Machine Learning and Deep Learning Techniques for Colocated MIMO Radars: A Tutorial Overview

Alessandro Davoli *, Giorgio Guerzoni * and Giorgio M. Vitetta*
*Dept. of Engineering "Enzo Ferrari", University of Modena and Reggio Emilia
Email: alessandro.davoli@unimore.it, giorgio.guerzoni@unimore.it, giorgio.vitetta@unimore.it

Abstract

Radars are expected to become the main sensors in various civilian applications, ranging from health-care monitoring to autonomous driving. Their success is mainly due to the availability of both low cost integrated devices, equipped with compact antenna arrays, and computationally efficient signal processing techniques. An increasingly important role in the field of radar signal processing is played by machine learning and deep learning techniques. Their use has been first taken into consideration in human gesture and motion recognition, and in various healthcare applications. More recently, their exploitation in object detection and localization has been also investigated. The research work accomplished in these areas has raised various technical problems that need to be carefully addressed before adopting the above mentioned techniques in real world radar systems. In this manuscript, a comprehensive overview of the machine learning and deep learning techniques currently being considered for their use in radar systems is provided. Moreover, some relevant open problems and current trends in this research area are analysed. Finally, various numerical results, based on both synthetically generated and experimental datasets, and referring to two different applications are illustrated. These allow readers to assess the efficacy of specific methods and to compare them in terms of accuracy and computational effort.

Index Terms

Radar, multiple-input multiple-output, machine learning, beamforming, micro-Doppler, range-azimuth estimation.

I. Introduction

RECENT advances in the production of *monolithic microwave integrated circuits* (MMIC) have paved the way for the implementation of low cost and compact colocated *multiple-input multiple-output* (MIMO) radar devices. These devices, being equipped with transmit and receive antenna arrays, are able to detect multiple targets, and estimate their spatial coordinates and their velocity. Moreover, unlike cameras and lidars, they can operate in adverse weather and lighting conditions, guaranteeing the privacy of people that act in the surrounding propagation environment. For this reason, in recent times, substantial research efforts have been devoted to the development of new MIMO radar systems and to the assessment of their accuracy in a number of applications.

It is well known that the full exploitation of the potentialities offered by modern colocated MIMO radar devices requires the use of proper detection and estimation methods. In the last two decades, significant advances have been made in the development of *deterministic* methods accomplishing these tasks. These are mainly based on a *maximum likelihood* approach [1], [2] or on *sub-space methods*, like the *MUltiple SIgnal Classification* (MUSIC) technique [3]. Moreover, they are *model-based*, since they require the full knowledge of the employed radar device and rely on a parametric description of the propagation environment; note that, in such a description, targets are usually represented as *points* reflecting electromagnetic energy. An overview of deterministic methods is provided by refs. [4], [5], whereas some interesting applications of them can be found in refs. [6]–[8]; relevant examples of these applications include the detection and the estimation of the position of cars or pedestrians in a street [9], as well as the analysis of human vital signs [10]. In many cases, these methods allow to achieve good estimation accuracy at the price of an acceptable computational effort. Unluckily, in a number of recent applications, MIMO radars operate in extremely complex, highly dynamic and time varying scenarios, affected by multipath propagation, clutter and interference, and in the presence of *extended* targets. In such conditions, deterministic algorithms may fail, since they are unable to achieve acceptable estimation accuracy and are prone to generate ghost targets [11]. When this occurs, *machine learning* (ML) and *deep learning* (DL) techniques represent an appealing alternative or the only viable technical solution. A relevant example of this class of techniques is represented by *neural networks* (NNs) [12], [13]. These networks can automatically learn specific data patterns and extract useful information directly from raw data, even in the presence of strong interference. In fact, they can be trained to recognise interference and remove it, so making the recovery of useful signal components possible. Unfortunately, the application of NNs and related methods to MIMO radars is challenging,

because, on the one hand, the problems tackled in this field are often substantially different from those to which such methods have been applied for a number of years (e.g., processing of RGB images in computer vision); on the other hand, the large radar dataset required for the proper training of a NN may be unavailable. Another critical issue emerging from the exploitation of ML and DL methods in real world applications is represented by the fact that a trained machine is, by and large, a black box mapping inputs to outputs; for this reason, generally speaking, it cannot be inferred why a given output has been produced on the basis of its input data. This explains why, in various radar applications, a model-based approach could be preferred. Despite this relevant limitation, it is widely accepted that the use of ML and DL methods in colocated MIMO radars will allow to solve a number of real world problems. For instance, recent work has evidenced that they can be successfully exploited in the classification of human activities and gestures, in the detection of human falls [14] and in the classification of dynamic targets in dense and dynamic urban scenarios [15]. This manuscript aims at providing an overview of the ML and DL methods employed in all the above mentioned applications, analysing their pros and cons, discussing the main lessons that have been learnt from their use and illustrating some trends in this research area. As far as we know, in the technical literature, the few manuscripts offering related contributions refer to specific applications, namely human-motion recognition [14] and assisted living [10]. The scope of this work, instead, is offering a wider perspective on this research area. Furthermore, our description of learning methods is interspersed with various numerical examples on synthetically generated dataset and an entire section is devoted to the analysis of various numerical results generated the measurements acquired through a commercial colocated MIMO radar.

The remaining part of this manuscript is organized as follows. In Section II, essential information about the history of colocated MIMO radars, their architecture and some well known deterministic detection/estimation algorithms that can be employed in these radar systems are provided. The most relevant ML and DL methods currently being investigated for their use in colocated MIMO radars are described in Sections III and IV, respectively; a brief comparison among such methods is illustrated in Section V. An overview of the specific applications of these techniques to colocated MIMO radars is illustrated in Section VI, where we focus on human-motion and human-gesture classification, healthcare monitoring, and target detection and localization in automotive scenarios. Some trends emerging in the current research activities about the application of DL techniques to colocated MIMO radars are illustrated in Section VII. Various ML and DL methods are compared, in terms of accuracy and computational effort, in Section VIII, where their use in human activity classification, and in the detection and position estimation of a moving target is illustrated. Finally, some conclusions are offered in Section IX. Multiple acronyms are employed in our work; their meaning is illustrated in Table I.

II. Colocated MIMO Radars: Basic Principles, History, Architecture and Deterministic Algorithms

This section provides an introduction to the world of colocated MIMO radar systems. After illustrating some basic information about their characteristics and outlining their evolution in the last two decades, the architecture of a colocated MIMO radar system is described. Finally, the received signal model is briefly analyses, and essential information about various detection and estimation algorithms that can be employed in colocated radar systems is provided.

A. Basic principles and classification

The initial excitement about the use of antenna arrays at both transmit and receive sides (i.e., briefly, about MIMO) in wireless systems has been sparked by the pioneering work of J. H. Winters [16], G. J. Foschini [17], Foschini and M. J. Gans [18], and E. Telatar [19]; these researchers predicted huge capacity gains in *wireless communications* affected by multipath fading [20]. A few years later, the exploitation of antenna arrays has been also investigated in the radar field for the potential improvements it could provide in terms of *signal-to-noise ratio* (SNR), resolution and detection capability. In fact, in principle, the availability of multiple transmit/receive antennas allows to (e.g., see [21]–[23])

1) increase the SNR characterizing target echoes and make it more stable;

2) implement *spatial filtering* (i.e., *beamforming*) for directional signal transmission/reception and, consequently, achieve a large *field of view* (FOV);

3) increase the overall number of degrees of freedom and, consequently, the maximum number of targets that can be detected at a given range;

4) improve the *angular resolution* with respect to traditional radar systems;

5) exploit *spatial diversity*, so that uncorrelated aspects of a given target can be perceived.

Generally speaking, MIMO radar systems can be divided in *statistical* MIMO radars [24], [25] and *colocated* MIMO radars [26], [23] on the basis of the distance between their transmit and receive arrays. In fact, the

ABF	Analog Beamforming	LNA	Low Noise Amplifier
ADC	Analog to Digital Converter	LPC	Linear Predictive Coding
AE	Auto-Encoder	LRR	Long Range Radar
AI	Artificial Intelligence	LSTM	Long Short Term Memory
AWGN	Additive White Gaussian Noise	MIMO	Multiple Input Multiple Output
Bi	Bipolar	ML	Machine Learning
BN	Batch Normalization	MLP	Multi-Layer Perceptron
BPTT	Back-Propagation Through Time	MMIC	Monolithic Microwave Integrated Circuit
CAE	Convolutional Auto-Encoder	MRR	Medium Range Radar
CFAR	Constant False Alarm Rate	MUSIC	Multiple Signal Classification
CMOS	Complementary Metal Oxide Semiconductor	MUSIC	Multiple Signal Classification
CNN	Convolutional Neural Network	NN	Neural Network
CS	Compressed Sensing	OFDM	Orthogonal Frequency Division Multiplexing
CVD	Cadence Velocity Diagram	PCA	Principal Component Analysis
DBF	Digital Beamforming	PMCW	Phase Modulated Continuous Wave
DBSCAN	Density Based Spatial Clustering of Applications with Noise	R	Region
DCNN	Deep Convolutional Neural Network	RADAR	Radio Detection And Ranging
DCT	Discrete Cosine Transform	RCS	Radar Cross Section
DFT	Discrete Fourier Transform	RF	Radio Frequency
DL	Deep Learning	RMSE	Root Mean Square Error
DOA	Direction of Arrival	RNN	Recursive Neural Network
ERM	Empirical Risk Minimization	ROI	Region of Interest
ESPRIT	Estimation Signal Parameters Rotational Invariance Technique	RX	Receive
FC	Fully Connected	SAMME	Stagewise Additive Modeling Multi-class Exponential
FCN	Fully Convolutional Network	SFCW	Stepped Frequency Continuous Wave
FDM	Frequency Division Multiplexing	SGD	Stochastic Gradient Descent
FM	Frequency Modulation	SiGe	Silicon-Germanium
FMCW	Frequency Modulated Continuous Wave	SNR	Signal-to-Noise Ratio
FOV	Field of View	SRR	Short Range Radar
FPGA	Field Programmable Gate Array	SVM	Support Vector Machine
FFT	Fast Fourier Transform	TDM	Time Division Multiplexing
GAN	Generative Adversarial Network	TOF	Time Of Flight
GPU	Graphic Processing Unit	TX	Transmit
HCI	Human Computer Interface	ULA	Uniform Linear Array
HGR	Human Gesture Recognition	UWB	Ultra-wideband
HMM	Hidden Markov Model	VCO	Voltage Controlled Oscillator
IAA	Iterative Adaptive Approach	XAI	eXplainable Artificial Intelligence
IOU	Intersection Over Union	YOLO	You Look Only Once
KNN	K - Nearest Neighbour		

Table I: Table of acronyms.

transmit and receive antennas of the radar systems belonging to the first class are widely separated; on the contrary, in radar systems of the second class, transmit antennas are close to the receive ones and, in particular, are usually placed on the same shield. Colocated MIMO radars can be further classified as: a) *mono-static radars*, where transmit and receive arrays share their antenna elements; b) *pseudo-bistatic* radars, where transmit and receive arrays are made of distinct antenna elements, placed at different positions. It is important to keep in mind that, in statistical MIMO radars, spatial diversity originates from the fact that distinct receive antennas, being well separated, can observe uncorrelated parts of the same target. In colocated MIMO radars, instead, a large spatial aperture is achieved by radiating *orthogonal* waveforms. Based on the way these waveforms

are generated, colocated MIMO radars can be divided in: a) *time division multiplexing* (TDM) radars [27], b) *frequency division multiplexing* (FDM) radars [28] and c) *orthogonal frequency division multiplexing* (OFDM) radars [29]. On the one hand, in TDM (FDM) radars, orthogonality is achieved by transmitting through distinct antennas over disjoint time (frequency) intervals; on the other hand, in OFDM radars, any transmit antenna can be used to radiate multiple orthogonal waveforms at the same time. A further classification of colocated MIMO radars, commonly adopted in the automotive field, is based on the maximum measurable range. According to this classification, these systems are divided in (see Table II, where, for each type of radar, the achievable range, the transmission frequency and the typical applications are listed):

1) *Short range radars* (SRRs) - These are able to measure a maximum range of about 30 m and offer the highest angular resolution.
2) *Medium range radars* (MRRs) - These are characterized by a maximum range of about 100 m, offer a quite large azimuthal FOV and achieve a reasonable angular resolution.
3) *Long range radars* (LRRs) - These are characterized by the largest maximum range (250 m) and the thinnest FOV.

In the last paragraph of this section, the architecture of a pseudo-bistatic colocated MIMO radar operating in TDM mode is described in some detail. Our interest in this specific architecture is motivated by its wide use in various civilian applications, and by its capability of detecting multiple targets and accurately estimating their position.

Radar type	Max range (m)	Freq. (GHz)	Typical applications
Short range	30	5-77	Park assist, pre-crash
Mid range	100	24-77	Blind spot detection Rear collision avoidance Cross traffic alert
Long range	250	40-77	Adaptive cruise control

Table II: Classification of colocated MIMO radars on the basis of their maximum measurable distance.

B. A brief history of the colocated MIMO radar technology

The birth of *radio detection and ranging* (briefly, radar) systems dates back to 1904, when the German inventor *Christian Hulsmayer* built a simple ship detection device for avoiding collisions in fog [30]. However, the first practical radar system was developed by the British physicist Sir *Robert Watson-Watt* in 1935, and was employed by the British army in World War II to detect air and sea aggressors [31]. Another fundamental step in the evolution of radar technology is represented by the early studies on *optimal filtering*; the rigorous formulation of this problem and its solution are due to the American scholar *Norbert Wiener* and date back to the 40' [32]. Since then, many advancements have been made in military and civilian radar systems, thanks to the development of signal processing techniques and to the evolution in electronic technology. The most significant advances in signal processing methods applicable to radar systems equipped with antenna arrays have involved both the transmit side and the receive side, and can be summarised as follows.

As far as the transmit side is concerned, substantial research efforts have been devoted to the study of *analog beamforming* (ABF) and *digital beamforming* (DBF) methods for controlling *phased arrays*; both types of methods allow to obtain electronic *beam steering*, i.e. to *steer* the main lobe of the array radiation pattern without any movement of the antennas forming it. It is worth stressing that phased arrays have been around for more than fifty years [33], and that a radar equipped with a phased array is much simpler than a MIMO radar. In fact, a radar system endowed with a phased array generates a single waveform feeding each transmit antenna with a different phase (or, equivalently, with a different delay); consequently, the waveforms radiated by distinct antennas are highly correlated. Moreover, analog beamforming represents the earliest method for electronic beam steering; in this case, each of the signal feeding a transmit antenna is first amplified and then delayed through a phase shifter in a *radio frequency* (RF) stage; an important drawback of this method is represented by the fact that the shape of the resulting beam is fixed. On the other hand, DBF is based on the idea of implementing beam steering in the (digital) baseband portion of the radar hardware by multiplying each signal by a complex gain [34]. This procedure allows to digitally customize the radiated beam, adapting its direction to channel conditions. This technique, also known as *adaptive beamforming* [35], plays an important role in the presence of severe path loss. However, it should be always kept in mind that any radar transmitter exploiting beamforming requires some time (in practice, multiple dwells) to scan the area of interest. On the contrary, if a MIMO radar is employed, the entire observed area is illuminated in a single dwell and beamforming is obtained through the use of different orthogonal waveforms [2].

Another important research area concerning the transmit side of radar systems equipped with antenna arrays concerns the design of the radiated waveforms [23]. Despite the fact that significant theoretical results have been achieved in the field of optimal design of waveforms (e.g., see [36]), few modulation techniques have been employed in commercial MIMO radars until now. These include the *frequency modulated continuous wave* (FMCW) technique [37] (also known as *chirp signal modulation*) and the *stepped frequency continuous wave* (SFCW) technique [38]. In the last years, considerable attention has been also paid to the use of the OFDM technique [39], [40] and to the *phase modulated continuous wave* (PMCW) technique [41].

Early research work regarding the receive side of radar systems endowed with antenna arrays has focused on the development of beamforming methods [42]. One of the most important contributions to this area is represented by the so called *Capon beamformer*, which can provide good resolution and interference rejection capability [43], [44]. Other fundamental contributions about the processing of multiple signals acquired by a radar systems through its antenna array concern the estimation of the *direction of arrival* (DOA) of the electromagnetic waves impinging on the array itself. Here, we limit ourselves to mention the MUSIC [3] and the *estimation of signal parameters via rotational invariance* (ESPRIT) techniques [45], [46].

The development of signal processing methods for MIMO radars started after the end of 2003; in fact, in that period, the concept of MIMO radar, defined as a device able to probe a wireless channel by transmitting multiple signals and receiving their echoes with similar multiplicity, was proposed for the first time [22]. Since the beginning, it was clear that MIMO technology could have represented an important tool to improve the SNR of received signals and to increase radar aperture [2], [21], [23], [47]. Since then, the exploitation of known DOA estimation strategies, developed in the previous years for antenna arrays (like MUSIC and ESPRIT), has been widely investigated for this new type of radars (e.g., see [48] and [49]). However, the availability of MIMO radars able to radiate wideband signals by a large number of antennas and to acquire their echoes through an even larger number of antennas have raised various problems, whose solution requires substantial research efforts. In fact, on the one hand, these devices allow to acquire a rich set of information about the surrounding propagation environment; on the other hand, they require storing and processing large datasets. This has motivated the investigation of *compressed sensing* (CS) and statistical sparsity-based techniques, since these can be exploited to perform signal detection and parameter estimation on the basis of a much smaller dataset than that available in the case in which the received waveforms undergo Nyquist sampling [50], [51]; various examples of CS-based estimation algorithms can be found in ref. [52].

As far as the advancement in electronic technology is concerned, in the remaining part of this paragraph we focus on some important results achieved in the development of compact integrated radar devices employed in the automotive field, since this is one of the first commercial markets in which MIMO radars have been playing a fundamental role. The first generation of commercial *ultra-wideband* (UWB) automotive radar sensors operating in the 77 GHz band has become available in 1999. These devices were not endowed with antenna arrays and their implementation was based on discrete electronic components (in particular, gallium-arsenide Gunn diodes mounted inside a waveguide cavity were employed in the generation of RF waveforms). However, electronic technology progressed quickly in this field and, after few years, MMICs employing high-performance *silicon-germanium* (SiGe) transistors became available for the implementation of fully integrated radars. Pioneering work in the development and manufacturing of such a technology has been accomplished by the *Infineon* company, that has started its production in 2004 [53]. It is also worth mentioning that, in the same year, a description of the first fully integrated 24-GHz eight-element phased array receiver in SiGe and of the first fully integrated 24-GHz four-element phased array transmitter with integrated power amplifiers in *complementary metal-oxide semiconductor* (CMOS) has appeared [54]; these devices were able to accomplish beamforming and could be used for communication, ranging, positioning, and sensing applications. Other examples of phased arrays operating in X and Ku-band have been described later in ref. [55]. The first FMCW MIMO radar transceiver operating at 77 GHz has been implemented in SiGe technology in 2008 [56], whereas the production of the first MIMO FMCW radar, operating according to a TDM strategy and equipped with an array of colocated antennas, started in 2009 [57], [58]. As far as we know, the last device represents the first compact MIMO radar system based on a MMIC in SiGe, operating at 77 GHz and radiating ultra-wideband signals. In this system, wide-band and high-frequency patch antennas are built on a RF substrate [59], while the base-band MIMO signal processing is accomplished off-chip by a *field programmable gate array* (FPGA) board. Moreover, the *analog-to-digital converters* (ADCs) at the receive side are implemented in CMOS technology and embedded in the transceiver chip; this has been made possible by the SiGe Bi-CMOS process, which has allowed to integrate multiple functions on a single chip and at low cost. In the last decade, radar designers working on the development of new integrated radar devices have investigated the use of the more scalable CMOS RF technology [60]. An important trend in the technological evolution of MIMO radar systems is also represented by the attempt of exploiting the same hardware for both radar and communications [61]. Some milestones achieved in the evolution of the signal processing methods and of the technology employed in colocated radar

systems during the last two decades are summarized in Fig. 1.

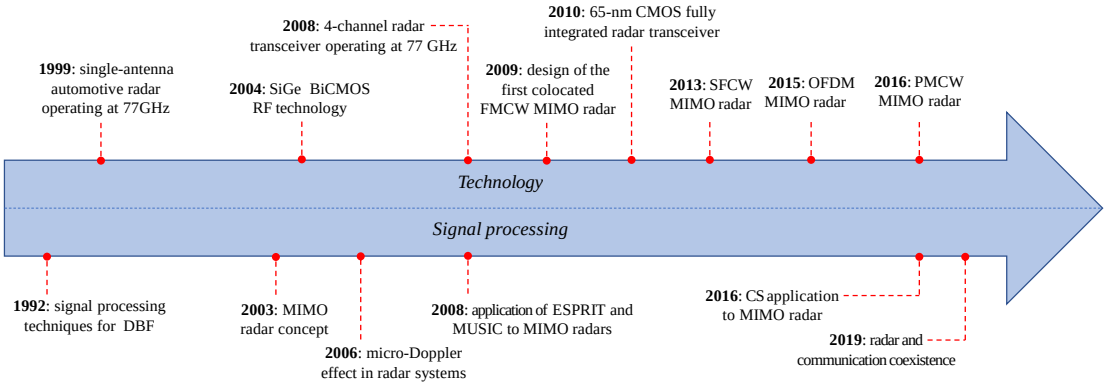


Figure 1: Milestones in the evolution of colocated radars.

C. Architecture of a colocated TDM MIMO radar

In the remaining part of this manuscript, we always refer to a colocated and bistatic MIMO radar system; its architecture is illustrated in Fig. 2. Moreover, we first assume that: a) the considered radar system is equipped with a *two-dimensional* (2D) array, consisting of N_T *transmit* (TX) and N_R *receive* (RX) antennas; b) it employs a TDM strategy; c) it exploits all the available transmit diversity (i.e., all the available TX antennas). Consequently, if a time slot of T_0 s is assigned to each TX antenna, transmission from all the TX antennas is accomplished over an interval lasting $T_F \triangleq N_T T_0$ s; this interval represents the duration of a single *transmission frame*.

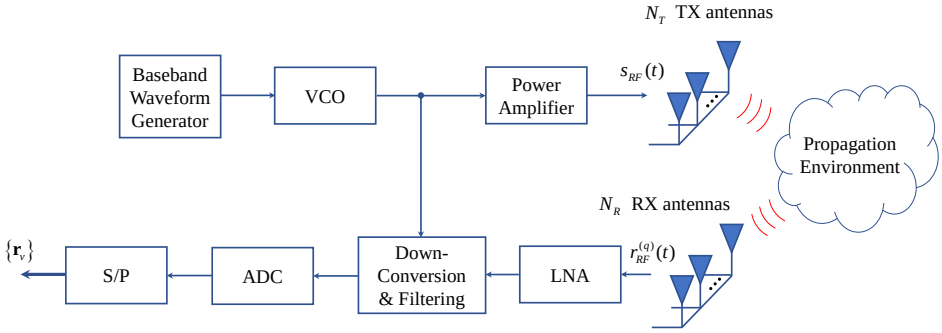


Figure 2: MIMO radar transmitter (upper part) and receiver (lower part).

In this manuscript, two different models are considered for the RF signal generated by the *voltage controlled oscillator* (VCO) of the radar transmitter and radiated by its transmit array after power amplification. In the first case, corresponding to a FMCW radar system, the VCO is fed by a *periodic ramp generator*; this produces a chirp FM signal, whose instantaneous frequency evolves periodically, as illustrated in Fig. 3. In this figure, the parameters T, T_R and T_0 represent the *chirp interval*, the *reset time* and the *pulse period* (or *pulse repetition interval*), respectively [9], whereas the parameters f_0 and B are the *start frequency* and the *bandwidth*, respectively, of the transmitted signal. For this reason, if we focus on the time interval $(0, T)$ and assume that, in that interval, the p-th TX antenna is employed by the considered radar system (with $p \in \{0\ ,\ 1,\ ...,\ N_T - 1\}$), the radiated signal can be expressed as

$$s_{RF}(t) = A_{RF} \Re\{s(t)\}, \tag{1}$$

where A_{RF} is its amplitude,

$$s(t) \triangleq \exp[j\theta(t)], \tag{2}$$

$$\theta(t) \triangleq 2\pi \left(f_0 t + \frac{\mu}{2} t^2 \right) \tag{3}$$

and

$$\mu = \frac{B}{T} \tag{4}$$

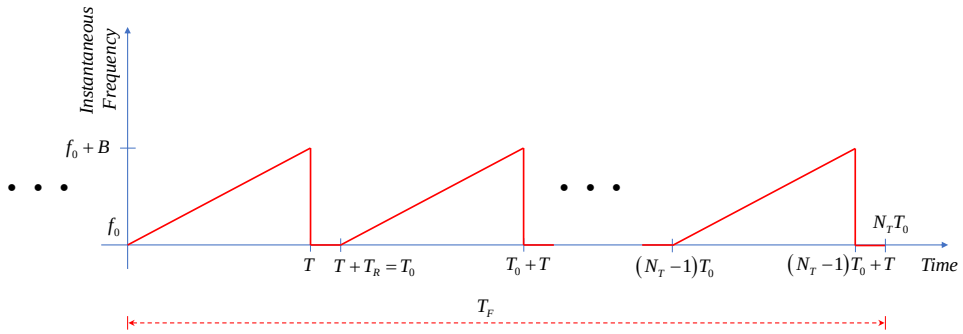


Figure 3: Representation of the instantaneous frequency of the RF signal generated by the VCO in a FMCW radar system.

is the *chirp rate*, i.e. the steepness of the generated frequency chirp.

Let $r_{RF}^{(q)}(t)$ denote the signal available at the output of the q-th receive antenna, with $q = 0, 1, ..., N_R - 1$ (see Fig. 2); this signal feeds a *low noise amplifier* (LNA), whose output undergoes downconversion, filtering and analog-to-digital conversion at a frequency $f_s = 1/T_s$, where T_s denotes the sampling period of the employed ADC. If we assume that the radiated signal $s_{RF}(t)$ (1) is reflected by L *static point targets*, the useful component of $r_{RF}^{(q)}(t)$ consists of the superposition of L echoes, each originating from a distinct target. In this case, if the propagation environment undergoes slow variations, a simple mathematical model can be developed to represent the sequence of samples generated by the ADC in a single chirp interval. In deriving this model, the couple of the involved *physical* TX and RX antennas (namely, the p-th TX antenna and the q-th RX antenna) of the considered *bistatic radar* is often replaced by a single *virtual antenna* of an equivalent *monostatic radar*. In particular, the abscissa x_v and the ordinate y_v of the v-th *virtual antenna element* associated with the p-th TX antenna and the q-th RX antenna are computed as[1]

$$x_v = \frac{x_p + x_q}{2} \tag{5}$$

and

$$y_v = \frac{y_p + y_q}{2}, \tag{6}$$

respectively, with $v = 0, 1, ..., N_V - 1$; here, (x_p, y_p) $((x_q, y_q))$ are the coordinates[2] of the TX (RX antenna) and $N_V \triangleq N_T \cdot N_R$ represents the overall size of the resulting *virtual array*. Based on these assumptions, the n-th received signal sample acquired through the v-th virtual antenna element (with $v = 0, 1, ..., N_V - 1$) can be expressed (e.g., see [62, Par. 4.6, eq. (4.27)])

$$r_{v,n} = \sum_{l=0}^{L-1} a_l \cos(2\pi n F_{v,l} + \psi_{v,l}) + w_{v,n}, \tag{7}$$

with $n = 0, 1, ..., N - 1$; here, N is the overall number of samples acquired over a chirp period, a_l is the amplitude of the l-th component of the useful signal (this amplitude depends on both the range R_l and the reflectivity of the l-th target, but is assumed to be independent of v for simplicity),

$$F_{v,l} \triangleq f_{v,l} T_s \tag{8}$$

is the normalized version of the frequency

$$f_{v,l} \triangleq \mu \tau_{v,l}\,, \tag{9}$$

characterizing the l-target detected on the v-th virtual receive antenna,

$$\tau_{v,l} = \frac{2}{c}\left[R_l + x_v \cos(\theta_l)\sin(\phi_l) + y_v \sin(\theta_l)\right] \tag{10}$$

is the delay of the echo generated by the l-th target and observed on the v-th virtual channel, R_l, ϕ_l and θ_l denote the range of the l-th target, its azimuth and its elevation, respectively,

$$\psi_{v,l} \cong 2\pi f_0 \tau_{v,l}, \tag{11}$$

[1]This is not the only rule adopted in the technical literature to compute the coordinates of the v-th virtual antenna element. For instance, in ref. [2, Par. 4.3.1, pp. 159-161], the abscissa (ordinate) of this element is evaluated as $2x_v$ ($2y_v$), where x_v and y_v are expressed by eqs. (5) and (6), respectively. Keep in mind, however, that, if the last rule is adopted, all the following formulas involving such coordinates must be changed accordingly.

