico I own you all my success and I will be with you to share all of our failures. I could not hope for a better friend to walk with me down this path, I can just hope one day I will become for you what you are for me.

My family, my mother Barbara, Massimo, and my little sister Nicole (the greatest person I know). My mother, my first fan always there, since she gave me the first Lego piece till my Ph.D. in engineering, she never failed to show me her support. Massimo, that reminds me everyday that does not matter what destiny takes from your life, you can always find a way to create your own path. My sister, Nicole, she taught me how to dream and how to fill my mind with a million dreams to create the world I vision and that I want to create. They are my safe place where to find refuge from a storm, I own you my life, my strength and my hope. Thank you.

My father, Daniele, my green light at the end of the pier, a man who had that rare smile that probably you meet once in your life, an indelible example of the person that I want to be.

All my friends and relatives that support me every day, sometimes with simple gesture, sometimes with great supports, I hope to spend as much time as possible with them and to give them back at least half of the good they gave me.

I want to thanks my co-advisor Prof. Alessandro Marzani, always present when I needed suggestions or help, he represents a great support for me.

Finally I would like to thanks my advisor and mentor Prof. Andrea Giorgetti, a great person and an illuminating professor, he always supported me during these years and I hope to continue to work with him as long as possible. I have the merit of having followed him, he has the merit of having believed in me.

Thank you all, for your trust, your support and your love. Is not always easy to believe in yourself during life, there are moments that the only thought you have in your mind is "I am not gonna make it I am not good enough" well today I finally proved to myself, that I am good enough. Thank you.

"A true master is an eternal student"