[2]A reference system lying on the physical antenna array is assumed.

and $w_{v,n}$ is the n-th sample of the *additive white Gaussian noise* (AWGN) sequence affecting the received signal (the noise variance is denoted σ_w^2 in the following and is assumed to be independent of v).

The second case we consider for the generation of the radiated waveform corresponds to a SFCW radar system. Its name is motivated by the fact that the VCO of its transmitter is fed by a *staircase generator*. For this reason, the instantaneous frequency of the resulting RF signal takes on N distinct and uniformly spaced values in an interval lasting T s for each TX antenna; the n-th value of the instantaneous frequency is

$$f_n = f_0 + n\,\Delta f, \tag{12}$$

with $n = 0, 1, ..., N-1$, where f_0 is the minimum radiated frequency, Δf is the *frequency step size* and N is the overall number of transmitted frequencies. It is not difficult to prove that, under the same assumptions made in the derivation of eq. (7), the measurement acquired through the v-th virtual element at the n-th frequency can be expressed as

$$r_{v,n} = \sum_{l=0}^{L-1} a_l \exp\left[-j\left(2\pi n F_{v,l} + \psi_{v,l}\right)\right] + w_{v,n}, \tag{13}$$

with $v = 0, 1, ..., N_V - 1$; here, the phase $\psi_{v,l}$ is still expressed by eq. (11),

$$F_{v,l} \triangleq \Delta f\,\tau_{v,l} \tag{14}$$

is the normalised frequency characterizing the l-th target and observed on the v-th virtual antenna, and the parameters a_l, $\tau_{v,l}$ and the random variable $w_{v,n}$ have exactly the same meaning[3] as the one illustrated for the received signal model (7). It is important to point out that, similarly as the baseband signal model (7) developed for a FMCW radar, the model (13) allows to interpret the signal observed on the v-th channel as a superposition of L oscillations. However, the former model is *real*, whereas the latter one is *complex*. Moreover, in both cases, the samples $\{r_{v,n};\, n = 0, 1, ..., N-1\}$ can be collected in the N-dimensional vector

$$\mathbf{r}_v \triangleq \left[r_{v,0}, r_{v,1}, ..., r_{v,N-1}\right]^T, \tag{15}$$

which is processed by the next stages of the radar receiver for target detection and estimation. As it can be easily inferred from eq. (7) (eq. (13)), in a FMCW (SFCW) radar system, the problem of target detection and range estimation on the v-th virtual channel is equivalent to the classic problem of estimating the *frequencies* of multiple overlapped sinusoids (multiple overlapped complex exponentials) in the presence of AWGN [63]. In fact, if an estimate $\hat{f}_{v,l}$ of the frequency $f_{v,l}$ (9) and an estimate $\hat{F}_{v,l}$ of the normalised frequency $F_{v,l}$ (14) are available for the v-th virtual channel, an estimate of the range R_l can be computed as (see eqs. (9) and (10))

$$\hat{R}_{v,l} = \frac{1}{2}\frac{\hat{f}_{v,l}}{\mu}c \tag{16}$$

and as (see eqs. (10) and (14))

$$\hat{R}_{v,l} = \frac{1}{2}\frac{\hat{F}_{v,l}}{\Delta f}c \tag{17}$$

respectively, for any v and l. Information about the angular coordinates (namely, the azimuth and the elevation) of the l-th target, instead, can be acquired through the estimation of the set of N_V phases $\{\psi_{v,l};\, v = 0, 1, ..., N_V - 1\}$ observed over the available virtual antennas. In fact, since (see eqs. (10) and (11))

$$\psi_{v,l} \cong 4\pi\frac{f_0}{c}\left[R_l + x_v\cos\left(\theta_l\right)\sin\left(\phi_l\right) + y_v\sin\left(\theta_l\right)\right] \tag{18}$$

where

$$\lambda \triangleq \frac{c}{f_0} \tag{19}$$

is the wavelength associated with the frequency f_0, the sequence $\{\psi_{v,l};\, v = 0, 1, ..., N_V - 1\}$ exhibits a periodic behavior characterized by the *normalised horizontal spatial frequency*

$$F_{H,l} \triangleq 2\frac{d_H}{\lambda}\cos\left(\theta_l\right)\sin\left(\phi_l\right), \tag{20}$$

if the considered virtual elements form an horizontal *uniform linear array* (ULA), whose adjacent elements are spaced d_H m apart. Dually, if a vertical ULA is assumed, the periodic variations observed in the same sequence of phases are characterized by the *normalised vertical spatial frequency*

$$F_{V,l} \triangleq 2\frac{d_V}{\lambda}\sin\left(\theta_l\right), \tag{21}$$

[3]Note, however, that $w_{v,n}$ is a *complex* Gaussian random variable; its variance is also denoted σ_w^2 in the following.

where d_V denotes the distance between adjacent elements of the vertical virtual array. Consequently, angle finding can be easily accomplished by DBF, i.e. by performing FFT processing on the estimated phases taken across multiple elements of the virtual array in a single frame interval [4], [64]. Note, however, that other angle estimation methods, achieving a better resolution than FFT processing are also available; here, we limit to mention the so called subspace-based methods (such as MUSIC and ESPRIT), sparse sensing-based methods [65], [66] and the *iterative adaptive approach* (IAA) developed in ref. [67]. Subspace-based methods require computing an accurate estimate of the array covariance matrix; consequently, the measurements acquired over multiple snapshots must be processed. Moreover, they do not allow to estimate the amplitude of the echo associated with each detected target and require prior knowledge of the size of the useful signal subspace (i.e., of the number of detectable targets). On the contrary, sparse sensing-based methods and IAA can generate angle estimates on the basis of a single snapshot of the received signal; however, this result is obtained at the price of a significant computational effort.

Both the received signal models (7) and (13) hold if all the observed targets are static. Let us focus now on a FMCW radar system operating in the presence of L moving point targets and having the following characteristics: a) it is equipped with a single TX antenna and a single RX antenna (i.e., $N_T = N_R = 1$); b) its reset time T_R is equal to 0, so that $T_0 = T$ (see Fig. 3); c) its transmission frame consists of N_c chirps, so that the duration T_F of the transmission frame is equal to $N_c T_0 = N_c T$ s; d) N distinct ADC samples are acquired in each chirp interval at the receive side. Then, it is not difficult to prove that, if the ranges of all the targets are much larger than their displacements observed during the considered transmission frame, the n-th sample of the signal acquired in the k-th chirp interval (with $k = 0, 1, ..., N_c - 1$) can be expressed in a similar way as eq. (7), namely as (e.g., see [9, eq.(5)])

$$r_n^{(k)} \cong \sum_{l=0}^{L-1} a_l \cos(2\pi n \ (F_l + F_{D,l}) \ + \psi_l^{(k)}) + w_n^{(k)}, \tag{22}$$

where $F_l = \mu \, \tau_l \, T_s$ (see eqs. (8) and (9)), $\tau_l = 2R_l/c$ is the delay of the echo generated by the l-th target and observed in the first chirp interval (in this interval, the target range is assumed to be equal to R_l),

$$F_{D,l} = \frac{2 \, v_l}{\lambda} T_s \tag{23}$$

is the normalised Doppler frequency, v_l is the radial velocity[4] of the l-th target,

$$\psi_l^{(k)} \cong \frac{4\pi}{\lambda} R_l^{(k)}, \tag{24}$$

$$R_l^{(k)} = R_l + v_l \, kT \tag{25}$$

is the target range observed in the k-th chirp interval and $w_n^{(k)}$ is the n-th sample of the AWGN sequence affecting the received signal in the same chirp interval.

Let us focus next on a SFCW radar system operating in the same scenario as the one just described for a FMCW radar system and having the following characteristics: a) it is equipped with a single TX antenna and a single RX antenna; b) its reset time T_R is equal to 0, so that $T_0 = T$; c) its transmission frame consists of N_c frequency sweeps; d) in each sweep (lasting T s), N distinct and uniformly spaced frequencies are generated according to eq. (15). Then, it can be shown that, if $f_0 >> N\Delta f$, the measurement acquired at the n-th frequency in the k-th frequency sweep (with $k = 0, 1, ..., N_c - 1$) can be expressed in a similar way as eq. (13) and, in particular, as

$$r_n^{(k)} \cong \sum_{l=0}^{L-1} a_l \exp\left[-j\left(2\pi\left(nF_l + \bar{F}_{D,l}\right) + \psi_l^{(k)}\right)\right] + w_{k,n} \ , \tag{26}$$

where $F_l \triangleq \Delta f \, \tau_l$ (see eq. (14)), τ_l is the delay of the echo generated by the l-th target and observed in the first frequency sweep (when the target range is equal to R_l),

$$\bar{F}_{D,l} = \frac{2 \, v_l}{\lambda} \tau_l \tag{27}$$

is the normalised Doppler frequency, and the parameters λ and $\psi_l^{(k)}$ are still expressed by eqs. (19) and (24), respectively.

[4]This speed is positive (negative) if the target is approaching (is moving away) from the radar.

In both the considered radar systems, the rate of change observed in the sequence of phases $\{\psi_l^{(k)};\ k = 0,\ 1,\ ...,\ N_c - 1\}$ is proportional to v_l, since (see eqs. (24) and (25))

$$\psi_l^{(k+1)} - \psi_l^{(k)} \cong \frac{4\pi}{\lambda}\left(R_l^{(k+1)} - R_l^{(k)}\right) = 4\pi\frac{T}{\lambda}v_l \tag{28}$$

with $k = 0,\ 1,\ ...,\ N_c - 1$. Therefore, target velocity can be easily assessed by means of FFT processing after computing an estimate of the above mentioned phases.

In the technical literature, range and speed information of the moving targets detected in a given propagation environment are usually condensed in a 2D plot, called *range-Doppler map* [5], [9]. In a FMCW radar system equipped with a single TX antenna and a single RX antenna, this map is generated as follows. Let $\mathbf{r}^{(k)}$ denote the N-dimensional (column) vector consisting of the real measurements acquired in the k-th chirp of a transmission frame, with $k = 0,\ 1,\ ...,\ N_c - 1$, where N_c is the overall number of chirps forming the frame itself. The N_c vectors $\{\mathbf{r}^{(k)};\ k = 0,\ 1,\ ...,\ N_c - 1\}$ are collected in the matrix

$$\mathbf{R} = [\mathbf{r}^{(0)}\ \mathbf{r}^{(1)}\ \cdots\ \mathbf{r}^{(N_c-1)}]\,, \tag{29}$$

having size $N \times N_c$. This matrix undergoes zero-padding, that turns it into a matrix $\mathbf{R}_{\rm ZP}$ of size $N_0 \times N_0^{'}$. The last matrix feeds a $N_0 \times N_0^{'}$-th order FFT, that generates the *range-Doppler* (complex) *matrix*

$$\mathbf{D} = [d_{p,q}] \triangleq \mathrm{FFT}_{N_0 \times N_0^{'}}[\mathbf{R}]\,, \tag{30}$$

where $\mathrm{FFT}_{X\times Y}[\cdot]$ denotes 2D FFT operator of size $X \times Y$; note that the index p (q) labelling the elements of the matrix $\mathbf{D}$ refers to the range (Doppler) domain. Representing, on a Cartesian plane, the absolute value of the elements of the matrix $\mathbf{D}$ yields the above mentioned range-Doppler map.

In the last fifteen years, substantial attention has been also paid to the problem of estimating the *micro-movements* of detected targets; such movements usually originate from mechanical vibrations or rotations (overlapping to a bulk translation) and may generate a frequency modulation in the received signal; the last phenomenon is known as *micro-Doppler*. The recent interest in micro-Doppler is motivated by the fact that it can be exploited to establish the dynamic properties of targets [68] and, consequently, can be used to *classify* them or identify specific properties related to their motion. In a FMCW radar system equipped with a single TX antenna and a single RX antenna, the micro-Doppler phenomenon can be analysed as follows. Let us assume that N_f consecutive frames (each consisting of N_c chirps) are transmitted by the considered radar system and that the range-Doppler matrix $\mathbf{D}$ (30) is evaluated for each frame (the matrix referring to the m-th frame is denoted $\mathbf{D}_m = [d_{p,q}^{(m)}]$, with $m = 0,\ 1,\ ...,\ N_f - 1$). Relevant information about the micro-Doppler fluctuations, also known as the micro-Doppler *signatures*, characterizing a certain range interval can be acquired through the real matrix $\mathbf{E} = [E_{m,q}]$, having size $N_f \times N_0^{'}$ and whose element on its m-th row and q-th column is evaluated as

$$E_{m,q} \triangleq \sum_{p=p_{\min}}^{p_{\max}} |d_{p,q}^{(m)}|^2 \tag{31}$$

with $m = 0,\ 1,\ ...,\ N_f - 1$ and $q = 0,\ 1,\ ...,\ N_0^{'} - 1$; here, $p_{\min}$ ($p_{\max}$) denotes the value of the index p associated with the minimum (maximum) range of interest. Representing the elements of the matrix $\mathbf{E}$ on a Cartesian plane produces the so called *spectrogram* [68], that shows the time evolution of the Doppler phenomenon.

Additional information about the dynamical properties of a moving target can be acquired through another diagram, known as *cadence velocity diagram* (CVD). This diagram allows us to identify the most relevant frequency components associated with a given motion (e.g., if a walking pedestrian is considered, the speed of his arms can be extracted from the associated CVD). Moreover, its generation is based on the complex matrix $\mathbf{G} = [G_{l,q}]$, having size $N_f^{'} \times N_0^{'}$ and computed as the $N_f^{'} \times N_0^{'}$-th order FFT of the matrix $\mathbf{E}_{ZP} = [E_{m,q}^{(ZP)}]$, that results from zero padding of the matrix $\mathbf{E}$ defined above; therefore, we have that

$$G_{l,q} \triangleq \frac{1}{N_f}\sum_{m=0}^{N_f^{'}-1} E_{m,q}^{(ZP)} \exp\left(-j2\pi\frac{m}{N_f^{'}}\bar{f}_l\, T_F\right) \tag{32}$$

with $l = 0,\ 1,\ ...,\ N_f^{'} - 1$ and $q = 0,\ 1,\ ...,\ N_0^{'} - 1$; here, T_F is the duration of a single transmission frame, $E_{m,q}^{(ZP)} = E_{m,q}$ for $m = 0,\ 1,\ ...,\ N_f - 1$ and $E_{m,q}^{(ZP)} = 0$ for $m > N_f - 1$, and

$$\bar{f}_l \triangleq \frac{l}{T_F} \tag{33}$$

is the l-th *cadence frequency*. The CVD results from representing, on a Cartesian plane, the absolute value of the elements of the matrix $\mathbf{G}$.

III. Machine Learning Techniques for Colocated MIMO Radar Systems

In this section, after illustrating the main differences between a deterministic approach and a ML-based approach to target detection and estimation, the most important ML techniques applied in the field of colocated MIMO radar systems are described. Our introduction to these techniques is based on a specific case study, involving a FMCW radar system (see Paragraph II-C).

A. A case study

The most relevant conceptual differences between a deterministic approach and a ML approach to detection and estimation problems in MIMO radar systems can be understood by analysing the detection of a single point target, and the estimation of its range R and its azimuth ϕ in a 2D propagation scenario. In this case, we assume that an FMCW radar system equipped with an ULA, consisting of three antenna elements, is employed (see Fig. 4-a). This array is made of a central TX antenna and a couple of antipodal RX antennas (these are

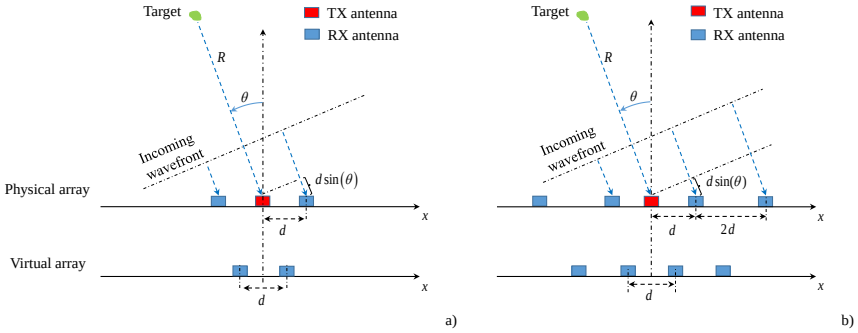


Figure 4: Physical geometry and virtual array of a colocated FMCW MIMO radar equipped with an ULA composed by a single TX antenna and: a) two RX elements; b) four RX elements.

identified by a red box and two blue boxes, respectively, in the considered figure), so that $N_T = 1$ and $N_R = 2$; consequently, a *virtual* array, consisting of $N_V = 2 \cdot 1 = 2$ virtual elements, is available. The abscissa x_v of the v-th *virtual antenna element* associated with the TX antenna and the v-th RX antenna is computed as (see eq. (5))

$$x_v = \frac{x_t + x_{r,v}}{2} \tag{34}$$

with $v = 0$ and 1; here, $x_t = 0$ and $x_{r,0} = -d$ ($x_{r,1} = d$) are the abscissas of the TX and of the first (second) RX antenna, respectively (note that the origin of our reference system coincides with the center of the array). If the target is in far field[5], the wavefront of the electromagnetic echo originating from it is a straight line and is orthogonal to the line connecting the target with the center of the array. In these conditions, the n-th time-domain sample acquired on the v-th virtual antenna in a single snapshot can be expressed as (see eq. (7))

$$\begin{aligned} r_{v,n} &= a_v \cos\left(2\pi n f_v T_s + \psi_v\right) + w_{v,n}, \qquad (35) \\ &= A_v \exp\left(j\, 2\pi n F_v\right) + A_v^* \exp\left(-j\, 2\pi n F_v\right) + w_{v,n}, \qquad (36) \end{aligned}$$

for $n = 0, 1, ..., N-1$, where (see eqs. (8), (9) and (10))

$$F_v \triangleq f_v T_s, \tag{37}$$

$$f_v = \frac{2\mu}{c}\left[R + x_v \sin\left(\phi\right)\right], \tag{38}$$

$$x_v = (-1)^{(v+1)}\frac{d}{2}, \tag{39}$$

d is inter-antenna spacing of the considered ULA,

$$A_v \triangleq \frac{1}{2} a_v \exp\left(j\psi_v\right) \tag{40}$$

is a complex parameter depending on the target reflectivity a_v and (see eqs. (10) and (11))

$$\psi_v \triangleq \angle A_v \cong \frac{4\pi}{\lambda}\left[R + x_v \sin\left(\phi\right)\right] \tag{41}$$

[5]A rigorous definition of this condition can be found in ref. [69, Par. 2.2.4, pp. 34-36]

is the phase observed on the considered antenna (the wavelength λ is defined by eq. (19)). It is important to point out that:

a) Relevant information about the target *azimuth* are provided by the frequency difference

$$\Delta f_{0,1} \triangleq f_1 - f_0 \tag{42}$$

or by the phase variation

$$\Delta\psi_{0,1} \triangleq \angle A_1\, A_0^*, \tag{43}$$

where the quantity $\angle X$ represents the phase of the complex number X (it belongs to interval $[-\pi, \pi)$). In fact, on the one hand, from eqs. (38)-(39) it is easily inferred that (see the definition (42))

$$\Delta f_{0,1} = 2\frac{\mu d}{c}\sin(\phi); \tag{44}$$

on the other hand, based on eqs. (39)-(41), it is easy to show that (see the definition (43))

$$\Delta\psi_{0,1} = \psi_1 - \psi_0 = 4\pi\frac{d}{\lambda}\sin(\phi), \tag{45}$$

provided that the inequality

$$4\pi\frac{d}{\lambda}\left|\sin(\phi)\right| \leq \pi \tag{46}$$

holds for any ϕ. The last condition is met for any $\phi \in [-\frac{\pi}{2}, \frac{\pi}{2})$ if

$$d \leq \lambda/4. \tag{47}$$

b) If the received signal is noiseless, the frequency f_v is known and N is large, the complex amplitude A_v can be easily estimated as[6]

$$\hat{A}_v \cong \frac{1}{N}\bar{X}_v(f_v), \tag{48}$$

where

$$\bar{X}_v(f) \triangleq \sum_{n=0}^{N-1} r_{v,n}\exp(-j2\pi n f T_s) \tag{49}$$

represents the Fourier transform of the sequence $\{r_{v,n};\ n = 0,\ 1,\ ...,\ N-1\}$.

c) Information about the target *range* is provided by the *average frequency* (see eq. (38))

$$f_m \triangleq \frac{f_0 + f_1}{2} = \frac{2\mu}{c}R \tag{50}$$

Therefore, the estimation of the frequency of the sinusoid contained in the noisy data sequence acquired through each virtual antenna represents a fundamental problem in target detection and estimation. It is well known that the so called *periodogram method* can be employed to solve it in an approximate way [7], [70]. This method is based on the computation of the amplitude spectrum of the zero-padded measurement sequence and on the identification of its peak.

Based on the mathematical results and the considerations illustrated above, a simple deterministic algorithm, consisting of the three steps listed below, can be easily derived for the detection of the target and the estimation of its spatial coordinates $(R,\ \phi)$.

1. DFT *processing* - In this step, the N-dimensional vector

$$\mathbf{r}_v \triangleq \left[r_{v,0}, r_{v,1}, ..., r_{v,N-1}\right]^T, \tag{51}$$

with $v = 0$ and 1, undergoes *zero padding* (ZP); this results in the N_0-dimensional vector

$$\mathbf{r}_v^{(ZP)} \triangleq \left[\mathbf{r}_v^T\ \mathbf{0}_P^T\right]^T \tag{52}$$

where $N_0 \triangleq M_r\, N$, $\mathbf{0}_P$ denotes the P-dimensional (column) null vector and M_r represents the selected *oversampling factor* adopted in *time-domain processing*. Then, the vector $\mathbf{r}_v^{(ZP)}$ (52) feeds a N_0-th order *discrete Fourier transform* (DFT); this produces the N_0-dimensional vector

$$\mathbf{X}_v \triangleq \left[X_{v,0}, X_{v,1}, ..., X_{v,N_0-1}\right]^T, \tag{53}$$

where

$$X_{v,l} = \frac{1}{N_0}\bar{X}_v(\bar{f}_l) \tag{54}$$

[6]This result can be easily proved by substituting eq. (36) in the *right-hand side* (RHS) of the definition (49).

$\bar{X}_v(f)$ is defined by eq. (49) and

$$\bar{f}_l \triangleq \frac{l}{N_0 T_s}, \tag{55}$$

with $l = 0, 1, ..., N_0 - 1$. Finally, the N_0-dimensional vector

$$\mathbf{P} \triangleq [P_0, P_1, ..., P_{N_0-1}]^T, \tag{56}$$

where

$$P_l \triangleq \frac{M_r^2}{2}\left[|X_{0,l}|^2 + |X_{1,l}|^2\right], \tag{57}$$

with $l = 0$, 1, ..., $N_0 - 1$, is computed; note that the quantity P_l (57) represents a sort of *average power spectrum* evaluated at the frequency $\bar{f}_l$ (55).

2. *Target detection* - The problem

$$\hat{l} = \arg \max_{\tilde{l} \in \{0,1,...,N_0/2\}} P_{\tilde{l}} \tag{58}$$

is solved and a target is detected if the condition

$$P_{\hat{l}} > P_{th} \tag{59}$$

is satisfied, where P_{th} is a proper threshold. When this occurs, the next step is executed; otherwise, the algorithm stops.

3. *Estimation of target coordinates* - The estimate

$$\hat{f}_m = \frac{\hat{l}}{N_0 T_s} \tag{60}$$

of the frequency f_m (50) and the estimate

$$\hat{A}_v = M_r \, X_{v,\hat{l}} \tag{61}$$

of the complex amplitude A_v (40) (with $v = 0$ and 1) are computed. Then, the estimate (see eq. (50))

$$\hat{R} = \hat{f}_m \frac{c}{2\mu} \tag{62}$$

of the target range R and the estimate (see eq. (45))

$$\hat{\phi} = \arcsin\left(\frac{\lambda}{4\pi d}\Delta\hat{\psi}_{0,1}\right) \tag{63}$$

of the target azimuth θ are evaluated; here,

$$\Delta\hat{\psi}_{0,1} = \angle X_{1,\hat{l}} \left(X_{0,\hat{l}}\right)^*. \tag{64}$$

represents an estimate of $\Delta\psi_{0,1}$ (45) and its expression is based on eqs. (48), (54) and (61).

This concludes the description of the proposed detection and estimation algorithm. It is important to point out that:

a) The accuracy achievable in range estimation is influenced by the DFT order N_0 and, consequently, for a given N, by the oversampling factor M_r. Increasing the value of the parameter M_r leads to a more refined analysis of the spectrum $\bar{X}_v(f)$ (49) and, consequently, allows to locate the spectral peak originating from the target with better accuracy; however, this result is achieved at the price of an higher computational cost.

b) The estimate $\hat{\phi}$ (63) is *unambiguous* if the condition (47) is satisfied or if, for a given $d > \lambda/4$, the azimuth ϕ belongs to the interval $[-\phi_m, \phi_m)$, where (see eq. (46))

$$\phi_m \triangleq \arcsin\left(\frac{\lambda}{4\,d}\right) \tag{65}$$

c) Eq. (44) has not been exploited to compute an estimate of the target azimuth. This is due to the fact that the quality of this estimate is limited by the accuracy of frequency estimation on each antenna; such an accuracy, in turn, is intrinsically limited by the DFT order N_0.

d) If the target reflectivity observed on the two antennas is approximately the same (i.e., if $a_0 \cong a_1$), an estimate of it can be computed as (see eqs. (40) and (61))

$$\hat{a} \triangleq M_r \left[\left|X_{0,\hat{l}}\right| + \left|X_{1,\hat{l}}\right|\right]. \tag{66}$$

e) The estimation of the azimuth characterizing the echo from a specific target requires at least two RX antennas, since it is based on computation of the phase variation observed at a specific frequency on at least two receive antennas (see eqs. (63) and (64)).

f) The maximum number of detectable targets depends on the number of virtual elements of the whole array. It is worth noting that, unlike a phased array system, where a single waveform is transmitted, a MIMO radar system endowed with N_T different TX antennas can radiate N_T independent signals. This leads to the conclusion that the maximum number of targets that be can uniquely identified by a MIMO radar is N_T times larger than that of its counterpart employing a phased array [23], if the first system employs an ULA whose virtual elements do not overlap (like the ULAs shown in Fig. 4).

The estimation accuracy achieved by the considered radar system can be improved by increasing the size of its ULA, i.e. the overall number of its antennas, so that a larger number of virtual channels becomes available. For instance, if the ULA shown in Fig. 4-a) is replaced by the one represented in Fig. 4-b) (and characterized by $N_T = 1$ and $N_R = 4$), $N_V = 4$ virtual channels become available, i.e. the overall number of virtual antennas is doubled with respect to the previous case. Note that this results not only in an increase of the maximum number of detectable targets, but also in an improvement of the *angular resolution* $\Delta\phi$, defined as the minimum angular separation below which the DOAs of two distinct targets cannot be separated. More specifically, if an ULA is used and the bore-sight direction is considered, we have that (e.g., see [6, Par. 4, eq. (51)])

$$\Delta\phi = \frac{\lambda}{2\,d\,\left(N_V - 1\right)\cos(\phi)}. \tag{67}$$

It is also worth mentioning that the algorithm illustrated above for a couple of virtual channels can be easily extended to the case of an ULA providing N_V virtual channels. The only relevant modification concerns step 3., since the N_V-dimensional vector

$$\hat{\mathbf{A}} = \left[\hat{A}_0, \hat{A}_1, ..., \hat{A}_{N_V-1}\right]^T, \tag{68}$$

where $\hat{A}_v$ is still expressed by eq. (61) for any v, becomes available and, consequently, $(N_V - 1)$ phase variations, referring to the $(N_V - 1)$ distinct couples of adjacent virtual antennas can be evaluated. If we assume that the variations of the target reflectivity over the whole virtual array are negligible and that the SNR on each virtual antenna is high, such variations are approximately constant, being all expressed by the RHS of eq. (45). This means that a *phase modulation*, characterized by the *normalised spatial frequency*

$$F = 2\frac{d}{\lambda}\sin\left(\phi\right), \tag{69}$$

is observed in the sequence $\{\hat{A}_v;\ v = 0,\ 1,\ ...,\ N_V - 1\}$. An estimate of the parameter F can be computed by exploiting, once again, the *periodogram method*. In practice, this requires executing the following three steps:

1. DFT *processing* - The vector $\hat{\mathbf{A}}$ is zero padded by appending to it a null vector of size $(M_A - 1)N_V$, where M_A represents the *oversampling factor* adopted in *spatial processing*; this produces the $\bar{N}_0$-dimensional vector $\hat{\mathbf{A}}_{ZP}$, where $\bar{N}_0 \triangleq M_A N_V$. The vector $\hat{\mathbf{A}}_{ZP}$ feeds a $\bar{N}_0$-th order DFT, generating the $\bar{N}_0$-dimensional vector

$$\mathbf{s} \triangleq \left[s_0, s_1, ..., s_{\bar{N}_0/2}, s_{-\bar{N}_0/2+1}, s_{-\bar{N}_0/2+2}, ..., s_{-2}, s_{-1}\right]^T. \tag{70}$$

2. *Azimuth estimation* - After solving the problem

$$\hat{p} = \arg \max_{\tilde{p}\in\{-\bar{N}_0/2+1, -\bar{N}_0/2+2, ..., \bar{N}_0/2\}} \left|s_{\tilde{p}}\right|, \tag{71}$$

the estimate (see eq. (63))

$$\hat{\phi} = \arcsin\left(2\frac{\hat{p}}{\bar{N}_0}\right) \tag{72}$$

of the target azimuth ϕ is evaluated. Note that the angular resolution provided by the DFT computed in step 1. improves as $\bar{N}_0$ increases.

The deterministic algorithm and its extension illustrated above have the following relevant properties: a) their derivation is based on a well defined mathematical model originating from our knowledge of the propagation of electromagnetic waves and of the radar system (and, in particular, of the geometry of its array and of the processing it accomplishes); b) if they fail detecting a given target, or generate inaccurate estimates of its range and/or azimuth, the causes of such events can be identified; c) being based on the DFT and other simple formulas, they are computational efficient.

An alternative to the approach to algorithm design illustrated above is offered by ML methods [12]. In fact, if such methods are employed, the inner structure of the considered radar system and the physical laws on which its operation is based can be ignored, since the required information are automatically extracted by an algorithm able to *learn the regularities characterizing the set of observed data*. Let us reconsider now the detection and estimation problem described above from this new perspective and show how a solution based on ML methods can be devised. To this aim, we take into consideration again a FMCW radar system equipped with the antenna

array shown in Fig. 2-a) and assume that it is employed to perform a measurement campaign. In this campaign, N_t independent trials are accomplished in the presence of a single point target or in the absence of it; in each trial, the couple $[\mathbf{r}_0,\ \mathbf{r}_1]$ of noisy vectors (see eq. (51)) is acquired and stored in a database together with the target range and azimuth (when the target is present). In the following, $[\mathbf{r}_{q,0}, \mathbf{r}_{q,1}]$ and

$$\mathbf{t}_q \triangleq [d_q, R_q, \phi_q]^T \tag{73}$$

denote the value of the couple $[\mathbf{r}_0,\ \mathbf{r}_1]$ and the associated *label* acquired in the q-th trial (with $q = 0,\ 1,\ ...\ ,\ N_t - 1$); here, $d_q = -1$ (1) if the target is absent (present), and R_q and ϕ_q represent the target range and azimuth, respectively, in the same trial if $d_q = 1$ (if $d_q = -1$, the values of both R_q and ϕ_q are irrelevant). In this case, the ML approach consists in processing the dataset

$$\mathcal{D}_i \triangleq \{[\mathbf{r}_{q,0},\ \mathbf{r}_{q,1}]\,,\ \mathbf{t}_q; q = 0, 1, ..., N_t - 1\} \tag{74}$$

to learn how to detect the presence of a target on the basis of a new couple $[\mathbf{r}_0,\ \mathbf{r}_1]$ and, if a target is detected, how to estimate its position. The accuracy of the algorithm resulting from the learning phase (i.e., from *training*) depends not only on the adopted ML method, but also on the size of $\mathcal{D}_i$ (i.e., on N_t). Generally speaking, the use of ML methods requires the availability of a large set of measurements, i.e. a large N_t (say, at least, a few thousands). Unluckily, any ML method extracting the required knowledge directly from $\mathcal{D}_i$ (74) has to process high dimensional vectors if the size N of the vectors $\mathbf{r}_0$ and $\mathbf{r}_1$ is large. Actually, the dimensionality of the given problem can be easily reduced by exploiting our prior knowledge. In fact, in developing our deterministic algorithm, we have learnt that essential information for target detection and estimation is provided by the complex couple $[X_{0,\hat{l}},\ X_{1,\hat{l}}]$ (see eqs. (59)-(64)), where $\hat{l}$ is expressed by eq. (58). These considerations suggest to:

a) *Pre-process* the couple $(\mathbf{r}_{q,0},\ \mathbf{r}_{q,1})$ in order to generate the vector

$$\mathbf{X}_q \triangleq \left[X_0^{(q)}, X_1^{(q)}\right]^T, \tag{75}$$

where $X_0^{(q)}$ and $X_1^{(q)}$ are the values taken on by the quantities $X_{0,\hat{l}}$ and $X_{1,\hat{l}}$, respectively, in the q-th trial (with $q = 0,\ 1,\ ...,\ N_t - 1$); $X_0^{(q)}$ and $X_1^{(q)}$ can be considered as highly informative data extracted from the received signal, i.e., briefly, as the *features* available in the considered problem.

b) Replace the set $\mathcal{D}_i$ (74) with the new set

$$\mathcal{D} \triangleq \{\mathbf{X}_q,\ \mathbf{t}_q; q = 0, 1, ..., N_t - 1\}, \tag{76}$$

that consists of low dimensional vectors only, and use it to train the considered ML method; when this occurs, the last set is called *training set*.

Once training is over, the ML algorithm resulting from it is able to infer the unknown value of $\mathbf{t}_q$ (73) for any new vector $\mathbf{X}_q$ (75) of noisy data (with $q > N_t - 1$); in other words, it is able to predict: a) d_q; b) R_q and ϕ_q if a target is detected. It is important to point out that any ML algorithm predicting d_q solves a *binary classification problem*, since it assigns a new observation to one of two *categories* of noisy data, one associated with the presence of a target, the other one with its absence; in other words, *the algorithm is exploited to recognise a specific pattern in the noisy observations*. If the considered ML algorithm is also able to *predict* the value of the couple (R_q, ϕ_q) (i.e., of two continuous variables), it solves a *regression problem* too. In the considered radar system, different ML algorithms can be employed to learn classification and regression rules from the training set $\mathcal{D}$ (76). Moreover, all such algorithms can be considered as specific instances of the so called *supervised learning*, as shown in the following paragraph. Generally speaking, supervised learning techniques can be employed when:

1) A training set

$$\mathcal{D} \triangleq \{(\mathbf{r}_q, \mathbf{t}_q)\,;\ q = 0, 1, ..., N_t - 1\}, \tag{77}$$

collecting N_t, D_r-dimensional real *observations* (also called *covariates*, or *domain points*, or *explanatory variables*) $\{\mathbf{r}_q; q = 0,\ 1,\ ...,\ N_t - 1\}$, with

$$\mathbf{r}_q \triangleq [r_{q,0}, r_{q,1}, ..., r_{q,D_r-1}]^T, \tag{78}$$

and the associated D_t-dimensional real *labels* (also called *dependent variables* or *responses*) $\{\mathbf{t}_q;\ q = 0,\ 1,\ ...,\ N_t - 1\}$, with

$$\mathbf{t}_q \triangleq [t_{q,0}, t_{q,1}, ..., t_{q,D_t-1}]^T, \tag{79}$$

is available.

2) There exists some mechanism relating the variable $\mathbf{r}_q$ to the variable $\mathbf{t}_q$ for any q.

The last point is a fundamental one, since it does not make any sense to develop rules applicable to unseen examples in the absence of some assumptions about the mechanism underlying data generation; the set of these assumptions is known as the *inductive bias*.

B. Supervised learning

Supervised learning is a branch of ML frequently employed in the field of colocated MIMO radars for solving classification and regression problems. In this paragraph, we discuss some basic methods for supervised learning and analyse the approaches adopted in their derivation. Specific attention is paid to some classification algorithms that can be easily employed for target detection in radar systems.

1) Formulation of the supervised learning problem: Generally speaking, supervised learning concerns the identification of the conditional *probability density function* (pdf) $f(\mathbf{t}|\mathbf{r})$ (also called *predictive distribution*) minimizing the average *generalization loss*

$$L_p(\tilde{\mathbf{t}}) \triangleq \mathrm{E}_{f(\mathbf{t},\mathbf{r})}\left\{\ell\left(\mathbf{t},\tilde{\mathbf{t}}\left(\mathbf{r}\right)\right)\right\}; \tag{80}$$

here, $\mathrm{E}_{f(\mathbf{x})}\{\cdot\}$ denotes the expectation evaluated with respect to the joint $f(\mathbf{x})$, $\tilde{\mathbf{t}}(\mathbf{r})$ is a *prediction* of the label $\mathbf{t}$ computed on the basis of the observation $\mathbf{r}$ and $\ell(\cdot,\cdot)$ is a given *cost function*. If the label of each observation is *one-dimensional* (1D) and is real, the cost functions

$$\ell_2\left(t,\hat{t}\right) = (t-\hat{t})^2 \tag{81}$$

and

$$\ell_0\left(t,\hat{t}\right) = \begin{cases} 1 \text{ if } t=\hat{t} \\ 0 \text{ elsewhere} \end{cases} \tag{82}$$

are often employed for regression and binary classification, respectively. It is well known that, if the posterior pdf $f(\mathbf{t}|\mathbf{r})$ is *known*, the minimum value of the loss $L_p(\tilde{\mathbf{t}})$ (80) is achieved by selecting the *optimal prediction* (e.g., see [12, Par. III.C, eq. (4)])

$$\hat{t}\left(\mathbf{r}\right) = \arg\min_{\tilde{\mathbf{t}}} \mathrm{E}_{f(\mathbf{t}|\mathbf{r})}\left[\ell\left(\mathbf{t},\tilde{\mathbf{t}}\right)|\mathbf{r}\right], \tag{83}$$

whatever cost function is selected.

Supervised learning methods are employed when the conditioned pdf $f(\mathbf{t}|\mathbf{r})$ (or the joint pdf $f(\mathbf{t},\mathbf{r})$) is unknown, but a training set $\mathcal{D}$, collecting N_t distinct data generated on the basis of it and structured according to eq. (76), is available. The objective of these methods is to generate a *predictor*, denoted $\hat{t}_{\mathcal{D}}(\mathbf{r})$, exclusively based on the set $\mathcal{D}$ and whose performance, in terms of generalization loss, is as close as possible to that of the optimal predictor $\hat{t}(\mathbf{r})$ (83); this means that the loss evaluated for the prediction of the label associated with a new observation should be as small as possible. The derivation of the predictor $\hat{t}_{\mathcal{D}}(\mathbf{r})$ can be formulated as an optimization problem, whose form depends on the specific assumptions we make about the model that is being learnt. In fact, a *frequentist approach* or a *Bayesian approach* can be adopted, as illustrated in the following two paragraphs.

2) The frequentist approach to supervised learning: The frequentist approach relies on the assumption that all the points of the set $\mathcal{D}$ (77) are generated *independently* on the basis of the same *unknown* joint pdf $f(\mathbf{r},\mathbf{t})$, that is

$$(\mathbf{r}_q,\mathbf{t}_q) \sim f\left(\mathbf{r},\mathbf{t}\right) = f\left(\mathbf{t}|\mathbf{r}\right)\, f\left(\mathbf{r}\right), \tag{84}$$

with $q = 0,1,...,N_t-1$. Under this assumption, two possible approaches can be adopted to derive the above mentioned predictor $\hat{t}_{\mathcal{D}}(\mathbf{r})$, namely: a) *separate learning* and *inference*; b) direct inference via *empirical risk minimization* (ERM). The first approach consists in learning an approximation, denoted $f_{\mathcal{D}}(\mathbf{t}|\mathbf{r})$, of the conditional pdf $f(\mathbf{t}|\mathbf{r})$, and in using the former pdf in place of the latter one to derive the expression of the predictor $\hat{t}_{\mathcal{D}}(\mathbf{r})$ on the basis of eq. (83). The second approach, instead, aims at directly learning $\hat{t}_{\mathcal{D}}(\mathbf{r})$ by solving the problem

$$\hat{t}_{\mathcal{D}}\left(\mathbf{r}\right) = \arg\min_{\tilde{\mathbf{t}}} L_{\mathcal{D}}\left(\tilde{\mathbf{t}}\left(\mathbf{r}\right)\right), \tag{85}$$

where

$$L_{\mathcal{D}}\left(\tilde{\mathbf{t}}\left(\mathbf{r}\right)\right) \triangleq \frac{1}{N_t}\sum_{q=0}^{N_t-1}\ell\left(\mathbf{t}_q,\tilde{\mathbf{t}}\left(\mathbf{r}_q\right)\right) \tag{86}$$

is the so called *empirical loss*. In both cases, the optimization of a *set of parameters* characterizing the model selected for the conditional pdf $f_{\mathcal{D}}(\mathbf{t}|\mathbf{r})$ or that chosen for the predictor $\hat{t}_{\mathcal{D}}(\mathbf{r})$ is required. However, the first approach is more flexible than the second one since, in principle, the approximate pdf $f_{\mathcal{D}}(\mathbf{t}|\mathbf{r})$ it generates can be exploited to derive the predictor $\hat{t}_{\mathcal{D}}(\mathbf{r})$ for any cost function; on the contrary, the solution of the problem

(85) holds for a specific cost function only. Moreover, it should be kept in mind that, if the first approach is adopted, two options are available. The first option consists in learning a *discriminative probabilistic model*, i.e. in learning directly an approximation of the posterior $f(\mathbf{t}|\mathbf{r})$. On the contrary, the second option consists in learning a *generative probabilistic model*, i.e. in learning the joint pdf $f(\mathbf{t}, \mathbf{r})$ and, then, in deriving an estimate of the posterior $f(\mathbf{t}|\mathbf{r})$ from it.

Let us see now how the general principles illustrated above can be employed to solve a specific regression problem concerning the first FMCW radar system described in the previous paragraph and equipped with the array shown in Fig. 2-a). In this case, we assume: a) the presence of a single point target placed at a fixed and known range R; b) the availability of the synthetically generated dataset[7] (see eq. (77))

$$\mathcal{D} \triangleq \{r_q,\, t_q; q = 0, 1, ..., N_t - 1\}, \tag{87}$$

where[8]

$$t_q \triangleq \phi_q, \tag{88}$$

$$r_q = \Delta\hat{\psi}_q \tag{89}$$

is an estimate of the phase difference

$$\Delta\psi_q \triangleq \psi_{q,1} - \psi_{q,0} \tag{90}$$

and $\psi_{q,0}$ ($\psi_{q,1}$) is the phase of the sinusoidal oscillation associated with the considered target and observed on the first (second) RX antenna for any q (see eqs. (35)-(41) and (45)). Moreover, in generating our dataset, the following choices have been made:

a) the distance d between adjacent virtual channels is equal to $\lambda/4$;

b) the target range R is equal to 3.0 m, whereas the target azimuth ϕ_q is uniformly distributed over the interval $[\phi_m,\ \phi_M] = [-60°\ 60°]$, respectively, for any q (this interval is comparable to the horizontal FOV of a real radar system);

c) the amplitude $a_v^{(q)}$ characterizing the sinusoid observed on the v-th virtual antenna is randomly selected in the interval $[0.4,\ 1.2]$ V for any q (see eq. (7));

d) the random variable $a_v^{(q)}$ is independent of $a_u^{(p)}$ for any $u \neq v$ and/or $p \neq q$;

e) the observation r_q (89) is generated on the basis of eqs. (64) and (75), i.e. as $\Delta\hat{\psi}_q = \angle X_0^{(q)} \, (X_1^{(q)})^*$ for any q.

Moreover, the following choices have been made for the parameters of the radar system:

a) the generated frequency modulated waveform is characterized by $\mu = 7.8125 \cdot 10^{12}$ Hz s^{-1}, $T = 256\,\mu$s and $T_R = 64\,\mu$s;

b) the sampling period employed at the receive side is $T_s = 0.25$ μs and $N = 512$ time-domain samples are acquired from each of the two RX antennas;

c) the standard deviation of the noise affecting these samples is $\sigma_w = \sqrt{2}$ V (see eq. (7));

d) the oversampling factor $M_r = 4$ and the threshold $P_{th} = 0.5$ V^{2}Hz^{-1} are employed by the detection algorithm based on eqs. (58)-(59).

In this case, our objective is to derive a predictor of the the azimuth ϕ_q associated with the new observation $\Delta\hat{\psi}_q$ for any $q > N_t - 1$. To solve this problem, we adopt the discriminative probabilistic model

$$f\left(t|r_q, \mathbf{w}\right) = \mathcal{N}\left(t; \mu\left(r_q, \mathbf{w}\right), \beta^{-1}\right), \tag{91}$$

where

$$\mu\left(r_q, \mathbf{w}\right) \triangleq \sum_{j=0}^{M} w_j \, \varphi_j\left(r_q\right) = \mathbf{w}^T \, \boldsymbol{\varphi}\left(r_q\right), \tag{92}$$

M is the *order* of the model,

$$\mathbf{w} \triangleq \left[w_0, w_1, ..., w_M\right]^T \tag{93}$$

is a vector collecting $M + 1$ distinct real parameters (called *weights*),

$$\boldsymbol{\varphi}\left(r_q\right) \triangleq \left[\varphi_0(r_q), \varphi_1(r_q), ..., \varphi_M(r_q)\right]^T \tag{94}$$

is the so called *feature* vector, $\{\varphi_j(x);\, j = 0,\, 1,\, ...,\, M\}$ are $M+1$ non linear functions and β^{-1} is the variance of the noise affecting the labels. In the following, we assume that

$$\varphi_j(x) = x^j \tag{95}$$

[7]This dataset and all the other synthetic datasets processed in our work have been generated by resorting to various functions available in the MATLAB and/or Python environment.

[8]Note that, in this case, $d_q = 1$ and $R_q = R$ in eq. (73), so that the label $\mathbf{t}_q$ turns into a scalar.

for $j = 0, 1, ..., M$; consequently, eq. (92) becomes

$$\mu(r_q, \mathbf{w}) \triangleq w_0 + \sum_{j=1}^{M} w_j \, r_q^j. \tag{96}$$

It is worth noting that:

a) Adopting the probabilistic model (91) with the mean $\mu(r_q, \mathbf{w})$ (96) is tantamount to postulating a polynomial dependence of the label ϕ_q on the corresponding observation $\Delta\hat{\psi}_q$.

b) The selected model depends on its order M and on the $(M+2)$-dimensional parameter vector $\boldsymbol{\theta} \triangleq \left[\mathbf{w}^T, \beta\right]^T$.

c) The parameter M defines the number of degrees of freedom available in the model and, consequently, determines its *bias*.

As far as the last point is concerned, it is important to mention that, if M is too small, the resulting predictor may *underfit* the observations, since it is unable to accurately represent this dependence on their labels. On the contrary, if M is too large, the model is able to account for the observations of the training set, but it may generate inaccurate predictions; in other words, it memorizes the training set, but it is unable to generalise what has learnt to new examples. The last problem is known as *overfitting*. For instance, in the considered problem, good results are obtained if $M = 3$ is selected.

If the ERM approach is adopted to adjust the parameters of the probabilistic model (91) (and, in particular, the weight vector $\mathbf{w}$ (93)) in an optimal fashion, the obtained result depends on the selected cost function and cannot be always put in a closed form. However, if the cost function $\ell_2(t, \hat{t})$ (81) is chosen and noise is neglected (i.e., β^{-1} is assumed to be very small), a closed form expression can be derived for $\hat{\mathbf{w}}$ for any M. In fact, under these assumptions, it can be proved that:

1) The optimal predictor $\hat{t}_{\mathcal{D}}(\mathbf{r})$ (85) becomes (e.g., see [71, Sect. 3.1.1, eq. (3.20)])

$$\hat{t}_{\mathcal{D}}(r_q) = \mu\left(r_q, \hat{\mathbf{w}}\right), \tag{97}$$

where

$$\hat{\mathbf{w}} = \arg\min_{\tilde{\mathbf{w}}} L_{\mathcal{D}}\left(\tilde{\mathbf{w}}\right), \tag{98}$$

$\tilde{\mathbf{w}}$ denotes a trial value of $\mathbf{w}$ and

$$L_{\mathcal{D}}\left(\tilde{\mathbf{w}}\right) \triangleq \frac{1}{N_t} \sum_{q=0}^{N_t-1} \left(t_q - \mu\left(r_q, \hat{\mathbf{w}}\right)\right)^2 \tag{99}$$

is the empirical loss (see eq. (86)).

2) The solution of the minimization problem appearing in the RHS of eq. (98) is

$$\hat{\mathbf{w}} = \left(\boldsymbol{\Phi}_{\mathcal{D}}^T \boldsymbol{\Phi}_{\mathcal{D}}\right)^{-1} \boldsymbol{\Phi}_{\mathcal{D}}^T \, \boldsymbol{t}_{\mathcal{D}}, \tag{100}$$

where

$$\boldsymbol{\Phi}_{\mathcal{D}} \triangleq \left[\boldsymbol{\varphi}(r_0), \boldsymbol{\varphi}(r_1), ..., \boldsymbol{\varphi}(r_{N_t-1})\right] \tag{101}$$

is a $N_t \times (M+1)$ matrix and

$$\mathbf{t}_{\mathcal{D}} \triangleq \left[t_0, t_1, ..., t_{N_t-1}\right]^T. \tag{102}$$

Given the weight vector $\hat{\mathbf{w}}$ (100), the estimate

$$\hat{\beta}^{-1} \triangleq \frac{1}{N_t} \sum_{q=0}^{N_t-1} \left(t_q - \hat{\mathbf{w}}^T \boldsymbol{\varphi}\left(r_{\boldsymbol{q}}\right)\right)^2, \tag{103}$$

of the noise variance β^{-1} can be easily evaluated.

Training the algorithm illustrated above consists in computing the weight vector $\hat{\mathbf{w}}$ (100) on the basis of the available training set $\mathcal{D}$ (87). Once training has been carried out, the generalization capability of the resulting algorithm can be assessed by evaluating the *empirical loss* (86) on the basis of a different dataset, called *test set* $\mathcal{D}_{ts}$ and collecting $\bar{N}_t$ observations generated in the same way as the ones of $\mathcal{D}$, but in an independent fashion.

In our computer simulations, the training set $\mathcal{D}$ (87) and the test set $\mathcal{D}_{ts}$ consist of $N_t = 200$ and $\bar{N}_t = 25$ observations, respectively; the points of these sets are represented in Figs. 5 and 6, respectively. First, the weight vector $\hat{\mathbf{w}}$ (100) and the estimate $\hat{\beta}^{-1}$ (103) of the noise variance have been computed on the basis of $\mathcal{D}$. Then, the accuracy of the resulting regression algorithm has been assessed on $\mathcal{D}_{ts}$. The predictions associated with the points of $\mathcal{D}_{ts}$ are represented in Fig. 6; in this figure, two (red) straight lines, generated on the basis of the linear equations

$$t = \mu(r, \hat{\mathbf{w}}) \pm \hat{\beta}^{-1/2}, \tag{104}$$

are also shown to highlight the meaning of the noise standard deviation $2\hat{\beta}^{-1/2}$. These results lead to the conclusion that, in the considered scenario, the developed regression method is able to predict the azimuth of a target with good accuracy. This is confirmed by the fact that the empirical loss computed over the set $\mathcal{D}_{ts}$ (i.e., the *generalization loss*) is close to the empirical loss $L_{\mathcal{D}}(\hat{\mathbf{w}})$ evaluated over the set $\mathcal{D}$ (see eq. (86)); in fact, the *root mean square error*[9] (RMSE) evaluated over $\mathcal{D}$ is equal to[10] $1.7°$, whereas that computed over $\mathcal{D}_{ts}$ is equal to $1.3°$.

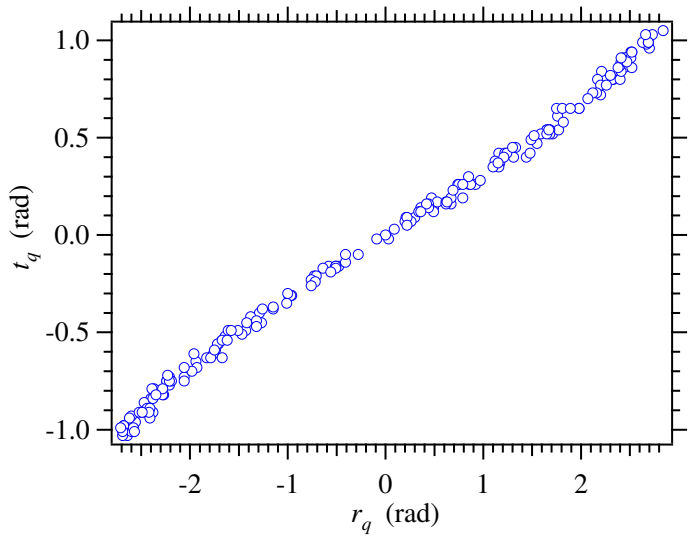


Figure 5: Representation of the points of the synthetically generated training set $\mathcal{D}$ (87); $N_t = 200$ is assumed.

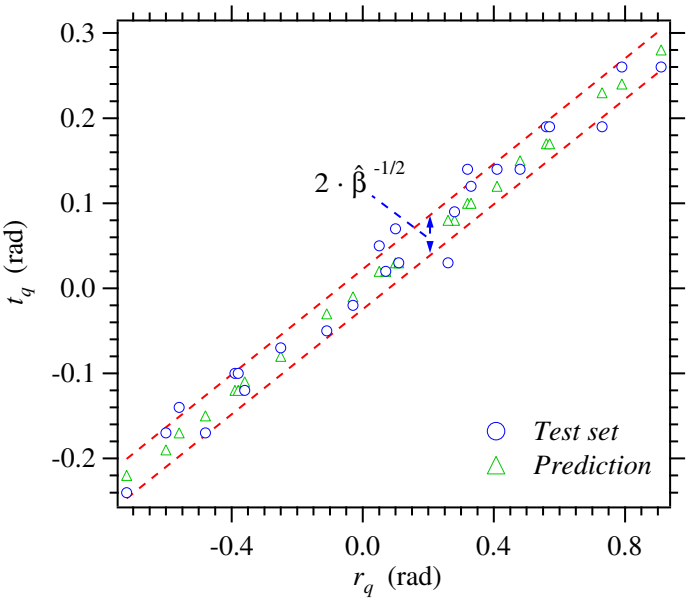


Figure 6: Representation of the points of the synthetically generated test set $\mathcal{D}_{ts}$ (blue circles) and of the corresponding predictions (green triangles) evaluated on the basis of eq. (97); $\bar{N}_t = 25$ is assumed. Two straight lines, expressed by eq. (104), are also shown.

In general, if the *discriminative probabilistic model* adopted to solve a specific regression problem is represented by the parametric pdf $f(\mathbf{t}|\mathbf{r}, \boldsymbol{\theta})$, a closed form expression for the optimal value

$$\hat{\boldsymbol{\theta}} = \arg\min_{\tilde{\boldsymbol{\theta}}} L_{\mathcal{D}}\left(\tilde{\boldsymbol{\theta}}\right), \tag{105}$$

[9]This parameter represents the square root of the empirical loss.
[10]The RMSE computed over $\mathcal{D}$ is given by $\hat{\beta}^{-1/2}$.

of the D_θ-dimensional parameter vector $\boldsymbol{\theta}$ is unavailable in most cases. When this occurs, iterative optimization techniques, like the *stochastic gradient descent* (SGD) method, can be employed to estimate $\hat{\boldsymbol{\theta}}$. The application of the SGD to the considered problem leads easily to the recursive equation

$$\hat{\boldsymbol{\theta}}^{(i+1)} = \hat{\boldsymbol{\theta}}^{(i)} + \gamma^{(i+1)} N_S^{-1} \sum_{q \in \mathcal{S}} \nabla_{\tilde{\boldsymbol{\theta}}} \ell \left(\mathbf{t}_q, \hat{\mathbf{t}} \left(\mathbf{r}_q, \ \tilde{\boldsymbol{\theta}} \right) \right) |_{\tilde{\boldsymbol{\theta}} = \hat{\boldsymbol{\theta}}^{(i)}}, \tag{106}$$

with $i = 0, 1, ..., N_E - 1$; here, $\hat{\boldsymbol{\theta}}^{(i)}$ denotes estimate of $\hat{\boldsymbol{\theta}}$ computed in the i-th recursion, $\mathcal{S}$ is a set of N_S integers randomly selected in the set $\{0, 1, ..., N_t - 1\}$ (with $N_S < N_t$), $\gamma^{(i)}$ is the *learning rate* adopted in the i-th iteration and $\nabla_{\mathbf{x}} f(\mathbf{x})$ denotes the gradient of the function $f(\mathbf{x})$. It can be proved that, if the learning rate schedule (i.e., the sequence $\{\gamma^{(i)}\}$) satisfies the so called *Robbins-Monro conditions*, the SGD converges to the optimal solution, provided that the function $L_{\mathcal{D}}(\tilde{\boldsymbol{\theta}})$ is strictly convex. The initial value $\hat{\boldsymbol{\theta}}^{(0)}$ can be either randomly selected or it can be inherited from the training procedure accomplished another model; the last solution represents a specific application of the so called *transfer learning* technique (see Paragraph VII-A). Iterations are stopped when negligible variations are observed in the estimates generated by consecutive recursions or an upper limit set on the overall number of recursions is reached. Once the final estimate of $\hat{\boldsymbol{\theta}}$ has been computed on the basis of the available training set, the generalization capability of the resulting algorithm can be assessed by evaluating the *empirical loss* (86) on a given *test set* $\mathcal{D}_{ts}$.

Finally, it is worth mentioning that the selection of the parameter D_θ (i.e., of the *model complexity*) plays a fundamental role in the considered problem. In fact, if its value is too small (too large), the resulting regression method can suffer from underfitting (overfitting). The overfitting phenomenon is usually prevented by including a *regularization* term in the training of the adopted model. For instance, if the optimization problem (105) is considered, this result can be achieved by adopting the cost function

$$L_D \left(\tilde{\boldsymbol{\theta}} \right) + \frac{\lambda}{N_t} \left\| \tilde{\boldsymbol{\theta}} \right\|^2, \tag{107}$$

where λ is a real positive weight influencing the predictive capability of the resulting solution and $||\mathbf{x}||$ is the *Euclidean norm* of the vector $\mathbf{x}$.

3) The Bayesian approach to supervised learning: The frequentist approach illustrated in the previous paragraph leads to the identification of a specific probabilistic model through the estimation of its parameter vector $\boldsymbol{\theta}$. The Bayesian approach, instead, consists in formulating our uncertainty about the parameters of the adopted probabilistic model in statistical terms, i.e. in treating its parameter vector $\boldsymbol{\theta}$ as a *random vector*. In this paragraph, we show how the specific regression problem analysed in the previous paragraph can be tackled from this perspective; for this reason, we assume that each observation and its label are 1D (i.e., $D_t = D_r = 1$), so that all the labels of the training set $\mathcal{D}$ (77) and the associated observations can be collected in the N_t-dimensional vectors $\mathbf{t}_{\mathcal{D}}$ (102) and

$$\mathbf{r}_{\mathcal{D}} \triangleq [r_0, r_1, ..., r_{N_t - 1}]^T, \tag{108}$$

respectively. If the discriminative probabilistic model (91) introduced in the previous paragraph is exploited, a Bayesian method based on it can be developed as follows. To begin, the joint pdf

$$f\left(t, \mathbf{t}_{\mathcal{D}}, \mathbf{w} | r_q, \mathbf{r}_{\mathcal{D}}, \right) = f\left(\mathbf{t}_{\mathcal{D}}, \mathbf{w} | \mathbf{r}_{\mathcal{D}}, \alpha, \beta\right) \ f\left(t | r_q, \mathbf{w}\right) \tag{109}$$

is considered in place of the pdf $f(t|r_q, \mathbf{w})$ (91); here, β^{-1} is the variance of the noise affecting the labels,

$$f\left(\mathbf{t}_{\mathcal{D}}, \mathbf{w} | \mathbf{r}_{\mathcal{D}}, \alpha, \beta\right) = f\left(\mathbf{t}_{\mathcal{D}} | \mathbf{r}_{\mathcal{D}}, \mathbf{w}, \beta\right) \ f\left(\mathbf{w} | \alpha\right) \tag{110}$$

is the joint probability of the $(M+1)$-dimensional weight vector $\mathbf{w}$ (93) and the label vector $\mathbf{t}_{\mathcal{D}}$ (102) conditioned on $\mathbf{r}_{\mathcal{D}}$ (108), on the *hyperparameter* α and on the parameter β, and $f(\mathbf{w}|\alpha)$ is the prior pdf of $\mathbf{w}$. The Gaussian model

$$\begin{aligned} f(\mathbf{w}|\alpha) &= \mathcal{N}\left(\mathbf{w}; \mathbf{0}, \alpha^{-1} \mathbf{I}_{M+1}\right) \\ &= \left(\frac{\alpha}{2\pi}\right)^{(M+1)/2} \exp\left\{-\frac{\alpha}{2} \mathbf{w}^T \mathbf{w}\right\} \end{aligned} \tag{111}$$

is employed for the second pdf appearing in the RHS of eq. (110) (e.g., see [71, Sect. 1.2.4, p. 30, eq. (1.65)]); here, $\mathbf{I}_N$ is the $N \times N$ unit matrix and α represents the precision of the last pdf. The first pdf appearing in the RHS of eq. (110), instead, represents a *likelihood function* expressing how likely the response $\mathbf{t}_{\mathcal{D}}$ are, given $\mathbf{r}_{\mathcal{D}}$, $\mathbf{w}$ and β; this function can be factored as

$$f\left(\mathbf{t}_{\mathcal{D}} | \mathbf{r}_{\mathcal{D}}, \mathbf{w}, \beta\right) = \prod_{k=0}^{N_t - 1} f\left(t_k | r_k, \mathbf{w}, \beta\right), \tag{112}$$

and, consequently, can be expressed in terms of the probabilistic model (91).

Given the joint pdf $f(t, \mathbf{t}_{\mathcal{D}}, \mathbf{w}, |r_q, \mathbf{r}_{\mathcal{D}})$ (109), the *predictive distribution* $f(t|r_q, \mathbf{r}_{\mathcal{D}}, \mathbf{t}_{\mathcal{D}})$ can be evaluated as

$$f\left(t|r_q, \mathbf{r}_{\mathcal{D}}, \mathbf{t}_{\mathcal{D}}\right) = \frac{1}{f\left(\mathbf{t}_{\mathcal{D}}|\mathbf{r}_{\mathcal{D}}, \alpha, \beta\right)} \int f\left(t, \mathbf{t}_{\mathcal{D}}, \mathbf{w}|r_q, \mathbf{r}_{\mathcal{D}}\right) d\mathbf{w}, \tag{113}$$

where

$$\begin{aligned} f(\mathbf{t}_{\mathcal{D}}|\mathbf{r}_{\mathcal{D}}, \alpha, \beta) &= \int f\left(\mathbf{t}_{\mathcal{D}}, \mathbf{w}|\mathbf{r}_{\mathcal{D}}, \alpha, \beta\right)\, d\mathbf{w} \\ &= \int f\left(\mathbf{t}_{\mathcal{D}}|\mathbf{r}_{\mathcal{D}}, \mathbf{w}, \beta\right) f\left(\mathbf{w}|\alpha\right) d\mathbf{w} \end{aligned} \tag{114}$$

is a *marginal likelihood*. The expression (113) can be also reformulated as follows. Substituting the RHS of eq. (110) in that of eq. (109) and the resulting factorization in the RHS of eq. (113) yields

$$\begin{aligned} f\left(t|r_q, \mathbf{r}_{\mathcal{D}}, \mathbf{t}_{\mathcal{D}}\right) \quad = \quad & \int \frac{f\left(\mathbf{t}_{\mathcal{D}}|\mathbf{r}_{\mathcal{D}}, \mathbf{w}, \beta\right)\, f\left(\mathbf{w}|\alpha\right)}{f\left(\mathbf{t}_{\mathcal{D}}|\mathbf{r}_{\mathcal{D}}, \alpha, \beta\right)} \\ & \cdot f\left(t|r_q, \mathbf{w}\right) d\mathbf{w}. \end{aligned} \tag{115}$$

Then, since

$$\frac{f\left(\mathbf{t}_{\mathcal{D}}|\mathbf{r}_{\mathcal{D}}, \mathbf{w}, \beta\right) f\left(\mathbf{w}|\alpha\right)}{f\left(\mathbf{t}_{\mathcal{D}}|\mathbf{r}_{\mathcal{D}}, \alpha, \beta\right)} = f\left(\mathbf{w}|\mathbf{r}_{\mathcal{D}}, \mathbf{t}_{\mathcal{D}}, \alpha, \beta\right), \tag{116}$$

eq. (115) can be rewritten as

$$f\left(t|r_q, \mathbf{r}_{\mathcal{D}}, \mathbf{t}_{\mathcal{D}}\right) = \int f\left(\mathbf{w}|\mathbf{r}_{\mathcal{D}}, \mathbf{t}_{\mathcal{D}}, \alpha, \beta\right)\, f\left(t|r_q, \mathbf{w}\right)\, d\mathbf{w}. \tag{117}$$

The last equation shows how the predictive distribution is influenced by our uncertainty about the weight vector; such an uncertainty is expressed by the pdf $f(\mathbf{w}|\mathbf{r}_{\mathcal{D}}, \mathbf{t}_{\mathcal{D}}, \alpha, \beta)$.

Let us apply now the mathematical results derived above to the considered regression problem. If the pdf $f(\mathbf{w}|\mathbf{r}_{\mathcal{D}}, \mathbf{t}_{\mathcal{D}}, \alpha, \beta)$ is assumed to be Gaussian and, in particular,

$$f\left(\mathbf{w}|\mathbf{r}_{\mathcal{D}}, \mathbf{t}_{\mathcal{D}}, \alpha, \beta\right) = \mathcal{N}(\mathbf{w}|\mu_{\mathcal{D}}, \sigma^2_{\mathcal{D}}), \tag{118}$$

where (e.g., see [71, Sec. 3.3, p. 153, eqs. (3.53)-(3.54)])

$$\mu_{\mathcal{D}} = \beta\, \sigma^2_{\mathcal{D}}\, \mathbf{\Phi}^T_{\mathcal{D}} \mathbf{t}_{\mathcal{D}}, \tag{119}$$

$$\sigma^2_{\mathcal{D}} = \left(\alpha \mathbf{I}_{M+1} + \beta \mathbf{\Phi}^T_{\mathcal{D}} \mathbf{\Phi}_{\mathcal{D}}\right)^{-1} \tag{120}$$

and the $N_t \times (M+1)$ matrix $\mathbf{\Phi}_{\mathcal{D}}$ is given by eq. (101), the expression

$$f\left(t|r_q, \mathbf{r}_{\mathcal{D}}, \mathbf{t}_{\mathcal{D}}\right) = \mathcal{N}(t; \mu(r_q), \sigma^2(r_q)) \tag{121}$$

can be derived from eq. (117) (e.g., see [71, Sec. 1.2.4, p. 31, eq. (1.69)] for a proof of this result); here,

$$\mu\left(r_q\right) = \beta\, \boldsymbol{\varphi}\left(r_q\right)^T \mathbf{S} \sum_{k=0}^{N_t-1} \boldsymbol{\varphi}\left(r_k\right)\, t_k, \tag{122}$$

$$\sigma^2\left(r_q\right) = \beta^{-1} + \boldsymbol{\varphi}\left(r_q\right)^T \mathbf{S}\, \boldsymbol{\varphi}\left(r_q\right), \tag{123}$$

$\boldsymbol{\varphi}(r_q)$ is the $(M+1)$-dimensional vector (94) and

$$\mathbf{S}^{-1} \triangleq \alpha\, \mathbf{I}_{M+1} + \beta \sum_{k=0}^{N_t-1} \boldsymbol{\varphi}\left(r_k\right)\ \ \boldsymbol{\varphi}\left(r_k\right)^T \tag{124}$$

is an $(M+1) \times (M+1)$ matrix. It is important to point out that the variance $\sigma^2(r_q)$ (123) of the predictive distribution $f\left(t|r_q, \mathbf{r}_{\mathcal{D}}, \mathbf{t}_{\mathcal{D}}\right)$ (121) (and, consequently, the accuracy of the prediction), unlike that of the Gaussian model $f(t|r_q, \mathbf{w})$ (91), is given by the sum of two terms; the first term originates from the noise affecting the labels, whereas the second one from our uncertainty about the parameter vector $\mathbf{w}$. Moreover, the second term is influenced by the considered observation (i.e., it depends on r_q); in practice, smaller values of the standard deviation $\sigma(r_q)$ are usually obtained when r_q is close to the observations of the training set.

The accuracy of the new regression algorithm described above has been assessed on the test set shown in Fig. 6 after training it on the set illustrated in Fig. 5; moreover, $\alpha = 0.05$ has been selected in this case. The prediction $\mu(r_q)$ evaluated on the basis of eq. (122) for each observation of the test set and the corresponding

standard deviation $\sigma(r_q)$ (i.e., the square roof of the RHS of eq. (123)) are represented in Fig. 7. The RMSE evaluated over the test set is equal to 1.4° and is approximately equal to the one computed over the training set. Note that this value is comparable to the ones computed for the predictor described in the previous paragraph (and based on a frequentist approach). For this reason, in this case, the Bayesian approach does not offer any advantage with respect to the frequentist one.

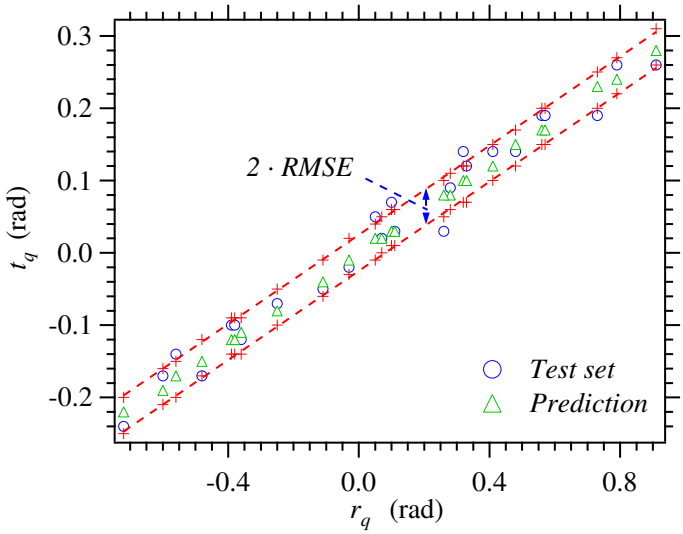


Figure 7: Representation of the regression technique based on the probabilistic model (121). The blue circles represent the true domain points, whereas the green triangles the corresponding predictions; the red curves are generated by interpolating the points generated on the basis of the two equations $t_q = \mu(r_q) \pm \sigma(r_q)$, with $r_q \in \mathcal{D}_{ts}$.

4) Specific methods for binary classification: In the remaining part of this section we focus on a specific supervised problem, namely *binary classification*, and develop two classification methods, based on *discriminative deterministic models*, to solve it. Moreover, we show how different classification methods can be combined to improve the overall accuracy. Note that, in general, classification methods based on *discriminative deterministic models* are able to represent the deterministic mapping between domain points and labels through specific functions, called *discriminant functions*. In the field of radar systems, these methods can be exploited for target detection.

The first method we take into consideration in this paragraph is the *support vector machine* (SVM) technique; in the following, we limit our analysis to its *linear form*, for simplicity, and assume that the label of each observation can take on only the values ± 1 (consequently, $D_t = 1$). The SVM technique processes the training set $\mathcal{D}$ (77) to find the *maximum-margin hyperplane*; this divides the subset of observations for which $t_q = 1$ from that for which $t_q = -1$ in a way that the distance between itself and the nearest point from either group is maximized. In the considered case, the above mentioned hyperplane can be defined as the set of points satisfying the equation

$$y(\mathbf{r}_q, \mathbf{w}) = 0 \tag{125}$$

for any q, where

$$y(\mathbf{r}_q, \mathbf{w}) \triangleq \mathbf{w}^T \mathbf{r}_q + b \ , \tag{126}$$

$\mathbf{r}_q$ is the q-th observation of $\mathcal{D}$ (see the definition (78)), $\mathbf{w}$ represents a D_r-dimensional weight vector (expressed by eq. (93), with $M = D_r - 1$) and b is a real parameter called *bias*. The adoption of a classification strategy based on the approach illustrated above relies on the implicit assumption that, if the parameters $\mathbf{w}$ and b appearing in eq. (126) are properly selected, the dataset $\mathcal{D}$ (77) is *linearly separable in the feature space*. In fact, when this occurs, two parallel hyperplanes separating the above mentioned two subsets of observations and having their mutual distances as large as possible can be found. If the observations of the set $\mathcal{D}$ are normalised, the hyperplanes delimiting the subsets of observations associated with $t_q = 1$ and $t_q = -1$ can be represented by the equations

$$y(\mathbf{r}_q, \mathbf{w}) = 1 \tag{127}$$

and

$$y(\mathbf{r}_q, \mathbf{w}) = -1, \tag{128}$$

respectively, i.e. briefly as

$$t_q\, y(\mathbf{r}_q, \mathbf{w}) = 1. \tag{129}$$

The last formula expresses the *canonical representation* of the decision hyperplanes. Based on the last result, the constraint according to which each point of the set $\mathcal{D}$ (77) must lie on the correct side of each of the two hyperplanes (i.e., that it must fall in the correct decision region) can be formulated as

$$t_q\, y(\mathbf{r}_q, \mathbf{w}) \geq 1 \tag{130}$$

for any q.

A method for the optimization of the parameters b and $\mathbf{w}$ appearing in eq. (126) can be developed as follows. The perpendicular distance of the point $\mathbf{r}_q$ from the decision hyperplane can expressed as

$$\frac{t_q\, y(\mathbf{r}_q)}{||\mathbf{w}||} = \frac{t_q\left(\mathbf{w}^T\mathbf{r}_q + b\right)}{||\mathbf{w}||} \tag{131}$$

for any q; its minimum value over the set $\mathcal{D}$ is known as *margin*. The optimal choice $(\hat{\mathbf{w}},\, \hat{b})$ of the parameters $(\mathbf{w},\, b)$ is the one *maximizing the margin* and, consequently, can be evaluated as

$$\left(\hat{\mathbf{w}}, \hat{b}\right) = \arg\max_{\tilde{\mathbf{w}}, \tilde{b}} \left\{ \frac{1}{\|\tilde{\mathbf{w}}\|} \min_q \left[t_q(\tilde{\mathbf{w}}^T\mathbf{r}_q + \tilde{b}) \right] \right\}\ , \tag{132}$$

where $(\tilde{\mathbf{w}},\, \tilde{b})$ denotes a trial value of the couple $(\mathbf{w},\, b)$; the data points closest to the max-margin hyperplane are called *support vectors*. Unluckily, the optimization problem appearing in the RHS of eq. (132) does not admit a simple solution. However, since there is always at least one support vector satisfying eq. (129), this problem can be reformulated in a simpler form, i.e. as the maximization of $||\tilde{\mathbf{w}}||^{-1}$ or, equivalently, as

$$\hat{\mathbf{w}} = \arg\min_{\tilde{\mathbf{w}}} \frac{||\tilde{\mathbf{w}}||^2}{2} \tag{133}$$

under the constraint expressed by eq. (130); note that the parameter $\tilde{b}$ is no more visible in the last formulation, but its value is implicitly determined by the above mentioned constraint. To solve the constrained optimization problem (133), the *Lagrangian* function

$$\mathcal{L}\left(\tilde{\mathbf{w}}, \tilde{b}, \tilde{\mathbf{a}}\right) \triangleq \frac{||\tilde{\mathbf{w}}||^2}{2} - \sum_{q=0}^{N_t-1} \tilde{a}_q \{ t_q(\tilde{\mathbf{w}}^T\mathbf{r}_q + \tilde{b}) - 1 \}. \tag{134}$$

is defined; this function depends not only on the parameters $\tilde{\mathbf{w}}$ and $\tilde{b}$, but also on the non negative parameters $\{\tilde{a}_q\}$, called *Lagrange multipliers* and collected in the vector $\tilde{\mathbf{a}} \triangleq [\tilde{a}_0,\, \tilde{a}_1,\, ...,\, \tilde{a}_{N_t-1}]^T$ (the q-th element of this vector is associated with the q-th constraint expressed by eq. (130)). Taking the partial derivatives of the function $\mathcal{L}(\tilde{\mathbf{w}}, \tilde{b}, \tilde{\mathbf{a}})$ (134) with respect to $\tilde{\mathbf{w}}$ and $\tilde{b}$ and setting them to zero results in

$$\sum_{q=0}^{N_t-1} \tilde{a}_q\, t_q = 0 \tag{135}$$

and

$$\tilde{\mathbf{w}} = \sum_{q=0}^{N_t-1} \tilde{a}_q\, t_q\, \mathbf{r}_q, \tag{136}$$

respectively. Then, substituting eqs. (135)-(136) in the RHS of eq. (134) produces the so called *dual representation* of the margin maximization problem. Solving the last problem requires maximizing the function

$$\mathcal{L}\left(\tilde{\mathbf{a}}\right) \triangleq \sum_{q=0}^{N_t-1} \tilde{a}_q - \frac{1}{2} \sum_{q=0}^{N_t-1} \sum_{k=0}^{N_t-1} \tilde{a}_q\, \tilde{a}_k\, t_q\, t_k (\mathbf{r}_q^T\, \mathbf{r}_k) \tag{137}$$

with respect to the vector $\tilde{\mathbf{a}}$, under the set of constraints $\{\tilde{a}_q \geq 0 \text{ for any } q\}$ and the constraint expressed by eq. (135) and produces the optimal value $\hat{\mathbf{a}}$ of the vector $\tilde{\mathbf{a}}$. Given $\hat{\mathbf{a}}$, the optimal values $\hat{\mathbf{w}}$ and $\hat{b}$ of $\tilde{\mathbf{w}}$ and $\tilde{b}$, respectively, are computed as (see eq. (136))

$$\hat{\mathbf{w}} = \sum_{q=0}^{N_t-1} \hat{a}_q\, t_q\, \mathbf{r}_q, \tag{138}$$

and

$$\hat{b} = N_{\mathcal{S}_M}^{-1} \sum_{q \in \mathcal{S}_M} \left(t_q - \sum_{k \in \mathcal{S}_M} \hat{a}_k\, t_k\, \mathbf{r}_q^T\, \mathbf{r}_k \right), \tag{139}$$

respectively, where $\mathcal{S}_M$ and $N_{\mathcal{S}_M}$ denote the set of support vectors and its cardinality, respectively. Given ($\hat{\mathbf{w}}$, $\hat{\mathbf{a}}$, $\hat{b}$), the classification of a new data point $(\mathbf{r}_q, t_q)$ (with $q > N_t - 1$) is accomplished on the basis of the sign of the quantity (see eq. (126))

$$y(\mathbf{r}_q, \hat{\mathbf{w}}) \triangleq \hat{\mathbf{w}}^T \, \mathbf{r}_q + \hat{b}, \tag{140}$$

that can be also expressed as (see eq. (138))

$$y(\mathbf{r}_q) = \sum_{k=0}^{N_t-1} \hat{a}_k \, t_k \, \mathbf{r}_q^T \, \mathbf{r}_k + \hat{b}. \tag{141}$$

As already mentioned above, this classification method is derived under the assumption that the set of feature vectors $\{\mathbf{r}_q\}$ is linearly separable. When this does not occur, a specific *kernel function*, denoted $\boldsymbol{\phi}(\cdot)$, can be used to map the vector $\mathbf{r}_q$ (78) into the new feature vector $\boldsymbol{\phi}(\mathbf{r}_q)$ for any q (e.g., see [71, Chap. 6]). The objective is transforming the available classification space into a one characterized by linear boundaries; in principle, the dimensionality of $\boldsymbol{\phi}(\cdot)$ may be different from D_r. Well known examples of the kernels employed with the SVM method are the polynomial, the Gaussian and the Laplace kernels. It is important to note that kernel selection is very critical, since its choice can significantly influence classification accuracy.

The second method we take into consideration is the so called *K nearest-neighbour* (K-NN) technique [62], that represents an example of non-parametric approach to the classification problem. In the case of binary classification, this method can be summarised as follows. The points of the training set $\mathcal{D}$ (77) are partitioned into two classes, denoted $\mathcal{C}_0$ and $\mathcal{C}_1$, where

$$\mathcal{C}_k \triangleq \{(\mathbf{r}_{q_k}, \, t_{q_k}); q_k = 0, 1, ..., N_k - 1\}, \tag{142}$$

with $k = 0$ and 1, and N_k denotes the number of points belonging to the k-th class, so that

$$\sum_{k=0}^{1} N_k = N_t. \tag{143}$$

Let us assume now that a new D_r-dimensional observation, denoted $\mathbf{r}_q$ (with $q > N_t - 1$) and called *query instance*, becomes available. The K-NN strategy classifies $\mathbf{r}_q$, i.e. assigns it to one of the two classes defined above, on the basis of the votes of its K *nearest neighbours* (i.e., of the K points of $\mathcal{D}$ closest to $\mathbf{r}_q$); here, K is an integer parameter, whose value is usually small and odd. The identification of the nearest neighbours unavoidably requires the computation of the distance of $\mathbf{r}_q$ from all the points of the set $\mathcal{D}$; if the Euclidean distance is employed, the distance of $\mathbf{r}_q$ from $\mathbf{r}_t \in \mathcal{D}$ is given by

$$d_q \triangleq ||\mathbf{r}_t - \mathbf{r}_q||, \tag{144}$$

with $t = 0, 1, ..., N_t - 1$. Given the set $\{d_q\}$, consisting of N_t distances, the nearest neighbours $\{\mathbf{r}_{\mathrm{nn},j}; j = 0, 1, .., K - 1\}$ are identified by searching for the K points of $\mathcal{D}$ that satisfy the inequality

$$d_q < V_q \tag{145}$$

where V_q is a fixed threshold, such that all the required K points are found. Then, if K_k denotes the number of nearest neighbours belonging to $\mathcal{C}_k$ (i.e., the number of *representatives* of $\mathcal{C}_k$), $\mathbf{r}_q$ is assigned to the class having the largest number of representatives, i.e. to $\mathcal{C}_0$ ($\mathcal{C}_1$) if $K_0 > K_1$ ($K_1 > K_0$).

It is worth pointing out that the parameter K controls the degree of smoothing, i.e. the size of the regions assigned to each class. In fact, a small value of K usually results in many small regions assigned to each class, whereas a large one leads to fewer larger regions [71, Par. 2.5.2]. Moreover, if $K = 1$ is selected, a *nearest-neighbour* classifier is obtained; in this case, if the dataset is quite large, it can be shown that the error rate of a K-NN classifier is never larger than twice the minimum achievable error rate of an optimal classifier[11], i.e. of a classifier having full knowledge of the pdf of the observations [72].

Multiple classification methods can be combined to improve the overall accuracy; this idea leads to the development of the so called *ensemble classifiers* [71, Ch. 14.2]. Specific examples of these classifiers are represented by the so called *bootstrap aggregating* (also known as *bagging* [73]) and *boosting* methods [71]. The first method can be employed when M predictions, denoted $\{y^{(m)}(\mathbf{r}_q); m = 0, 1, ..., M - 1\}$ and generated by M different classifiers (called *base classifiers*), are available; the output is computed as

$$Y_M \triangleq \frac{1}{M} \sum_{m=0}^{M-1} y^{(m)} \left(\mathbf{r}_q\right), \tag{146}$$

[11]The optimal classification strategy can be easily formulated on the basis of eq. (83) (see Paragraph III-B1).

i.e. as an average of all the above mentioned predictions and the predicted class is identified by the sign of this quantity; this reduces the impact of the error due to each single classifier M times. This method is really effective when the errors originating from distinct classifiers are uncorrelated; unluckily, in some cases, such errors may be significantly correlated. When this occurs, classification accuracy can be improved through boosting and, in particular, through the *adaptive boosting* method, also known as *AdaBoost* [74]. In fact, the AdaBoost technique can achieve good accuracy even if its M base classifiers do not perform well (say, their behaviour is only slightly better than random), i.e. they are *weak learners*. If a *binary classification problem* is considered, the training phase of this method evolves through M classification stages, each involving a distinct base classifier; moreover, this method is initialised assigning the same weight to all the observations, i.e. setting $\tilde{w}_q^{(0)} = 1/N_t$ for any q, where $\tilde{w}_q^{(0)}$ denotes the initial weight assigned to the q-th observation. The m-th stage (with $m = 0$, 1, ..., $M-1$) evolves through the following steps:

1) The m-th base classifier is trained to minimise the weighted error function

$$J^{(m)} \triangleq \sum_{q=0}^{N_t-1} \tilde{w}_q^{(m)} I(y^{(m)}(\mathbf{r}_q)), \tag{147}$$

where

$$I\left(y^{(m)}(\mathbf{r}_q)\right) \triangleq \begin{cases} 1 \text{ if } y^{(m)}(\mathbf{r}_q) \neq t_q \\ 0 \quad \text{otherwise} \end{cases} \tag{148}$$

and $\{\tilde{w}_q^{(m)}\}$ is a set of non negative weights such that

$$\sum_{q=0}^{N_t-1} \tilde{w}_q^{(m)} = 1. \tag{149}$$

2) The weighted measure of the *error rate*

$$\varepsilon^{(m)} \triangleq \frac{\sum_{q=0}^{N_t-1} \tilde{w}_q^{(m)} I\left(y^{(m)}(\mathbf{r}_q)\right)}{\sum_{q=0}^{N_t-1} \tilde{w}_q^{(m)}} \tag{150}$$

and the *weighting coefficient* (e.g., see [71, Par. 14.3, eq. (14.16)])

$$\alpha^{(m)} \triangleq \ln\left(\frac{1-\varepsilon^{(m)}}{\varepsilon^{(m)}}\right) \tag{151}$$

are computed.

3) The weight assigned to the q-th data point is updated on the basis of the recursive formula (e.g., see [71, Par. 14.3, eq. (14.18)])

$$\tilde{w}_q^{(m+1)} = \tilde{w}_q^{(m)} \exp\left(\alpha^{(m)} I\left(y^{(m)}(\mathbf{r}_q)\right)\right) \tag{152}$$

for any q.

These steps force the classifier employed in each stage to put more emphasis on those points that have been misclassified by previous classifiers. In fact, an higher error rate entails a larger increase of the weight assigned to the q-th observation (see eqs. (151) and (152)), provided that it has not been correctly classified (i.e., that $I(y^{(m)}(\mathbf{r}_q)) = 1$). The final prediction generated by the AdaBoost technique is

$$Y_{B_M}(\mathbf{r}_q) = \text{sign}\left(\sum_{m=0}^{M-1} \alpha^{(m)}\, y^{(m)}(\mathbf{r}_q)\right). \tag{153}$$

In assessing the accuracy of any classification method, N-fold cross validation can be used when the size of the available dataset is not so large. This consists in:

a) randomly partitioning the whole available dataset in N blocks;

b) assessing the classification accuracy on the n-th block (taken as test set) after that the considered method has been trained on the basis of the remaining $(N-1)$ blocks (with $n = 0$, 1, ..., $N-1$).

At the end of this procedure, N distinct accuracies are available; the final score is expressed by their average.

Let us focus now on a specific application of the SVM and K-NN techniques to an FCMW radar system equipped with the antenna array shown in Fig. 4-b) (and characterized by $d = \lambda/4$) and operating in the presence of *at most a single point target*. In the q-th trial, the set $\{\mathbf{r}_{0,q}, \mathbf{r}_{1,q}, \mathbf{r}_{2,q}, \mathbf{r}_{3,q}\}$, consisting of four N-dimensional noisy vectors, each associated with one of the $N_V = 4$ virtual receive channels, is available for any q (see eq.

(51)). The dimensionality reduction technique illustrated in Paragraph III-A is applied to this set in order to extract the 4-*dimensional* (4D) feature vector

$$\begin{aligned} \mathbf{R}_q &= \left[R_0^{(q)}, R_1^{(q)}, R_2^{(q)}, R_3^{(q)}\right]^T \\ &\triangleq \left[\left|X_0^{(q)}\right|, \left|X_1^{(q)}\right|, \left|X_2^{(q)}\right|, \left|X_3^{(q)}\right|\right]^T; \end{aligned} \tag{154}$$

here,

$$X_v^{(q)} = M_r \; X_{v,\hat{l}}^{(q)} \tag{155}$$

and $X_{v,\hat{l}}^{(q)}$ is computed on the basis of eqs. (54), (58) and (61), i.e. by sampling the spectrum $\bar{X}_v^{(q)}(f)$ (49) of the zero-padded sample sequence acquired on the v-th virtual antenna at the target frequency $\hat{f}_m$ (60) (with $v = 0$, 1, 2 and 3). The target detection strategy we adopt in our radar system is different from the one illustrated in Paragraph III-A and based on the computation of the average power

$$P_q \triangleq N_V^{-1} \sum_{v=0}^{N_V-1} |X_v^{(q)}|^2, \tag{156}$$

and on its comparison with a threshold (see eqs. (57) and (59)). This choice is motivated by the fact that, the amplitude $a_v^{(q)}$ of the sinusoid observed on the v-th virtual channel and associated with the detected point target is assumed to depend on the antenna index[12] v; the last assumption allows us to account for: a) the dependence of the target reflectivity on the direction of observation; b) the differences in the amplifications introduced by distinct receive chains of the employed MIMO radar. In fact, in the considered radar system, a target is detected if the inequality

$$\max_{v \in \{0,1,2,3\}} \left|X_v^{(q)}\right|^2 > P_{da}, \tag{157}$$

holds, i.e. if $|X_v^{(q)}|^2 > P_{da}$ for at least a single value of v. This strategy outperforms the one based on the average power P_q (156) in terms of missed detection probability; however, the price to be potentially paid for this is an increase of the false alarm probability, i.e. of the probability of erroneously detecting the presence of target.

In our experiment, the training set

$$\mathcal{D} \triangleq \{(\mathbf{R}_q, \, t_q); q = 0, 1, ..., N_t - 1\}, \tag{158}$$

referring $N_t = 100$ independent trials, has been synthetically generated. Half of its data are associated with the detection of a *real* target, the remaining half with the detection of a *false* target; for this reason, the vector $\mathbf{R}_q$ (154) is labelled by $t_q = 1$ (-1) in the presence of a *real* (*false*) target. Moreover, the following assumptions have been made in generating the q-th observation of the training set $\mathcal{D}$ and the test set $\mathcal{D}_{ts}$:

a) The amplitude $a_v^{(q)}$ of the sinusoid observed on the v-th antenna in the presence of a real target is uniformly distributed over the interval $[0, 1]$ V;

b) The random variable $a_v^{(q)}$ is independent of $a_u^{(p)}$ for any $u \neq v$ and/or $p \neq q$.

c) The range R_q and the azimuth ϕ_q of the target (if present) are uniformly distributed over the intervals $[R_m \, , \, R_M] = [1.0$ m, 5.0 m$]$ and $[\phi_m \, , \, \phi_M] = [-60°, \, 60°]$, respectively, for any q.

The values selected for most of the parameters of the considered radar system are equal to those listed in the examples of Paragraphs III-B2 and III-B3, the only differences being represented by the fact that:

a) the standard deviation of the noise affecting the received signal samples is $\sigma_w = 1.0$ V (see eq. (7));

b) the threshold $P_{da} = 0.3$ V^{2}Hz^{-1} is employed by the detection algorithm based on eq. (157).

The dataset $\mathcal{D}$ (158) has been employed to train the linear SVM, K-NN and Adaboost techniques; $K = 4$ and $M = 100$ has been selected for the second classifier and the third one, respectively. Moreover, the weak learner employed in the m-th step of the Adaboost technique consists in comparing one of the components of the vector $\mathbf{R}_q$ (154) with a threshold[13]. More specifically, the classification criterion adopted by each weak learner can be expressed as

$$R_v^{(q)} \underset{t_q=-1}{\overset{t_q=1}{\gtrless}} P_v \tag{159}$$

[12]For this reason, the assumption we made in writing eq. (7) does not hold any more.

[13]This simple classifier can be interpreted as a form of a decision tree known as *decision stump* and characterized by a single node (e.g., see [71, Ch. 14.3-14.4]

where the index v is randomly selected in the set $\{0, 1, 2, 3\}$ and $P_v \sim \mathcal{U}(\min_q(R_v^{(q)}), \max_q(R_v^{(q)}))$ is the decision threshold associated with the v-th feature $R_v^{(q)}$ acquired in the q-th trial. Note that the classification criterion (159) leads to partitioning the observation space into two regions, separated by an hyperplane (perpendicular to one of the reference axes).

In this case, the aim of the three classifiers is discriminating between the presence of a real target and that of a false target any time a target is detected; for this reason, they are exploited to reduce the false alarm probability. Some numerical results are shown in Figs. 8, 9 and 10, that refer to the SVM, to the K-NN and to the Adaboost techniques, respectively; in all these figures, the set of points[14] $\{(R_0^{(q)}, R_1^{(q)})\}$ extracted from the dataset $\mathcal{D}$ (158) are represented on a Cartesian plane and are identified by a green (blue) circle if associated with a false (real) target. These results deserve the following comments:

1) SVM training leads to generating the linear decision boundary shown in Fig. 8; in this figure, a new observation is classified as a false target, since it falls in the lower decision region.

2) The K-NN method classifies the new observation shown in Fig. 9 as a false target, since class $\mathcal{C}_1$ is the one having the largest number of representatives contained in the black circumference (having radius equal to $V_q = 0.07$ V and centered at the new observation).

3) Adaboost training leads to generating the decision boundary shown in Fig. 10. In the same figure, the *critical points* of the base classifiers (i.e., their misclassified points) are also shown; as it can be easily inferred from eq. (152), their weights of these points tend to increase with iterations. In the same figure, a new observation is classified as a false target, since it falls in the lower decision region.

In the considered scenario, our computer simulations have evidenced that the accuracy achieved by the considered classification techniques is around 90%, assuming N-fold cross validation with $N = 5$; in particular, the accuracies of SVM, K-NN and Adaboost are 91%, 89% and 93%, respectively.

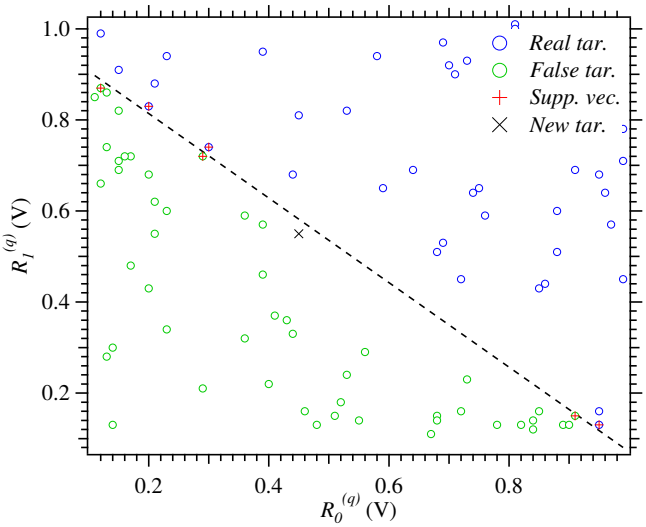


Figure 8: Representation of the decision mechanism employed by a linear SVM classifier. The points of the training set corresponding to false (real) targets are identified by the green (blue) circles. The decision boundary of the SVM is represented by a dashed line, whereas the red crosses identify support vectors. A new observation, identified by a black cross, is classified as a false target, since it falls in the lower half plane delimited by the decision boundary.

The binary classification methods illustrated above can be also exploited to develop solutions to multi-class problems; in fact, in general, any problem of this type can be represented as a sequence of binary classification problems [75]. This approach is exemplified in Paragraph VI-A and, in more detail, in Paragraph VIII-A, where its application to the classification of three human activities is illustrated. In particular, in Paragraph VIII-A, it is shown how a specific solution to this problem can be devised by exploiting *pairwise classification* (also known as *round-robin* class binarization). If K denotes the overall number of classes, this classification method is based on a) combining $L = K(K-1)/2$ binary classifiers (called *base* learners) and b) using the so called *one-versus-one* coding scheme. In this case, each binary classifier is trained assuming one class as positive, another class as negative (the labels associated with the q-th observation are $t_q = 1$ and $t_q = -1$ for the first

[14]Note that the observations of the dataset belong to a 4D space in this case; for this reason, all their components cannot be represented in the same figure.

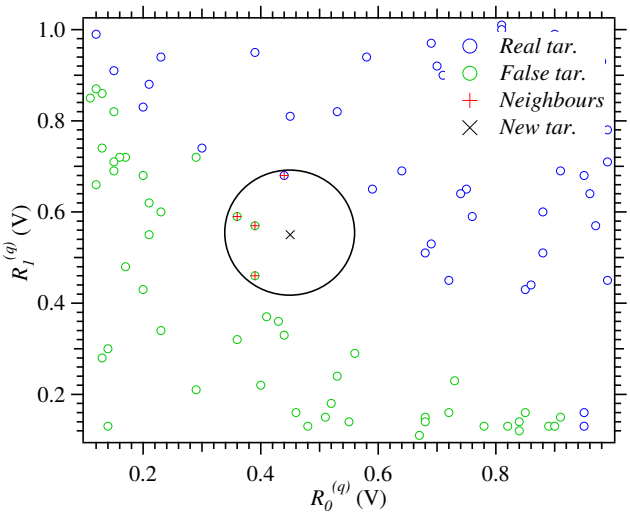


Figure 9: Representation of the decision mechanism employed by a K-NN classifier (with $K = 4$). The points of the training set corresponding to false (real) targets are identified by the green (blue) circles. A new observation, identified by a black cross, is classified as a false target, since class $\mathcal{C}_1$ is the one having the largest number of representatives contained in the black circle.

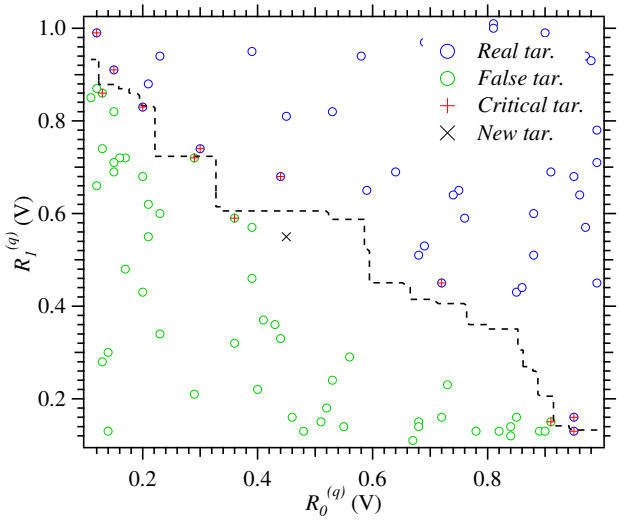


Figure 10: Representation of the decision mechanism employed by the Adaboost classifier. The points of the training set corresponding to false (real) targets are identified by the green (blue) circles. The decision boundary of the Adaboost is represented by a dashed line, whereas the red crosses identify critical targets. A new observation, identified by a black cross, is classified as a false target, since it falls in the lower region delimited by the decision boundary.

class and the second one, respectively), and ignoring all the other classes (the label associated with this case is $t_q = 0$). When a new observation is available, it is processed by each binary classifier, so that all its possible assignments to every class pair are taken into consideration. This procedure leads to generating a codeword of size L for each class; the l-th element of this codeword can take on the values ± 1 or 0 on the basis of the class membership established by the l-th binary learner (with $l = 0, 1, ..., L-1$). The K codewords produced by the L learners in response to the q-th observation represent the rows of the $K \times L$ *design matrix* $\mathbf{T}_q = [t_{k,l}^{(q)}]$; note that the presence of a '0' on the k-th row of the l-th column for any q means that all the observations associated with the k-th class are ignored by the l-th classifier. The class predicted for the q-th observation is the one minimizing the average of the binary losses over the L different binary learners [76]; in practice, the

value of the class index for the q-th observation is computed as

$$\hat{k}_q = \arg \min_{k \in \{0,1,...,K-1\}} \frac{\sum_{l=0}^{L-1} |t_{k,l}^{(q)}| \, g\left(t_{k,l}^{(q)}, y_{q,l}\right)}{\sum_{l=0}^{L-1} |t_{k,l}^{(q)}|}, \tag{160}$$

where $y_{q,l}$ is the *score*[15] assigned by the l-th binary learner to the considered observation and

$$g\left(t_{k,l}^{(q)}, y_{q,l}\right) \triangleq \frac{1}{2} \exp\left(-t_{k,l}^{(q)} y_{q,l}\right) \tag{161}$$

is the *binary loss* function.

C. Unsupervised learning

Unsupervised learning is less well defined than its supervised counterpart, since it deals with learning some specific properties of the mechanism on which the generation of the considered set of observations is based. Unlike supervised methods, unsupervised learning operates over *unlabelled* datasets. In the following, we assume that:

1) Learning is based on the dataset

$$\mathcal{D} \triangleq \{\mathbf{r}_q; q = 0, 1, ..., N_t - 1\}, \tag{162}$$

that consists of N_t i.i.d. *unlabelled* D_r-dimensional observations;

2) All the available observations are realizations of the same random variable $\mathbf{r}$, characterized by its *unknown* pdf $f(\mathbf{r})$.

The goal of unsupervised methods is to learn some useful properties of the pdf $f(\mathbf{r})$. It is important to keep in mind that the D_r elements which the random vector $\mathbf{r}$ is made of can be highly correlated. These mutual dependencies are often modelled by introducing a new vector, denoted $\mathbf{z}$ and collecting the so called *latent* or *hidden variables*. This approach allows to model the dependencies between the elements of the observations indirectly, i.e. through the direct dependencies between such elements and the hidden vector. The relationship between the vectors $\mathbf{z}$ and $\mathbf{r}$ can be modelled in different ways. This results in various different models that can be adopted in unsupervised learning; further details about this issue can be found in ref. [77].

In the remaining part of this paragraph, we first list the typical unsupervised problems tackled in the field of MIMO radar systems. Then, we describe two specific unsupervised methods and illustrate their application to specific problems in that field.

1) Unsupervised problems: Unsupervised learning methods can be exploited to solve the following four relevant technical problems:

a) *Clustering* - Data clustering consists in partitioning the dataset $\mathcal{D}$ (162) in a number of groups such that data points in the same group are dissimilar from the data points belonging to all the other groups. In clustering problems, an hidden random variable, called *class variable*, is usually added to all the elements of the dataset; this variable describes the cluster membership for every observation of the dataset. In the last years, significant attention has been paid to the use of clustering methods in automotive radar systems, since distinct clusters can be related to different types of targets, like pedestrians, cars or obstacles. A description of two clustering methods employed in the above mentioned field is provided in Paragraph VI-D.

b) *Dimensionality reduction* - This aims at generating a reduced dimensionality representation of the observations. Such a representation eases the visualization and interpretation of the dataset, and the identification of specific patterns in it. A well known technique for dimensionality reduction is the *principal component analysis* (PCA); its description is provided in Paragraph VI-C, whereas its application to a dataset referring to a specific MIMO radar system is illustrated in Paragraph III-C2.

c) *Feature extraction* - This consists in deriving a vector-valued function, denoted $\mathbf{g}(\cdot)$ and such that $\mathbf{g}(\mathbf{r})$ represents a useful and lower-dimensional representation of the feature vector $\mathbf{r}$; the vector $\mathbf{g}(\mathbf{r})$ can be used as an input to a supervised learning method. A well known method for synthetizing the function $\mathbf{g}(\cdot)$ is represented by the *autoencoder*, as illustrated in Paragraph IV-D3. A simple method for feature extraction in a MIMO radar system has been described in Paragraph III-A; other techniques are illustrated in Paragraphs VI-A and VI-D, where their use of radar in human motion characterization and in autonomous driving, respectively, is considered.

d) *Generation of new samples* - This aims at producing new samples of a random vector $\mathbf{r}$ in a way that these are approximately distributed according to its true pdf $f(\mathbf{r})$. Methods for generating new samples can be exploited to de-noise data and for interference mitigation in autonomous driving applications, as illustrated in Paragraph VI-D.

[15]This quantity can be computed on the basis of eq. (141) (eq. (153)) if the SVM (Adaboost) method is used.

2) Specific unsupervised methods: In this paragraph we focus on two specific unsupervised methods, namely the PCA technique for dimensionality reduction [78] and the *K-means* algorithm for data clustering [79].

The PCA method is employed to project the dataset $\mathcal{D}$ (162) onto a new space, called *principal subspace* and having dimensionality $D_r' < D_r$; in doing so, the variance of the projected data is maximised, in order to retain the most relevant variations characterizing the original dataset. This method can be easily understood by illustrating its application to the case in which $D_r = 4$, $D_r' = 1$ and $\mathbf{r}_q = \mathbf{R}_q$, where the 4D vector $\mathbf{R}_q$ is expressed by eq. (154). In this case, the 4D observation $\mathbf{R}_q$ is projected onto the scalar

$$R_q^{'} \triangleq \mathbf{u}_0^T \, \mathbf{R}_q, \tag{163}$$

where $\mathbf{u}_0$ is a 4D unit vector [71]. If we define the data covariance matrix

$$\mathbf{\Delta} \triangleq \frac{1}{N_t} \sum_{q=0}^{N_t-1} \left(\mathbf{R}_q - \bar{\mathbf{R}}\right) \left(\mathbf{R}_q - \bar{\mathbf{R}}\right)^T, \tag{164}$$

where

$$\bar{\mathbf{R}} \triangleq \frac{1}{N_t} \sum_{q=0}^{N_t-1} \mathbf{R}_q, \tag{165}$$

is the data mean, the variance

$$\sigma_R^2 = \mathbf{u}_0^T \, \mathbf{\Delta} \, \mathbf{u}_0 \tag{166}$$

of the projected dataset is maximized if

$$\mathbf{u}_0^T \mathbf{\Delta} \, \mathbf{u}_0 = \lambda_0, \tag{167}$$

where λ_0 is the largest eigenvalue of the matrix $\mathbf{\Delta}$ (164) and $\mathbf{u}_0$ (that represents the *first principal component*) is the associated eigenvector.

In general, if a D_r'-dimensional projection space is considered, the principal components are represented by D_r' eigenvectors $\{\mathbf{u}_l; \, l = 0, \, 1, \, ..., \, D_r' - 1\}$ of the data covariance matrix $\mathbf{\Delta}$; these eigenvectors are associated with its D_r' largest eigenvalues $\{\lambda_l; l = 0, \, 1, \, ..., \, D_r' - 1\}$ and are chosen to be *orthonormal*. The quality of the resulting transformation can be assessed by evaluating the *distortion measure* (e.g., see [71, Par. 12.1.2, eq. (12.18)])

$$J \triangleq \sum_{l=D_r'}^{D_r-1} \lambda_l, \tag{168}$$

i.e., the sum of the eigenvalues associated with the eigenvectors that are orthogonal to the principal subspace; the smaller is the value taken on by this parameter, the better is the original dataset approximation.

In our experiment, the PCA method has been applied to extract a 2D dataset from the 4D dataset which Figs. 8-10 refer to (see Paragraph III-B4). The 2D points of the new dataset, denoted $\mathcal{D}'$, are represented in the *principal component biplot*[16] shown in Fig. 11. In this figure, the axes of the Cartesian plane are associated with the principal components, whereas the vector $\mathbf{w}_i$, represented by an oriented segment, allows to quantify, through its amplitude and orientation, the weight of the contribution provided by the i-th component of the original feature vectors (i.e., of the set $\{\mathbf{R}_q\}$; see eq. (154)) to the principal components (with $i = 0, \, 1, \, 2$ and 3). From Fig. 11 it is easily inferred that:

1) The weights of the contributions due to $R_2^{(q)}$ and $R_3^{(q)}$ are similar and are about half of those provided by $R_0^{(q)}$ and $R_1^{(q)}$.
2) The new 2D observations referring to real (false) targets are spread over the right (left) half plane of the Cartesian plane.

The *K-means* method allows to partition the available dataset $\mathcal{D}$ into K clusters, each collecting the samples whose mutual distances are small with respect to the distances from the points outside the cluster itself. In practice, if the *center* of the k-th cluster is denoted $\boldsymbol{\mu}_k$ (with $k = 0, \, 1, \, ..., \, K-1$), the K-means method assigns the q-th data point $\mathbf{r}_q$ to the cluster whose center is closest to $\mathbf{r}_q$. This strategy can be formalised as the one minimizing the so-called *distortion measure*

$$V \triangleq \sum_{q=0}^{N_t-1} \sum_{k=0}^{K-1} p_{q,k} ||\mathbf{r}_q - \boldsymbol{\mu}_k||^2, \tag{169}$$

with respect to the variables $\{p_{q,k}\}$ and the vectors $\{\boldsymbol{\mu}_k\}$; here, $p_{q,k}$ is a binary indicator variable implementing the 1-of-K coding scheme, i.e. such that $p_{q,k} = 1$ ($p_{q,k} = 0$) if $\mathbf{r}_q$ is (is not) assigned to the k-th cluster. The

[16] A detailed description of how a bi-plot is generated can be found in ref. [78, Sect. 5.3]

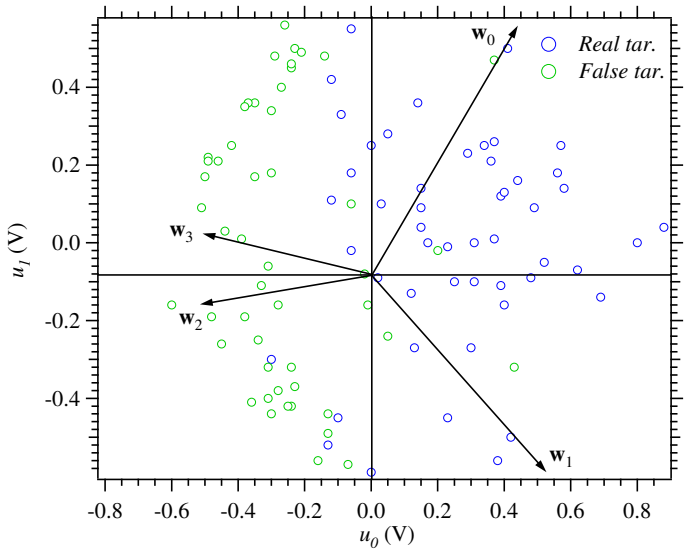


Figure 11: Biplot of the dataset $\mathcal{D}'$ generated by the PCA technique. The points of the reduced dataset corresponding to false (real) targets are identified by the green (blue) circles. The four oriented segments allow to quantify the contribution provided by each of the four components of the original feature vector to the two principal components.

problem of minimizing the function V (169) is solved by means of an iterative procedure, whose iterations consist of two steps. In the first step, known as *expectation*, the metric V is minimized with respect to each of the variables $\{p_{q,k}\}$, keeping the centers $\{\boldsymbol{\mu}_k\}$ fixed; on the contrary, in the second step, called *maximization*, the same metric is minimized with respect to the vectors $\{\boldsymbol{\mu}_k\}$, keeping the variables $\{p_{q,k}\}$ fixed. More specifically, in the first step, the values of the variables $\{p_{q,k}\}$ employed are computed as

$$p_{q,k} \triangleq \begin{cases} 1 \text{ if } \quad k = \arg\min_j ||\mathbf{r}_q - \boldsymbol{\mu}_j||^2 \\ 0 \text{ otherwise} \end{cases} \tag{170}$$

for any q (in other words, the q-th data point is assigned to the cluster whose center is closest to it). Then, in the second step, the center of the k-th cluster is evaluated as

$$\boldsymbol{\mu}_k = \frac{\sum_{q=0}^{N_t-1} p_{q,k}\mathbf{r}_q}{\sum_{q=0}^{N_t-1} p_{q,k}}, \tag{171}$$

with $k = 0, 1, ..., K-1$. It is important to point out that:

1) In principle, the initial values of the cluster centers can be arbitrarily chosen. In this case, however, the algorithm may require several iterations to reach convergence. A better initialization procedure consists in choosing the initial centers in a random fashion.
2) The sum appearing in the denominator of the RHS of the eq. (171) gives the overall number of points assigned to the k-th cluster; consequently, the cluster center evaluated on the basis of the same equation represents the mean of all the data points $\mathbf{r}_q$ assigned to the k-th cluster.
3) Iterations are stopped when is no further change in the assignments of the data points to the K clusters or their overall number has reached a fixed threshold.

Let us analyse now an application of the K-means technique to the dataset $\mathcal{D}$ (162), where $N_t = 100$,

$$\mathbf{r}_q \triangleq \left[\hat{R}_q\,,\,\hat{\phi}_q\right]^T, \tag{172}$$

and $\hat{R}_q$ and $\hat{\phi}_q$ represent the estimates of the range and of the azimuth, respectively, of the single point target observed in the q-th trial; these estimates are generated by the algorithm illustrated in Paragraph III-A and employed in a FCMW radar system equipped with the antenna array illustrated in Fig. 4-b) ($d = \lambda/4$ is assumed). Moreover, in generating the q-th observation of the dataset $\mathcal{D}$ (162), the following assumptions have been made:

a) The amplitude $a_v^{(q)}$ of the sinusoid observed on the v-th virtual antenna is uniformly distributed over the interval $[0.3, 1.0]$ V.

b) The random variable $a_v^{(q)}$ is independent of $a_u^{(p)}$ for any $u \neq v$ and/or $p \neq q$.

c) The overall number of time-domain samples (N) acquired from each of the four RX antennas is equal to 512 and the standard deviation σ_w of the noise affecting them is equal to 1.0 V (see eq. (7)).

d) The oversampling factor $M_r = 4$ and the threshold $P_{th} = 0.5$ V^{2}Hz^{-1} are employed by the detection algorithm based on eqs. (58)-(59).

e) The range R_q of the target detected in the q-th trial is uniformly distributed over the interval $[R_m\,,\,R_M] = [1.0$ m, 9.0 m$]$, whereas its azimuth is randomly selected in the set of relative integers ranging from ϕ_m to ϕ_M, with $\phi_M = -\phi_m = 45°$.

f) The parameters of the employed radar system take on the same values as those selected for the example illustrated for the SVM and K-NN methods in Paragraph III-B4.

In this case, the K-means algorithm is employed to group the detected targets in three different clusters (consequently, $K = 3$ is selected) on the basis of their azimuth only; the points of the first (third) cluster are characterized by $\phi_q < -15°$ ($\phi_q > 15°$), whereas those of the second one by $|\phi_q| \leq 15°$.

The observations collected in the synthetically generated dataset and their partitioning into the clusters generated by the K-means technique are shown in Fig. 12, where circles of different colours are used to identify targets assigned to distinct classes. From these results it is easily inferred that:

1. all the points are correctly classified on the basis of their azimuth, even if an unlabelled dataset is used;

2. each of the centroids is located in the middle of the corresponding cluster and its position is influenced by the distribution of the detected targets along the range dimension.

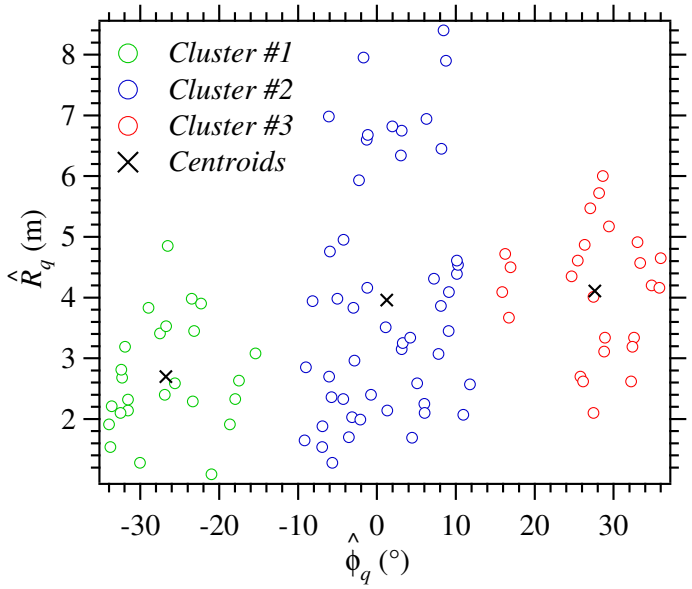


Figure 12: Representation of three clusters generated by the K-means algorithm. The green (red) circles refer to the targets detected on the left (right) of the considered radar system, whereas the blue circles to the targets detected in front of it. The black crosses identify the centroids of the clusters.

IV. Deep Learning Techniques for Colocated MIMO Radar Systems

In this section, we first analyse some relevant differences between ML and DL techniques. Then, we introduce readers to deep neural networks by illustrating their architecture, their training and a specific application to an FMCW radar system. Finally, we briefly illustrate a few fundamental DL methods employed in the field of MIMO radar systems.

A. Relevant differences between ML and DL techniques

Machine learning techniques allow to achieve satisfying accuracy in various applications at the price of a reasonable computational complexity. Nevertheless, in pattern recognition problems, their capability is often limited by the features selected to learn common patterns and to detect them; in fact, in these cases, devising a transformation able to extract a suitable internal representation from the observed raw data requires good expertise and engineering skills [80]. A revolutionary data-driven approach to feature extraction is offered by DL methods. Despite the significant computational complexity of these methods, in recent times their implementation has

become possible thanks to the availability of low-cost powerful *graphic processing units* (GPUs), which make the exploitation of their inner parallelism possible.

Deep learning solves the problem of feature extraction by adopting a multilayer representation of raw data. This fundamental principle is exemplified by a *feedforward deep network*, also known as *multilayer perceptron* (MLP); such a network is able to represent a complicated mathematical function by composing multiple simpler functions, i.e. multiple *layers*. Generally speaking, a MLP consists of three different types of layers: an input layer, multiple hidden layers with learnable weights and an output layer. Its architecture can be represented through a *directed acyclic graph* (DAG), whose structure is exemplified in Fig. 13, that refers to the specific case of a fully connected MLP containing a single inner layer. The basic building block of each layer is the so called *neuron*. In general, the output $z_j^{(k)}$ generated by the j-th neuron of the k-th layer can be expressed as

$$z_j^{(k)} = h\left(a_j^{(k)}\right), \tag{173}$$

with $j = 1, 2, ..., M_k$ and $k = 1, 2, ..., K$; here, M_k denotes the overall number of neurons in the k-th layer, K denotes the overall number of layers, $h(\cdot)$ is a differentiable non-linear function (i.e., a sigmoid function, an hyperbolic tangent or rectifier linear unit) and the quantity

$$a_j^{(k)} \triangleq \sum_{i=1}^{M_{k-1}} w_{j,i}^{(k)} z_i^{(k-1)} + w_{j,0}^{(k)}, \tag{174}$$

known as *activation* function, is a linear combination of the neuron inputs $\{z_i^{(k-1)};\ i = 1, 2, ..., M_{k-1}\}$ (whose *learnable weights* are the M_k parameters $\{w_{j,i}^{(k)};\ i = 1, 2, ..., M_{k-1}\}$) and the *bias* $w_{j,0}^{(k)}$. The outputs of the neurons of the k-th layer are collected in the vector

$$\mathbf{z}^{(k)} \triangleq \left[z_1^{(k)}, z_2^{(k)}, ..., z_{M_k}^{(k)}\right], \tag{175}$$

that feeds the successive hidden layer. The input layer is fed by the D_x-dimensional input vector

$$\mathbf{x} \triangleq [x_1, x_2, ..., x_{D_x}], \tag{176}$$

whereas the output layer generates the D_y-dimensional output vector

$$\mathbf{y} \triangleq [y_1, y_2, ..., y_{D_y}], \tag{177}$$

on the basis of eqs. (173) and (174).

It is important to mention that: a) the learnable weights of the hidden layers can be interpreted as an encoded representation of the inputs; b) unlike ML methods, where a number of manually extracted features are chosen a priori, the considered neural network automatically extracts features through the use of non linear functions.

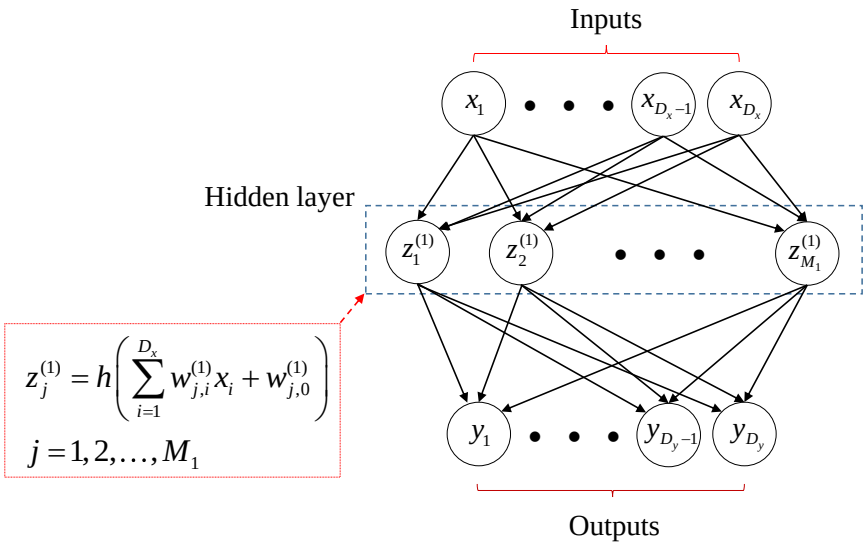


Figure 13: Directed acyclic graph describing the architecture of a fully-connected neural network. Variables are represented by circles (i.e., by nodes), whereas weights by the links between nodes. A single inner layer (i.e., $K = 1$) is assumed for simplicity.

B. Training of a deep neural network

Training a deep neural network is an art [77]. Its objective is the same as that already illustrated for ML methods, i.e. the minimization of a *loss* or an *error function* (see Paragraph III-B). However, in a feedforward neural network, this result is achieved by using a local message passing scheme, according to which the internal representations of each neuron are sent, alternately, forward and backward along the graph representing the network itself (e.g., see [71, Par. 5.3]). This scheme, known as *back-propagation* [81], operates as follows (batch processing is assumed here). For each pattern of the training set, the activations of the hidden and output layers of the considered network are computed through successive applications of eqs. (174) and (173), respectively; this process is known as *forward propagation*, since it proceeds from the input to the output of the network. The back-propagation algorithm, instead, allows to compute the gradient of the selected *error function*, denoted E and corresponding to the *loss function* defined for ML methods (see eq. (80)), with respect to the weights appearing in each layer. The derivative of the error function E with respect to the weight $w_{j,i}^{(k)}$, referring to the i-th input of the j-th neuron in the k-th layer of the network, can be expressed as

$$\frac{\partial E}{\partial w_{j,i}^{(k)}} = \sum_q \frac{\partial E_q}{\partial w_{j,i}^{(k)}}, \tag{178}$$

with $j = 1, 2, ..., M_k$, $i = 1, 2, ..., M_{k-1}$ and $k = 1, 2, ..., K$; here, E_q represents the error associated with the q-th observation. Based on the chain rule, the partial derivative appearing in the RHS of eq. (178) can be evaluated as

$$\frac{\partial E_q}{\partial w_{j,i}^{(k)}} = \sigma_j^{(k)} \, z_i^{(k)}, \tag{179}$$

where $\sigma_j^{(k)} \triangleq \partial E_q / \partial a_j^{(k)}$, $z_i^{(k)} \triangleq \partial a_j^{(k)} / \partial w_{j,i}^{(k)}$ and $a_j^{(k)}$ is defined by eq. (174). Consequently, eq. (178) can be put in the form

$$\frac{\partial E}{\partial w_{j,i}^{(k)}} = \sum_q \sigma_j^{(k)} \, z_i^{(k)}. \tag{180}$$

The quantity $\sigma_j^{(k)}$ appearing in the last formula can be evaluated as follows. First, the quantity

$$\sigma_l^{(K)} \triangleq y_l - t_l \tag{181}$$

is computed for the l-th unit of the output layer, where t_l denotes its target. Then, the backpropagation formula

$$\sigma_j^{(k)} = h' \left(a_j^{(k)} \right) \sum_l w_{l,j}^{(k+1)} \, \sigma_l^{(k+1)}. \tag{182}$$

is applied for $k = K-1, K-2, ..., 1$ and, given k, for $j = 1, 2, ..., M_k$; here, $h'(\cdot)$ denotes the first derivative of the function $h(\cdot)$ appearing in eq. (173). This allows us to recursively compute all the quantities $\{\sigma_l^{(k)}\}$ on the basis of the similar quantities $\{\sigma_l^{(k+1)}\}$ made available by all the units appearing in the $(k+1)$-th layer of the network.

It is worth noting that: a) the computational complexity of the network depends on the number of neurons in each hidden layer, since this determines the number of parameters to be tuned in the network; b) overfitting may be observed in the presence of a larger number of neurons. The last problem can be mitigated by including a *regularization term* in the considered error function (a similar strategy has been also proposed for ML methods; see eq. (107) in Paragraph III-B2). An alternative to this approach is represented by the so called *early stopping* procedure, that consists in stopping network training when the error over a given validation dataset[17] is minimised.

C. A specific application

Let us focus now on a neural network having the architecture illustrated in Fig. 13 and analyse its possible use in an FMCW radar system equipped with the antenna array shown in Fig. 4-b) ($d = \lambda/4$ is assumed). In our experiment, the overall synthetically generated dataset includes $\hat{N}_t = 2500$ observations, all acquired in the presence of a single point target, whose range R_q and the azimuth ϕ_q are uniformly distributed over the intervals $[R_m, R_M] = [1 \text{ m}, 7 \text{ m}]$ and $[\phi_m, \phi_M] = [-60°, 60°]$, respectively, for any q. Moreover, the values selected for the parameters of the employed radar system are equal to those listed in the example of Paragraphs

[17]The *validation dataset* is a set of data on which the performance of the considered network is evaluated during its training.

III-B2 and III-B3; the only difference is represented by the standard deviation of the noise affecting the received signal samples, that is $\sigma_w = \sqrt{2}/2$ V. The q-th observation and the associated label are[18]

$$\begin{aligned} \mathbf{r}_q &\triangleq [r_{q,0}, r_{q,1}, r_{q,2}, r_{q,3}, r_{q,4}]^T \\ &= \left[\hat{\psi}_{0,q}, \hat{\psi}_{1,q}, \hat{\psi}_{2,q}, \hat{\psi}_{3,q}, \hat{f}_q\right]^T \end{aligned} \tag{183}$$

and

$$\mathbf{t}_q \triangleq [t_{q,0}, t_{q,1}]^T = [R_q, \phi_q]^T, \tag{184}$$

respectively; here, $\hat{\psi}_{v,q} = \angle \hat{A}_{v,q}$ (with $v = 0$, 1, 2 and 3) and $\hat{A}_{v,q}$ is the complex amplitude measured on the v-th virtual element at the frequency $\hat{f}_q$ (60) (see eqs. (61) and (68)).

The aim of the neural network is predicting the position of the target (i.e., its azimuth and range) on the basis of a new observation. In this case, the network has 5 inputs two outputs, since $x_j = r_{q,j}$ (with $j = 0$, 1, ..., 4) and $y_k = t_{q,k}$ (with $k = 0$, 1). Moreover, a single hidden layer consisting of $M_1 = 10$ neurons is used; each of these neurons is connected to all the available inputs and employs the hyperbolic tangent transfer function

$$h(x) \triangleq \frac{\exp(2x) - 1}{\exp(2x) + 1} \tag{185}$$

in the evaluation of its output on the basis of eqs. (173)-(174). The predictions of the target range and azimuth are computed by the output layer, that contains two neurons.

The *scaled conjugate gradient* method [82] has been employed to train the network described above. The size of the training set $\mathcal{D}$ is $N_t = 2225$, since 85% of the overall dataset has been exploited for network training; the remaining part $\mathcal{D}_{ts}$ of the dataset, whose size is $\bar{N}_t = 375$, has been used as a test set. Our simulation results have evidenced that the adopted network is able to accurately predict the position of a new target; in fact, the RMSEs evaluated for the range and the azimuth on the set $\mathcal{D}_{ts}$ are approximately equal to 4 cm and to $0.2°$, respectively. Finally, it is worth noting that:

a) The use of the network described above does not require a specific expertise.

b) Unlike the regression methods illustrated in Paragraph III-B, the employed network is able to predict both the azimuth and the range of a single point target; however, a by far larger dataset is used for its training.

In general, the main drawback of DL methods is represented by the size of the dataset, which is usually much larger than that needed by ML techniques; this results in a significant increase in the computational effort of the required training.

D. Specific methods

In this paragraph, we focus on specific deep learning methods, namely autoencoders, convolutional neural networks, convolutional autoencoders, recurrent neural networks and generative adversarial networks. Each method is briefly described and some considerations on its use in the field of MIMO radars are made.

1) Autoencoders: An *autoencoder* (AE) is a neural network that, similarly as the PCA technique, is able to perform *dimensionality reduction* by learning an efficient representation of its input data in an unsupervised fashion. Since the goal of an AE is to approximate the identity function without learning it exactly, its D_y-dimensional output vector (177) can be expressed as

$$\mathbf{y} = \mathbf{h}_w(\mathbf{x}) \approx \mathbf{x}, \tag{186}$$

where $\mathbf{h}_w(\cdot)$ represents the transformation performed by the network on its D_x-dimensional input vector $\mathbf{x}$.

The architecture of an *under-complete* AE based on a symmetric encoding-decoding structure is illustrated in Fig. 14 [83]. If we consider the encoder side, the number of units contained in each hidden layer[19] decreases as we move from the network input to the output of that side; this is due to the fact that the network tries to learn a compressed version of the input data. On the other hand, the decoder has the goal to reconstruct, as faithfully as possible, the data vector $\mathbf{x}$ available at the AE input, starting from its compressed representation. For this reason, the dimensionality of input layer of the decoder side is lower than that of its output layer. If this network is trained to minimise a reconstruction error, it is able to learn the most important attributes of the input data and how to best reconstruct the original input from an encoded state; ideally, this encoding learns and describes the *latent* attributes of the input data.

Other well known AE architectures are the *denoising* AE and the *sparse* AE. The former AE is largely used for the denoising of images, i.e. to reconstruct a clean image from a corrupted version of it. This task is

[18] *Unwrapped* phases are employed in this case, since they ease network training

[19] Note that, in Fig. 14 and in the following figures, each layer is represented by a prism having a rectangular base and whose height is proportional to the overall number of its units.

accomplished by storing only the relevant and recurrent features of an image inside the hidden layers, so that the noise affecting it can be filtered out. The latter one, instead, makes an over-complete representation of its input available at its output.

In the following, we will take into consideration under-complete AEs only, since they are employed in various radar applications, as shown in Paragraphs VI-A-VI-D. It is also worth mentioning that, in such applications, autoencoding is often employed as pre-processing method preceding supervised classification; this allows to learn repetitive structures of input data when the training dataset is not so large. The last application of AEs will be analysed in Paragraph IV-D3 in more detail.

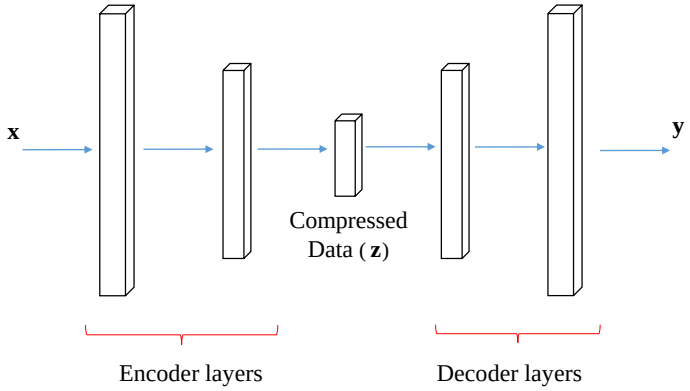


Figure 14: Architecture of an under-complete autoencoder characterized by a symmetric encoding-decoding structure.

Let us focus now on a possible application of auto-encoding to an FMCW radar system equipped with the antenna array shown in Fig. 4-b) ($d = \lambda/4$ is assumed) and operating in the presence of at most a single point target. In this case, the set $\{\mathbf{r}_{0,q}, \mathbf{r}_{1,q}, \mathbf{r}_{2,q}, \mathbf{r}_{3,q}\}$ (see eq.(7)), consisting of four N-dimensional noisy vectors becomes available in the q-th trial, with $q = 0, 1, ..., N_t - 1$. Each of these vectors undergoes the DFT processing described in Paragraph III-A; this allows to compute the $4\hat{N}_0$-dimensional feature vector

$$\begin{aligned}\mathbf{R}_q &= [R_{q,0}, R_{q,1}, ..., R_{q,4\hat{N}_0-1}]^T \\ &\triangleq \left[\left(\mathbf{Y}_0^{(q)}\right)^T, \left(\mathbf{Y}_1^{(q)}\right)^T, \left(\mathbf{Y}_2^{(q)}\right)^T, \left(\mathbf{Y}_3^{(q)}\right)^T\right]^T\end{aligned} \tag{187}$$

for any q; here, for any v,

$$\begin{aligned}\mathbf{Y}_v^{(q)} &= [Y_{v,0}^{(q)}, Y_{v,1}^{(q)}, ..., Y_{v,\hat{N}_0-1}^{(q)}]^T \\ &\triangleq M_r[|X_{v,b_m}|, |X_{v,b_m+1}|, ..., |X_{v,b_M}|]^T\end{aligned} \tag{188}$$

is an $\hat{N}_0$-dimensional vector, $X_{v,k}$ is the k-th element of the N_0-dimensional vector $\mathbf{X}_v^{(q)}$ computed on the basis of eq. (53) (with $k = b_m, b_m + 1, ..., b_M$), M_r is the oversampling factor employed in DFT processing,

$$\hat{N}_0 \triangleq b_M - b_m + 1, \tag{189}$$

and b_m and b_M are integer parameters delimiting the portion of the received signal spectrum over which an amplitude peak, due to the presence of a possible target, is expected. Note that the couple (b_m, b_M) represents a form of a priori information and that, in general, the inequality $0 \leq b_m < b_M \leq N_0 - 1$ holds. Let assume now that the overall data set

$$\mathcal{D}_o = \{(\mathbf{R}_q, t_q); q = 0, 1, ..., \hat{N}_t - 1\}, \tag{190}$$

acquired in $\hat{N}_t = 2400$ independent trials, is available; here, the label $t_q = 1$ (-1) refers to the presence of a *real* (*false*) target detected on the basis of the deterministic strategy expressed by eq. (157). Moreover, the following assumptions are made in synthetically generating the set $\mathcal{D}_o$ (190):

a) Half of its data are associated with the detection of a real target, the remaining half with the detection of a false target.

b) The parameters of the employed radar system take on the same values as those selected for the example illustrated for the SVM and K–NN methods in Paragraph III-B.

c) The stochastic models adopted for amplitude $a_v^{(q)}$ of the sinusoid observed on the v-th antenna in the presence of a real target, and for the range R_q and the azimuth ϕ_q of the target (if present) are the same as those defined in the example illustrated for the SVM and K–NN methods in Paragraph III-B.

d) The size $\hat{N}_0$ of the vector $\mathbf{X}_v^{(q)}$ is equal to 121, since (see eq. (189))

$$b_m = \left\lfloor \frac{2R_m\,\mu}{c} N_0 T_s \right\rfloor = 13 \tag{191}$$

and

$$b_M = \left\lfloor \frac{2R_M\,\mu}{c} N_0 T_s \right\rfloor = 133, \tag{192}$$

where $R_m = 0.5$ m ($R_M = 5.0$ m) represent the minimum (maximum) range expected for the target.

An AE is employed in the considered radar system to reduce the dimensionality of the feature vector $\mathbf{R}_q$ (187) (whose size is $4\hat{N}_0 = 484$); note that, unlike the deterministic approach described in Paragraph III-A and based on a maximum search, an unsupervised data-driven method is exploited in this case. The AE architecture we adopt is similar to the one illustrated in Fig. 14, but includes only a single layer in its encoder and a single layer in its decoder, for simplicity. The compressed representation available at the output of the encoder layer is represented by the M-dimensional vector

$$\mathbf{z}_q \triangleq \mathbf{h}_e\left(\mathbf{W}_e \mathbf{R}_q + \mathbf{b}_e\right) \tag{193}$$

collecting the hidden variables; here, $\mathbf{W}_e$ is a weight matrix of size $M \times 4\,\hat{N}_0$, $\mathbf{b}_e$ is an M-dimensional bias vector and $\mathbf{h}_e\left(\mathbf{x}\right)$ is an M-dimensional vector resulting from the element-by-element application of the positive saturating linear transfer function

$$h(x) = \begin{cases} 0 \;\text{ if }\; x \leq 0 \\ x \;\text{ if }\; 0 < x < 1 \\ 1 \;\text{ if }\; x \geq 1 \end{cases} \tag{194}$$

to the $4\hat{N}_0$-dimensional input vector $\mathbf{R}_q$ (187). The decoder maps the encoded representation $\mathbf{z}_q$ (193) back to the $4\hat{N}_0$-dimensional vector

$$\begin{aligned} \mathbf{y}_q &= [y_{q,0}, y_{q,1}, ..., y_{q,4\hat{N}_0-1}]^T \\ &\triangleq \mathbf{W}_d\,\mathbf{z}_q + \mathbf{b}_d, \end{aligned} \tag{195}$$

that represents an estimate of the original input vector; here, $\mathbf{W}_d$ is a $4\hat{N}_0 \times M$ weight matrix and $\mathbf{b}_d$ is an $4\hat{N}_0$-dimensional bias vector. In our simulations, $M = 60$ has been selected; consequently, a 60-dimensional hidden vector is extracted from a 484-dimensional observation (i.e., roughly an eightfold dimensionality reduction is achieved). Moreover, the *scaled conjugate gradient* method [82] has been employed to train the AE. Training is based on the dataset $\mathcal{D}$, that contains 90% of the dataset $\mathcal{D}_o$ (190) and, consequently, involves $N_t = 2160$ observations; the remaining part of the dataset, whose size is $\bar{N}_t = 240$, forms the test set $\mathcal{D}_{ts}$. The effectiveness of the employed AE is exemplified by Fig. 15, where the output vector $\mathbf{y}_q$ generated by the autoencoder in response to a specific feature vector $\mathbf{R}_q$ of the test set is shown; this is also confirmed by the small RMSE evaluated over $\mathcal{D}_{ts}$: $RMSE = 0.1$ V is found in this case. This leads to the conclusion that the compressed representation computed by the AE and expressed by the vector $\mathbf{z}_q$ (193) is really able to capture all the relevant information conveyed by the input vector $\mathbf{R}_q$ (187).

Finally, it worth mentioning that the compressed representation $\mathbf{z}_q$ (193) can be exploited to train the linear SVM and K-NN methods described in Paragraph III-B4 and employed to discriminate between real and false targets. In our experiment, these two supervised methods have trained on a dataset consisting of $N_t = 240$ observations ($K = 4$ has been selected for the K-NN method); half of them are associated with the detection of a real target, half with the detection of a false target. Our computer simulations have evidenced that, despite the dimensionality reduction, a slightly better accuracy is achieved by the considered classification techniques; in fact, the obtained accuracies are equal to 93% and 97% for the K-NN and the linear SVM, respectively (N-fold cross validation, with $N = 5$, has been used).

2) Convolutional neural networks: Convolutional neural networks (CNNs) play an important role in DL applications, since they allow to exploit the spatio-temporal information available in a sequence of images [80], [83]; for this reason, they are trained using a labelled dataset. The processing performed by a CNN aims at capturing the local features of input images and is based on *spatially localized convolutional filtering*. Its typical architecture includes *convolutional*, *pooling*, *fully connected* layers, and is motivated by the fact that, in images, local groups of values may exhibit high correlation and local statistics are invariant to position. In fact, convolutional layers aim at detecting local features on the basis of the data originating from the previous

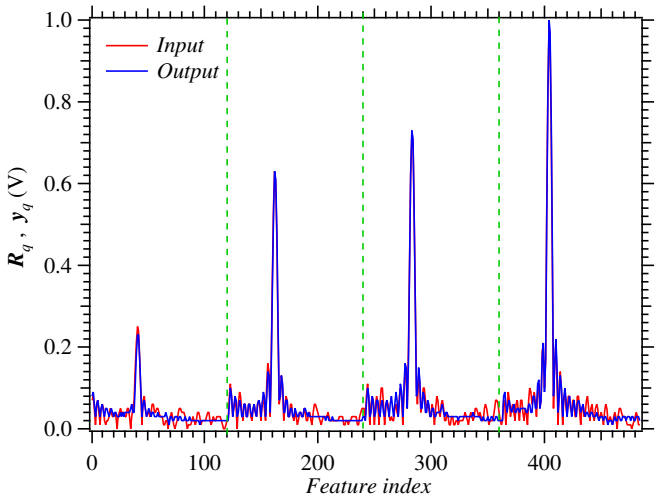


Figure 15: Example of a feature vector $\mathbf{R}_q$ (187) (red line) and of the corresponding output vector $\mathbf{y}_q$ (195) (blue line) predicted by the AE employed in the example of Paragraph IV-D1. The contributions of the four vectors $\{\mathbf{Y}_v^{(q)};\, v = 0,\, 1,\, 2,\, 3\}$ which $\mathbf{R}_q$ is made of are delimited by green dashed lines.

layer, pooling layers at merging semantically similar features and fully connected layers at generating the final feature vector.

The processing accomplished by the convolution and pooling operations on a greyscale image is outlined in Fig. 16. The output of the convolution depends on both its input, represented by a small portion of the image, and the adopted convolution *kernel*, denoted $\{K[m,n]\}$; moreover, this operation is repeated on multiple disjoint portions until the whole input image is scanned. From a mathematical viewpoint, the convolution input is a matrix, consisting of $I_S \times I_S$ pixels and denoted $\mathbf{I} = [I[i,j]]$, whereas the resulting output is a $S_Y \times S_Y$ matrix, called *activation* or *feature map* and denoted $\mathbf{Y} = [Y[i,j]]$. The (i,j)-th element (i.e., unit) of the activation map is evaluated as

$$Y[i,j] \triangleq \sigma\left(\sum_{m=-F/2}^{F/2-1}\sum_{n=-F/2}^{F/2-1} K[m,n]\, I[i-m, j-n]\right), \tag{196}$$

where F and $K[m,n]$ are the *size* of the convolutional filter (also known as *kernel size*) and its (m,n)-th weight, respectively, and $\sigma(\cdot)$ is a non linear *activation function*. Another relevant parameter of a convolutional layer is its *stride* S, that represents the number of pixels shifts over the input matrix when the kernel moves from a portion of the image to the next one; for instance, when the stride is one, the filter moves one pixel at a time. The area of the input image processed by the kernel can be also extended by adding a set of pixel (usually set at zero) to the border of image itself, as shown in Fig. 16; in that figure, the parameter P (dubbed *padding*) represents the number of zero columns and rows added to the input image. The stride, the padding and the kernel size of a convolutional layer influence the size S_Y of the output matrix; in fact, it can be shown that

$$S_Y = \frac{I_S - F + 2\,P}{S} + 1. \tag{197}$$

For this reason, the above mentioned parameters have to be jointly selected in a way that the RHS of last equation takes on an integer value.

Generally speaking, the convolution operation expressed by eq. (196) can be performed N_d times over the same image; in accomplishing this procedure, the parameters P and S do not change. This produces the output volume (i.e., matrix) $\mathbf{W}$ shown in Fig. 16 and having size $S_Y \times S_Y \times N_d$ (the parameter N_d is called *depth*); this matrix results from stacking N_d distinct activation maps, each representing a specific *slice*.

The convolutional layer represented in Fig. 16 feeds a *pooling layer*, whose task is reducing the dimensionality of each input slice and, consequently, the overall complexity of the considered CNN. The processing accomplished by the pooling layer can be easily described by referring to a single slice, denoted $\mathbf{Y}$, of the output volume $\mathbf{W}$. Similarly as the convolution operation, the pooling operation is fed by a portion, having size $F_p \times F_p$, of the considered slice and generates the $S_{Y_p} \times S_{Y_p}$ output matrix $\mathbf{Y}_p = [Y_p[i,j]]$. The most popular

layers of this type are known as *max pooling* and as *average pooling*. In the former case, the (i,j)-th pixel of the output matrix $\mathbf{Y}_p$ is computed as

$$Y_p[i,j] \triangleq \max_{p,q \in \mathcal{S}_{F_p}(i,j)} Y[p,q], \tag{198}$$

whereas in the latter one as

$$Y_p[i,j] \triangleq \frac{1}{F_p^2} \sum_{p,q \in \mathcal{S}_{F_p}(i,j)} Y[p,q], \tag{199}$$

where $\mathcal{S}_{F_p}(i,j) \triangleq \{(p,q)| -F_p/2+i \leq p \leq i+F_p/2-1, -F_p/2+j \leq p \leq F_p/2-1+j)\}$ and the parameter F_p is called *pool size*. It can be shown that

$$S_{Y_p} = \frac{S_Y - F_p}{S_p} + 1, \tag{200}$$

where S_p is the *stride* of the pooling (its meaning is similar to that illustrated above for the parameter S). Note that the depth N_d of the final output volume $\mathbf{W}_p$ generated by pooling is the same as that of $\mathbf{W}$.

Finally, it is important to point out that:

a) in CNN applications, a chain of pairs of convolutional and pooling layers is commonly used. Moreover, fully connected layers (FC) of different lengths are often added at the end of the cascade of convolutional/pooling layers, as illustrated in Fig. 17; this allows to combine all the extracted features in a 1D vector.

b) As shown in Paragraphs VI-A-VI-D, CNNs are employed in a number of radar applications ranging from human activity characterization to autonomous driving. Some experimental results about the use of CNNs in the classification of three different human activities CNN are illustrated in Section VIII.

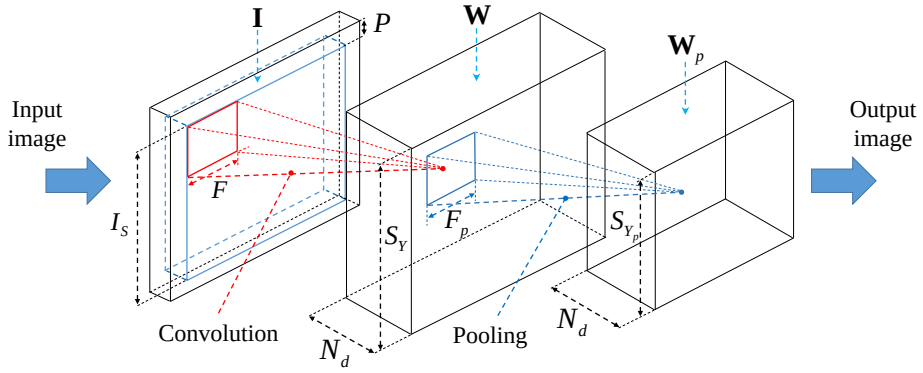


Figure 16: Representation of the *convolution* and *pooling* operations accomplished by a CNN on a greyscale image. The area corresponding to the convolution input (red square) is moved from left to right, and up and down over the input image. The convolution generates the activation map $\mathbf{Y}$, that represents a portion of the output volume $\mathbf{W}$. Pooling is employed to reduce the size of the final map $\mathbf{Y}_p$.

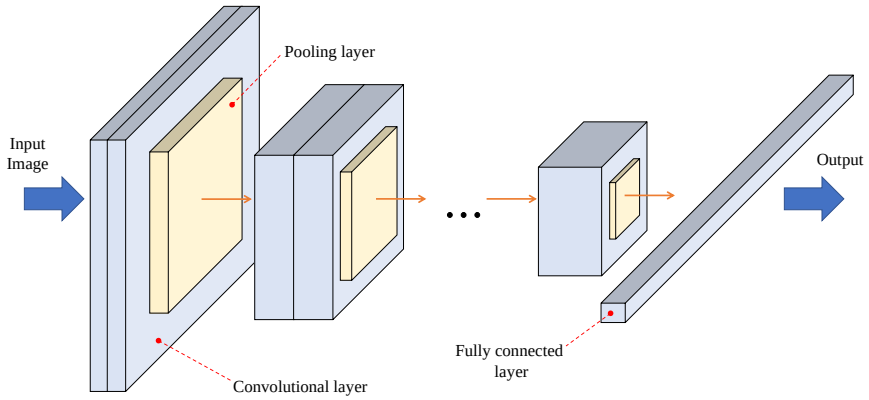


Figure 17: Architecture of a CNN containing multiple convolutional layers, max pooling layers and fully connected layers.

3) Convolutional autoencoders: A *convolutional autoencoder* (CAE) may represent an appealing technical option in those applications in which a large amount of labelled data is unavailable. In fact, it combines the advantages offered by unsupervised learning techniques (i.e., by AEs) with the capability of CNNs to extract the spatio-temporal information from a sequence of images. The architecture of a CAE is exemplified by Fig. 18; this network consists of an encoder side, combining convolutional and pooling (i.e., *downsampling*) layers, and of a decoder side, made of transposed convolutional (also called *deconvolutional*) and unpooling (i.e., *upsampling*) layers. Each transposed convolutional layer allows to upsample its input feature map with the aim of retrieving the original shape of the image available at the input of the first convolutional layer contained in the encoder side. In each unpooling layer, instead, an upsampling procedure exploiting the positions of the maxima stored in the corresponding max pooling operation executed at the encoder side is accomplished.

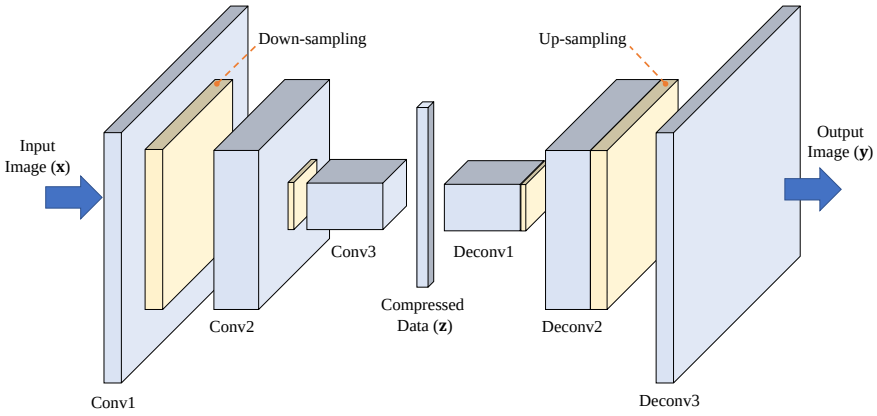


Figure 18: Example of CAE architecture.The acronym *ConvX* (with $X = 1, 2$ and 3) identifies the X-th convolutional and pooling layer, whereas *DeconvX* (with $X = 1, 2$ and 3) the transpose and unpooling layer. The vector $\mathbf{z}$ can be considered as a compressed representation of the input image $\mathbf{x}$.

4) Recurrent neural networks: In the neural networks treated so far, all the inputs and all the outputs are time-independent from each other. Features related to the time evolution of the observed data can be extracted through a *recurrent neural network* (RNN) [84]. A well known example of RNN is the so called *Vanilla* RNN, whose architecture is represented in Fig 19-a). In this network, past information contribute to the computation of its output, since they are reinjected into the network itself and stored in its internal (i.e., hidden) state. Moreover, the following three distinct weight matrices are employed by this network: a) the $M \times D_r$ matrix $\mathbf{U}$ employed in the mapping of the D_r-dimensional input vector $\mathbf{r}_q^{(t)}$ at time t to its M-dimensional hidden state $\mathbf{h}^{(t)}$; b) the $M \times M$ square matrix $\mathbf{W}$ involved in the update of its internal state; c) the $D'_r \times M$ matrix $\mathbf{V}$ employed to map $\mathbf{h}^{(t)}$ to its D'_r-dimensional output vector $\mathbf{o}^{(t)}$. In fact, based on these matrices, the state update of the network and the computation of its output at time t can be expressed as

$$\mathbf{h}^{(t)} = \boldsymbol{\phi}(\mathbf{W}\,\mathbf{h}^{(t-1)} + \mathbf{U}\,\mathbf{r}_q^{(t)}) \tag{201}$$

and as

$$\mathbf{o}^{(t)} = \mathbf{V}\,\mathbf{h}^{(t)}, \tag{202}$$

respectively; here, $\boldsymbol{\phi}(\cdot)$ denotes a non-linear activation vector function.
It is important to point out that:

a) A RNN can be thought as the result of the interconnection of multiple copies of the same network, each passing a message to a successor. In fact, *unrolling* it leads to a chain-like architecture, made of multiple replicas of the same module and such that each module passes a message to its successor.

b) The standard procedure for training a RNN is known as *backpropagation through time* (BPTT) [85]. Unluckily, it may not be so effective when training involves long time sequences, because of the so called *vanishing* and the *exploding gradient* problems [86]. The former problem refers to the exponential decrease observed in the norm of the gradient of the employed cost function during training, whereas the latter one concerns the opposite behaviour (more specifically, a large increase of the same gradient).

The problems mentioned in the last point can be circumvented by adopting a *long short term memory* (LSTM) neural network [87], whose architecture is illustrated in Fig. 19-b). This architecture consists of a *memory cell* and of three different *multiplicative gates*, namely an input gate, an output gate and a forget gate. The input gate, whose content at time t is denoted $\mathbf{i}^{(t)}$, represents the input of the memory cell (whose content at time t is denoted $\mathbf{c}^{(t)}$) and is employed to protect the content of this cell from perturbations due to irrelevant inputs.

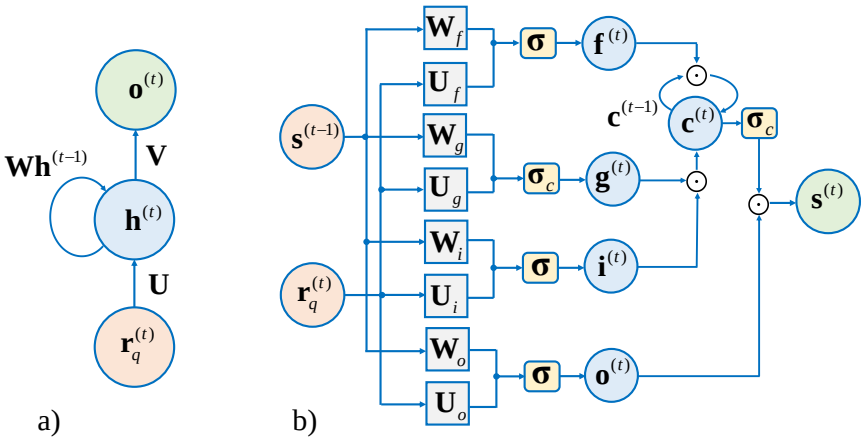


Figure 19: Architecture of: a) a Vanilla RNN; b) an LSTM neural network.

The output gate, whose content at time t is denoted $\mathbf{o}^{(t)}$, protects the other units connected to the output of the memory cell from perturbations due to irrelevant memory contents. Finally, the forget gate, whose content at time t is denoted $\mathbf{f}^{(t)}$, protects the contents stored in the vector $\mathbf{c}^{(t)}$ from the unwanted fluctuations of the memory at the previous instance (i.e., from $\mathbf{c}^{(t-1)}$). In summary, the cell allows for long term memory storage, whereas the gates prevent memory contents from being perturbed by irrelevant inputs and outputs.

If $\mathbf{r}_q^{(t)}$ denotes the vector of input features at time t, the time evolution of the LSTM network shown in Fig. 19-b) is described by the equations

$$\mathbf{i}^{(t)} = \boldsymbol{\sigma}(\mathbf{U}_i\,\mathbf{r}_q^{(t)} + \mathbf{W}_i\,\mathbf{s}^{(t-1)}), \tag{203}$$

$$\mathbf{f}^{(t)} = \boldsymbol{\sigma}(\mathbf{U}_f\,\mathbf{r}_q^{(t)} + \mathbf{W}_f\,\mathbf{s}^{(t-1)}), \tag{204}$$

$$\mathbf{o}^{(t)} = \boldsymbol{\sigma}(\mathbf{U}_o\,\mathbf{r}_q^{(t)} + \mathbf{W}_o\,\mathbf{s}^{(t-1)}), \tag{205}$$

$$\mathbf{g}^{(t)} = \boldsymbol{\sigma}_c(\mathbf{U}_g\,\mathbf{r}_q^{(t)} + \mathbf{W}_g\,\mathbf{s}^{(t-1)}), \tag{206}$$

$$\mathbf{c}^{(t)} = \mathbf{c}^{(t-1)} \odot \mathbf{f}^{(t)} + \mathbf{g}^{(t)} \odot \mathbf{i}^{(t)} \tag{207}$$

and

$$\mathbf{s}^{(t)} = \boldsymbol{\sigma}_c(\mathbf{c}^{(t)}) \odot \mathbf{o}^{(t)}; \tag{208}$$

here, $\boldsymbol{\sigma}(\cdot)$ is a logistic sigmoid vector function, $\boldsymbol{\sigma}_c(\cdot)$ is an hyperbolic tangent vector function, the operator $\odot$ denotes the Hadamard product, $\mathbf{s}^{(t)}$ is the output of the memory cell at time t, $\mathbf{U}_i$, $\mathbf{U}_f$, $\mathbf{U}_o$ and $\mathbf{U}_g$ ($\mathbf{W}_i$, $\mathbf{W}_f$, $\mathbf{W}_o$ and $\mathbf{W}_g$) are weight matrices characterizing the multiplicative gates and referring to the vector $\mathbf{r}_q^{(t)}$ ($\mathbf{s}^{(t-1)}$), and $\mathbf{g}^{(t)}$ can be interpreted as a candidate state, whose influence on the state $\mathbf{c}^{(t)}$ is controlled by the input gate through $\mathbf{i}^{(t)}$. From eqs. (203)-(208) it is easily inferred that: a) the contents of the input, output and forget gates at time t are proportional to a combination of both the vectors $\mathbf{r}_q^{(t)}$ and $\mathbf{s}^{(t-1)}$; b) the output state $\mathbf{s}^{(t)}$ depends not only on the cell content $\mathbf{c}^{(t)}$, but also on the content of the output gate (i.e. on $\mathbf{o}^{(t)}$).

Let us focus now on the application of an LSTM neural network to an FMCW radar system equipped with a single TX-RX pair and detecting a person that accomplishes specific activities and, in particular, that runs or walks. In this case, each observation processed by the LSTM involves N_f consecutive frames, in each of which N_c chirps are transmitted (see Paragraph II-C). For this reason, the q-th observation processed by the considered network is extracted from $N_f\,N_c$ noisy vectors, acquired over N_f consecutive frames, (i.e., over $N_f\,N_c$ consecutive chirps). In the p-th frame (with $p = 0, 1, ..., N_f - 1$), the set of vectors $\{\mathbf{r}_{p,0}^{(q)}, \mathbf{r}_{p,1}^{(q)}, ..., \mathbf{r}_{p,N_c-1}^{(q)}\}$, each having size N, is available; here,

$$\mathbf{r}_{p,k}^{(q)} = \left[r_{p,k,0}^{(q)}, r_{p,k,1}^{(q)}, ..., r_{p,k,N-1}^{(q)}\right], \tag{209}$$

represents the vector of signal samples acquired in the k-th chirp interval of the p-th frame and its n-th sample $r_{p,k,n}^{(q)}$ is expressed by a formula similar to eq. (22) (with $n = 0, 1, ..., N-1$). In our experiment, the *Phased Array System* toolbox available in the MATLAB environment is employed to generate the useful signal component (i.e., the contribution of the detected person) to the vector $\mathbf{r}_{p,k}^{(q)}$ (209) [88]. This contribution is modelled as the

superposition of L different echoes, each originating from a point-like target and associated with a different part of the body. Moreover, in the p-th frame contributing to the q-th observation, the l-th point target is characterized by its RCS $a_{p,l}^{(q)}$, its range $R_{p,l}^{(q)}$ and its radial velocity $v_{p,l}^{(q)}$ (with $p = 0, 1, ..., N_c - 1$ and $l = 0, 1, ..., L - 1$). These parameters are assumed to be static over each frame; in addition, the values they take on in the p-th frame are automatically computed by the above mentioned toolbox on the basis of the height h_p of the person, its position and its RCS in the previous (i.e., in the $(p-1)$-th) frame, the direction of its movement with respect to the radar system and its radial velocity v_q. The dataset processed by the network is

$$\mathcal{D}_o \triangleq \{(\mathbf{R}_q, t_q); q = 0, 1, ..., \hat{N}_t - 1\}, \tag{210}$$

where

$$\mathbf{R}_q \triangleq \left[\left(\mathbf{x}_0^{(q)}\right)^T, \left(\mathbf{x}_1^{(q)}\right)^T, ..., \left(\mathbf{x}_{N_f-1}^{(q)}\right)^T \right]^T, \tag{211}$$

is the q-th noisy observation, t_q its label,

$$\mathbf{x}_p^{(q)} \triangleq [\hat{R}_p^{(q)}, \hat{v}_p^{(q)}]^T, \tag{212}$$

and $\hat{R}_p^{(q)}$ and $\hat{v}_p^{(q)}$ are the estimates of the range $R_p^{(q)}$ and of the velocity $v_p^{(q)}$, respectively, of the considered person in the p-th frame (with $p = 0, 1, ..., N_f - 1$); moreover, it is assumed that $t_q = 1$ (-1) if the person is walking (running), i.e. if $|v_p^{(q)}| \leq v_{th}$ ($|v_p^{(q)}| > v_{th}$), being v_{th} a proper threshold.

In our experiment, the dataset $\mathcal{D}_o$ (210) has been acquired in $\hat{N}_t = 400$ independent trials; half of the labels of this dataset are associated with a walker and the remaining half with a runner; moreover, the estimates $\hat{R}_p^{(q)}$ and $\hat{v}_p^{(q)}$ are computed by the algorithm consisting of the following two steps:

1) *Range Estimation* - In this step, the N-dimensional vector $\mathbf{r}_{p,k}^{(q)}$ (209) undergoes zero padding; this results in the N_0-dimensional vector $\mathbf{r}_{p,k,\mathrm{ZP}}^{(q)}$, with $N_0 \triangleq M_r N$ (here, the parameter M_r represents the selected oversampling factor). The last vector feeds a N_0-th order FFT, whose output is the N_0-dimensional vector $\mathbf{X}_{p,k}^{(q)} = [X_{p,k,0}^{(q)}, X_{p,k,1}^{(q)}, ..., X_{p,k,N_0-1}^{(q)}]^T$. Then, the average power spectrum

$$P_m^{(q)} \triangleq \frac{1}{N_c} \sum_{k=0}^{N_c-1} |X_{p,k,m}^{(q)}|^2, \tag{213}$$

is computed for $k = 0, 1, ..., N_c - 1$. Finally, $\hat{R}_p^{(q)}$ is evaluated as (see eqs. (58), (60) and (62))

$$\hat{R}_p^{(q)} = \frac{c}{2\mu} \hat{f}_p^{(q)}, \tag{214}$$

where $\hat{f}_p^{(q)} = \hat{m}_p^{(q)} / N_0 T_s$ and

$$\hat{m}_p^{(q)} = \arg \max_{m \in \{0,1,...,N_0/2\}} P_m^{(q)}. \tag{215}$$

2) *Velocity Estimation* - This step is based on the N_c-dimensional vector

$$\hat{\mathbf{A}}_p^{(q)} = [\hat{A}_{p,0}^{(q)}, \hat{A}_{p,1}^{(q)}, ..., \hat{A}_{p,N_c-1}^{(q)}]^T, \tag{216}$$

where

$$\hat{A}_{p,k}^{(q)} = M_r \, X_{p,k,\hat{m}_p^{(q)}}^{(q)} \tag{217}$$

and $\hat{m}_p^{(q)}$ is expressed by eq. (215). Applying zero padding to this vector produces the $N_0^{'}$-dimensional vector $\hat{\mathbf{A}}_{p,\mathrm{ZP}}^{(q)}$, with $N_0^{'} \triangleq M_A N_c$ (here, the parameter M_A represents the selected oversampling factor); the last vector feeds a $N_0^{'}$-th order FFT, whose output is the $N_0^{'}$-dimensional vector

$$\mathbf{d}_p^{(q)} \triangleq \left[d_{p,0}^{(q)}, ..., d_{p,N_0'/2}^{(q)}, d_{p,-N_0'/2+1}^{(q)}, ..., d_{p,-1}^{(q)} \right]^T. \tag{218}$$

After solving the maximization problem

$$\hat{k}_p^{(q)} = \arg \max_{\tilde{k} \in \{-N_0'/2+1, -N_0'/2+2, ..., N_0'/2\}} \left| d_{p,\tilde{k}}^{(q)} \right|, \tag{219}$$

the estimate (see eqs. (23) and (60))

$$\hat{v}_p^{(q)} = \frac{1}{2} f_p^{(q)} \lambda \tag{220}$$

of the person velocity is evaluated; here,

$$f_p^{(q)} \triangleq \frac{2\hat{k}_p^{(q)}}{N_0' T_0} \tag{221}$$

represents the Doppler frequency estimated in p-th frame.

For any q, in generating the sequence of pairs $\{(\hat{R}_p^{(q)}, \hat{v}_p^{(q)}); t = 0, 1, ..., N_f - 1\}$, the following assumptions have been made about the detected person:

a) its response to the signal radiated by the radar system consists of $L = 16$ echoes;

b) its height h_p is uniformly distributed over the interval $(1.70\,,\ 2.0)$ m;

c) its initial coordinates in a 3D space are $(x_0^{(q)},\ y_0^{(q)},\ z_0^{(q)}) = (0,10,0)$ m, whereas the coordinates of the employed radar device in the same reference system are $(x_r,\ y_r,\ z_r) = (0,0,1)$ m;

d) the angle ϕ_i representing the initial direction of its velocity is uniformly distributed over the domain $(60^\circ\,,\ -60^\circ) \cup (120^\circ, 180^\circ)$ (the reference line, with respect to which this angle is measured, is perpendicular to the array of the radar system).

e) the radial velocity $v_p^{(q)}$ is uniformly distributed over the interval $(0.1,\ 2.1)$ m/s ($(-2.1,\ -0.1)$ m/s) if $\phi_i \in (60^\circ\,,\ -60^\circ)$ ($\phi_i \in (120^\circ,\ 180^\circ)$) for any p and, in each frame, changes in an independent fashion;

f) the initial amplitude $a_{0,l}^{(q)}$ is equal to 1V (with $l = 0,\ 1,\ ...,\ L - 1$).

Moreover, the following choices have been made for the employed radar system:

a) the transmitted waveform is characterized by $\lambda = 4$ mm, $\mu = 1.5625 \cdot 10^{13}$ Hz s^{-1}, $T = 64$ μs and $T_0 = 72$ μs;

b) each frame consists of $N_c = 128$ chirps;

c) consecutive frames are separated by a time interval lasting $\Delta t = 40$ ms;

d) the sampling period employed at the receive side is $T_s = 12.5\,\mu$s;

e) the overall number of time-domain samples acquired in each chirp interval is $N = 1024$ and the standard deviation of the noise affecting each sample is $\sigma_w = 0.1$ V (see eq. (22));

f) the oversampling factors $M_r = 2$ and $M_A = 8$, and the threshold $v_{th} = 1.1$ m/s are selected for the range/estimation algorithm illustrated above;

g) each observation refers to $N_f = 30$ consecutive frames.

The representation, on a Cartesian plane, of two different feature vectors (see eqs. (211)-(212)), is provided in Fig. 20. These vectors are denoted $\mathbf{R}_0$ and $\mathbf{R}_1$; the former refers to a runner, whereas the latter to a walker. Note that the range difference $\Delta\hat{R}_0 \triangleq |\hat{R}_{N_f-1}^{(0)} - \hat{R}_0^{(0)}|$ referring to the runner is greater than the corresponding quantity (i.e., $\Delta\hat{R}_1 \triangleq |\hat{R}_{N_f-1}^{(1)} - \hat{R}_0^{(1)}|$) referring to the walker. In this case, the proposed LSTM network is employed to discriminate a walker from a runner. The core of its architecture is characterized by an LSTM layer, able to learn the long term dependencies between different frames. The behaviour of network is described by the block diagram shown in Fig. 19 and by eqs. (203)-(207) (the time index t corresponds to the frame index p in this case). Moreover, in our experiment, the following choices have been made:

1) the size of the input vector is $D_r = 2$, whereas that of the inner state is $M = 10$;

2) the non-linear gate activation function $\sigma(x) = [1 + \exp(-x)]^{-1}$ is used;

3) the size of each of the weight matrices $\{\mathbf{U}_i,\ \mathbf{U}_f,\ \mathbf{U}_g,\ \mathbf{U}_o\}$ is $M \times D_r = 10 \times 2$, whereas that of the weight matrices $\{\mathbf{W}_i, \mathbf{W}_f, \mathbf{W}_g, \mathbf{W}_o\}$ is $M \times M = 10 \times 10$;

5) both the initial cell content $\mathbf{c}^{(0)}$ and the initial state $\mathbf{s}^{(0)}$ are independently chosen as random vectors of size $M = 10$;

6) a fully connected layer and a softmax layer[20] have been added at the output of the LSTM layer to perform classification.

The *adaptive moment estimation* (briefly, *adam*) optimizer [89] has been exploited to train the proposed network (i.e., to tune all the above mentioned weighted matrices); the batch size, the (constant) learning rate and the number of epochs selected for this procedure are $N_S = 32$, $\gamma = 10^{-3}$ and $N_E = 50$, respectively (see eq. (106)). Moreover, a training set $\mathcal{D}$ of size $N_t = 300$, corresponding to 75% of the dataset $\mathcal{D}_o$ (210) has been employed for training; the remaining part $\mathcal{D}_{ts}$ of $\mathcal{D}_o$ has been used as a test set (collecting $\bar{N}_t = 100$ observations)). Our results have evidenced that a 98% accuracy is achieved by the adopted LSTM network. These results suggest that:

a) Combining deterministic estimators with deep learning methods can result in classification techniques achieving excellent performance;

b) Merging range and velocity information can enhance the discrimination capability of the network;

c) Observing range/velocity evolution over time (i.e., over multiple consecutive frames) significantly contribute to improve network accuracy.

5) Generative adversarial networks: A *generative adversarial network* (GAN) is a probabilistic generative method consisting of two deep neural networks, called *generator* and *discriminator*, and competing one against each other [90]; its architecture is shown in Fig. 21. The generator produces a sample[21] $\mathbf{x} = \mathbf{G}(\mathbf{z}, \boldsymbol{\theta}_g)$ from

[20] See Par. IV-D6 for further details about this layer.

[21] Scalar variables are considered in this paragraph, for simplicity.

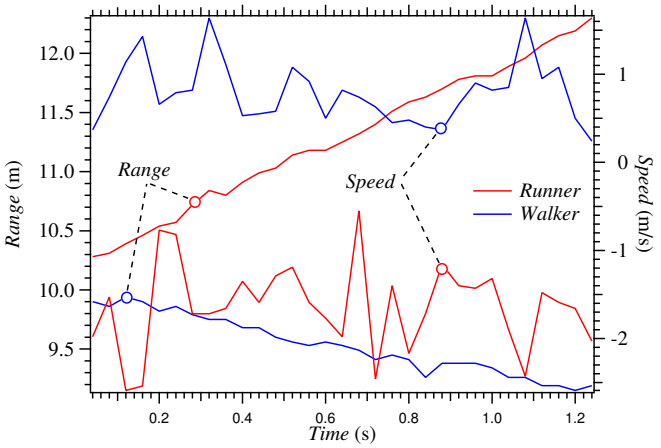


Figure 20: Representation of the elements of the two feature vectors $\mathbf{R}_0$ and $\mathbf{R}_1$; one refers to a runner (red lines), the other one to a walker (blue lines).

a pdf $f_g(\mathbf{x})$, starting from an input noise variable $\mathbf{z} \sim f_z(\mathbf{z})$; here, $\mathbf{G}(\cdot,\cdot)$ is called *generative model* and is typically implemented through a neural network, whereas $\boldsymbol{\theta}_g$ is the vector of training parameters.

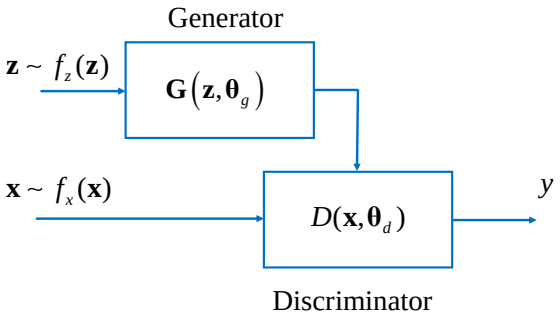


Figure 21: Architecture of a generative adversarial network.

The discriminator, instead, generates the output $y = D(\mathbf{x}, \boldsymbol{\theta}_d)$, that represents the probability that an input $\mathbf{x}$ originates from the training data (i.e., from their pdf $f_d(\mathbf{x})$) rather than from the generator pdf $f_g(\mathbf{x})$; here, $D(\cdot,\cdot)$ represents the *discriminative model* and $\boldsymbol{\theta}_d$ is the vector of the training parameters characterizing the network that implements the model itself. In practice, the generative model can be thought as a team of counterfeiters, trying to produce fake currency for fooling the police, while the discriminator, acting like the police, tries to detect the counterfeit currency. Competition in this game drives both teams to improve their methods. In fact, the objective of the training of the generative network is minimizing the accuracy of the discriminative network when the data generated by the former network are provided to the latter one; on the contrary, the objective of the discriminator is maximizing the probability of assigning the correct label to both the real data of the training set and the fake samples originating from the generator. For this reason, the interaction between the discriminator and the generator can be modelled as a *two-player minimax game*. This leads to formulating the optimal strategy of these networks as the solution of the minimax problem

$$\min_{\mathbf{G}} \max_{D} V(D, \mathbf{G}) \tag{222}$$

$$= \min_{\boldsymbol{\theta}_g} \max_{\boldsymbol{\theta}_d} V(D, \mathbf{G}), \tag{223}$$

where

$$\begin{aligned} V(D, \mathbf{G}) \triangleq\ & \mathrm{E}_{\mathbf{x}\sim f_d(\mathbf{x})} \left\{ \log D\left(\mathbf{x}, \boldsymbol{\theta}_d\right) \right\}] + \\ +\ & \mathrm{E}_{\mathbf{z}\sim f_z(\mathbf{z})} \left\{ \log\left(1 - D\left(\mathbf{G}\left(\mathbf{z}, \boldsymbol{\theta}_g\right), \boldsymbol{\theta}_d\right)\right) \right\}. \end{aligned} \tag{224}$$

The backpropagation algorithm can be used for training a GAN; the training process allows the discriminator of the considered GAN to learn, through a proper feature representation, how to identify real inputs among the generated data and, similarly, the generator how to generate realistic data.

Generative adversarial networks have the favourable property that a wide variety of functions can be incorporated into their model; these make them able to represent very sharp (and even degenerate) data distributions. However, their use require the availability of efficient tools to solve the minimax optimization problem (222). Moreover, a tight synchronization between the generator and the discriminator has to be guaranteed during training; in fact, if one of the two networks learns too quickly, the other one may fail to learn.

6) Softmax Classification Layer: Generally speaking, the DL methods illustrated above can be employed to extract the relevant features of an image. Once this result has been obtained, any multi-class problem referring to that image can be solved by adding a *softmax layer* to the employed network. If K classes are assumed, the target of this layer is generating the posterior probability

$$p_i = \frac{\exp(a_i(\mathbf{r}^{(L)}))}{\sum_{j=0}^{K-1} \exp(a_j(\mathbf{r}^{(L)}))} \tag{225}$$

for the i-th class, with $i = 0, 1, ..., K-1$; here,

$$a_j(\mathbf{r}^{(L)}) = \mathbf{w}_j^T \mathbf{r}^{(L)} + w_{j,0} \tag{226}$$

and $\mathbf{r}^{(L)}$ is an L-dimensional feature vector made available by the previous hidden (convolutional or LSTM) layer, and $\mathbf{w}_j$ and $w_{j,0}$ are an L-dimensional weight vector and a bias term, respectively, characterizing the softmax layer.

V. Comparison of ML and DL techniques

The ML and DL methods described in Secs. III and IV are compared in Table III in terms of: a) the type of their training procedure (*supervised*, S, or *unsupervised*, U); b) the complexity of their training procedure (*low*, L, *medium*, M, or *high*, H); c) their classification accuracy; d) their sensitivity to clutter and noise; e) the method they use for extracting the salient features (*manual*, M, or *automatic*, A); f) the size of the dataset they require in order to achieve a good generalization capability (*small*, S, or *large*, L). If these methods are employed for image classification, it should be always kept in mind that:

1) The K-NN and SVM methods require a limited computational load, but achieve low classification accuracy. They are outperformed by CNNs and CAEs at the price, however, of a substantially larger computational effort. Moreover, the last methods are insensitive to a spatial transformation of input data. To understand the importance of the last property (known as *invariance property*), let us take into consideration a CNN employed to classify different objects in a radar image. This network, thanks to the above mentioned property, is able to select only those portions of the images relevant for its task and its behaviour is not influenced by other irrelevant characteristics, such as the position of a given target or its rotation.
2) *Long short term memory networks* are able to cope with a sequence of signals evolving over time.
3) *Generative adversarial networks* are able to generate synthetic images on the basis of a set of noisy input data. This property can be exploited in radar systems to de-noise images [91] or to detect abnormalities.

VI. Applications of Machine and Deep Learning Techniques to MIMO Radars

In this section we focus on some applications of the learning methods illustrated in Sections III and IV to MIMO radar systems. More specifically, we illustrate the exploitation of these methods in the following fields: a) human motion characterization; b) *human gesture recognition* (HGR); c) fall detection and health-care monitoring; d) autonomous driving. Various research results are available in the technical literature about these fields; some essential manuscripts concerning each of them and the use of specific learning methods are listed in Table IV.

Before delving into the analysis of each application, it is worth pointing out that the processing accomplished at the receive side of any MIMO radar system employing a learning method for classification and/or regression is based on the block diagram shown in Fig. 22. First, the received signal undergoes frequency downconversion to generate its in phase and quadrature components. Sampling these components produces a stream of raw data, which is pre-processed (e.g., it may undergo FFT processing; see Paragraphs II-C and III-A) before extracting relevant features from it. Finally, these features are processed by a classifier or by a regression algorithm; in the former case, a specific object class is selected, whereas, in the latter one, an estimate of the parameters of interest is evaluated. Feature extraction is based on our prior knowledge about the employed radar system if a ML method is exploited; on the contrary, features are automatically selected and extracted from pre-processed data if a DL method is adopted. In the following paragraphs, various details about the processing accomplished by the blocks appearing in Fig. 22 are provided.

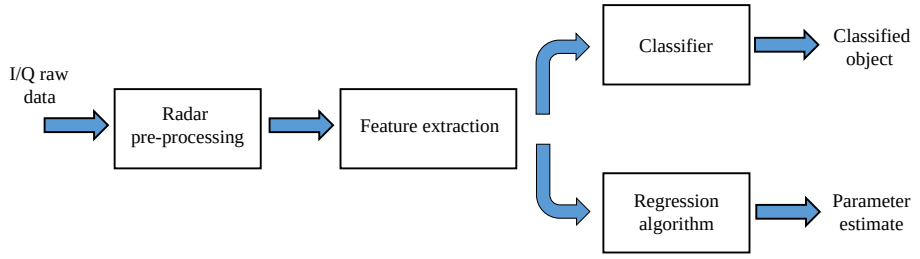


Figure 22: Block diagram representing the signal processing chain of a MIMO radar system that employs a learning method.

Methods	Learning (S/U)	Complexity (L/M/H)	Performance (L/H)	Clutter Sensitivity (L/H)	Features Extraction (M/A)	Dataset size (S/L)
KNN	S	L	L	L	M	S
SVM	S	L	L	L	M	S
Adaboost	S	M	H	H	M	S
PCA	U	L	L	L	A	S
K-Means	U	L	L	L	A	S
AE	U	H	L	H	A	S
CNN	S	H	H	H	A	L
CAE	U & S	H	H	H	A	L
RNN	S	H	H	H	A	L
GAN	S	H	H	H	A	L

Table III: Overview of the main characteristics of the learning techniques described in this manuscript.

A. Human motion characterization

Human motion characterization aims at recognizing and classifying different human activities on the basis of the micro-Doppler fluctuations observed in the spectrograms of radar signals. These fluctuations are known as *micro-Doppler signatures* (see Paragraph II-C). Classifiers employed for this application aim at: a) identifying different types of human motion (e.g., walking, running and sitting) [92]; b) differentiating human motion from that of other living animals [93]; c) remotely identifying potential active shooters [94], [95].

In the technical literature about this application, the following two methods are exploited to extract relevant features from spectrograms: a) manual extraction of *handcrafted* features; b) automatic extraction of features based on a *data-driven* approach. Machine learning methods exploiting manual extraction of features have been investigated in [95]–[103], whereas the automatic extraction of features from micro-Doppler signatures or spectrograms through DL methods has been proposed in [91], [92], [94], [104]–[106]. It is important to keep in mind that:

Learning method Field/Manuscript no.	Clustering	NB	K-NN	SVM	PCA	HMM	AE	CNN	CAE	LSTM	GAN
a) [91]–[106]		✓	✓	✓	✓		✓	✓	✓		✓
b) [107]–[111]						✓		✓		✓	
c) [10], [112]–[117]			✓	✓			✓	✓	✓		
d) [15], [118]–[131]	✓		✓	✓	✓		✓	✓	✓	✓	

Table IV: Specific learning methods investigated in various manuscripts that concern the four application fields considered in Section VI.

1) Machine learning methods relying on spectrogram information usually exploit: a) *physical features* related to the characteristics of the observed motion; b) *transform-based* features; c) *speech-inspired* features. Physical features include the frequency and the bandwidth of the received waveforms, the offset and the signal intensity of the associated signature (see Paragraph II-C). The first two physical features are strictly related to motion dynamics, whereas the other types of features to the RCS of the body of the observed person [93].
2) Transform-based features exploited by ML methods can be extracted from a received waveform by evaluating a) its spectral coefficients (e.g., its *discrete cosine transform* coefficients) or b) other signal-dependent coefficients. As far as point b) is concerned, the use of *linear predictive coding* (LPC) has been proposed in ref. [97] to transform a time-varying Doppler signal in a low dimensional set of prediction coefficients. A different approach, based on the computation of pseudo-Zernike moments, is illustrated in ref. [98, Sect. II-A, eqs. (10)-(12)]; this allows to extract relevant characteristics from micro-Doppler signatures, such as translational and scale invariance.
3) In many cases, the dimensionality of the feature space can be substantially reduced (see Paragraph III-C). An interesting example of this approach is offered in ref. [99], where the use of a 1D standard PCA and of a robust PCA for extracting physical features from a Doppler radar signal is investigated.

Frequently used ML methods for the classification of human motion based on a set of handcrafted features include the *naive Bayes* (NB) [100], the non linear SVM [101] and the K-NN methods [102]. The use of a binary SVM classifier for multi-class problems in human motion characterization is investigated in ref. [103], where a classification procedure based on a *decision-tree* is proposed. This procedure is based on the idea of representing a classification problem involving multiple choices as a set of binary classification problems, each of which is solved through a binary SVM. This approach is exemplified in [103, Fig. 8], where a decision tree referring to the case of seven classes is illustrated. In practice, a binary SVM is employed for each node of the employed decision tree in order to separate the possible activities in two groups; if each of the two groups is further divided, another SVM classifier is used at an underlying node.

The most relevant problems emerging from the study of ML-based classification of human motion concern the processing methods to be employed for the extraction of hand-crafted features from raw micro-Doppler signals, the sensitivity of these methods to noise and clutter, and the impact of similarities among the considered classes on their performance. The ability of a *deep neural network* to learn the relevant features directly from the available raw data allows to solve the above mentioned problems. This consideration has motivated the investigation of *deep* CNNs (DCNNs; see Paragraph IV-D2) for the automatic extraction of features in human motion characterization. The use of a DCNN, fed by spectrograms (converted in *red green blue*, RGB, or greyscale images), and employing convolutional layers and pooling layers of small size, has been proposed in ref. [104]. A different DL method, based on the same principles as convolutional autoencoding (see Paragraph IV-D3), has been developed in ref. [105, Par. IV-C, Fig.8]. It combines the ability of a DCNN to capture local features of input images with that of an AE to directly learn features through an unsupervised pre-training procedure. In this case, after an initial and unsupervised pre-training stage, the decoder of a CAE is substituted by a few fully connected layers and a softmax classifier. This procedure allows the resulting DCNN to learn specific patterns from the processed signatures, so easing training for supervised classification. The performance results obtained in this case lead to the conclusion that a CAE not only is able to outperform conventional classification methods based on handcrafted features (e.g., SVM), but also a standard DCNN.

Finally, it is useful to mention that another important research problem investigated in the field considered in this paragraph is represented by the *de-noising of micro-Doppler spectra*. In this case, the training set includes two different types of images: a) perfectly clean spectrograms; b) the same spectrograms affected by background noise. The use of a deep GAN, based on a convolutional encoder-decoder structure, has been proposed in ref. [91] for this application. The performance results obtained in this case evidence that this network does not affect the relevant components of micro-Doppler spectra and is able to outperform other classic de-noising techniques commonly used for the suppression of background noise.

B. Human gesture recognition

The significant attention paid to HGR is due to its exploitation in advanced *human computer interfaces* (HCIs), that are employed in a number of control, infotainment and security applications. Relevant information about the dynamics of human gestures are typically contained in the micro-Doppler signatures acquired over consecutive transmitted frames. Therefore, similarly as human activity characterization, relevant physical features can be easily extracted from spectrograms. A commonly employed ML tool for classifying vectors of handcrafted features in HGR systems is represented by *hidden Markov modelling* [107]. This approach leads to classifying a new sequence of data, called observation, on the basis of a stochastic model, called *hidden Markov model* (HMM), which has been extracted from past observations and describes their generation. If an HMM of a given

random phenomenon is available, the probability of observing a specific realization (e.g., a specific gesture), conditioned on a given sequence of hidden states, can be computed. In this case, model training aims at estimating the so called *transition* and *emission* probability matrices of the developed HMM; the former matrix collects the probabilities to move from a given state to another one, while the latter one the probabilities that a given observation is generated in each specific state. The efficacy of a HMM-based classifier depends on the overall number of states characterizing the model; in general, a larger number of states allows to model a more complicated process and to improve prediction accuracy. However, a discrete state space of small size is often adopted in HGR applications in order to mitigate the overall complexity of the developed HGR system. This choice makes the resulting classifier unable to distinguish gestures characterized only by subtle differences in their spectrograms. For this reason, DCNNs are usually preferred. One of the first important research activities focusing on the exploitation of this type of networks in HGR is the well known Google's *Soli project* [108], whose scope has been the development of a HGR mobile and wearable device based on a RF sensor. Various research results about this research field can be found in ref. [109], where it is shown that the accuracy of these deep classifiers gets worse if: a) the number of classes[22] increases; b) the incident angle and/or the distance between the gesture and the employed radar device get larger. The accuracy of a classifier based on a DCNN can be improved by extracting features not only from spectrograms, but also from range-Doppler maps [110]. Another DL architecture, specifically developed for RF HGR and combining the ability of a CNN network of capturing local features of input images with that of coping with time-varying signals, has been investigated in ref. [111]. This architecture consists of a 3D-CNN for spatial-temporal modelling of short consecutive frames, an LSTM for extracting global temporal features and a final classification layer (a detailed block diagram is illustrated in ref. [111, Fig. 7]). This architecture achieves a very high recognition accuracy, and outperforms other conventional ML and DL methods used in HGR applications, like HMMs or 2D-CNNs.

C. Fall detection and health-care monitoring

Human falls represent a worldwide health problem and are known to be one of the main causes of unintentional injury death in seniors; this motivates the recent interest in devising electronic systems able to detect their occurrence. Another important problem in the field of technology for human health concerns the development of non-invasive and non-contact devices for monitoring human vital signs, such as breath and heart rates, and sleep quality. Various results in both research areas have evidenced that innovative solutions to both problems can be developed by exploiting ML and DL methods fed by the micro-Doppler signatures acquired through a radar system. In any case, when the overall number of classes to be identified increases and the degree of dissimilarity between the Doppler signatures characterizing them reduces, DL methods are preferred, since they achieve better accuracy.

An interesting study on the dynamics of human falls analysed through micro-Doppler signatures can be found in in ref. [112], where it is shown that fall accidents can be distinguished from normal activities on the basis of: a) the strength of the received echo (i.e., the RCS of the subject under test); b) the distance of the radar device from the body of the subject under test during a fall; c) the Doppler information acquired during the movement of the subject itself. Experimental results have evidenced that, when a subject starts falling, the observed Doppler frequency increases steeply; on the contrary, the RCS of the human subject gradually decreases since its tilt angle gets larger. In this case, ML and DL algorithms can be trained to detect a fall on the basis of the time variations of Doppler signatures. A specific DL classifier based on a stacked AE and exploiting a range-Doppler radar has been developed in ref. [113], where it is shown that the proposed solution is more accurate than PCA-based methods in detecting different actions, such as falling, walking, sitting and bending.

The use of learning techniques in the analysis of sleep stages has been investigated in ref. [114], where a solution based on a K-NN classifier has been proposed.

The exploitation of learning techniques for heart and breath rate estimation represents a challenging problem, because a large and heterogeneous datasets for network training cannot be easily built and contactless systems for vital sign monitoring are strongly limited by body movements. Some interesting contributions to this field are provided by refs. [115], [116] and [117]. More specifically, a method based on a classical feed-forward NN for hearth rate estimation is proposed in ref. [115], whereas a DL method for body movement compensation is investigated in ref. [116]. Finally, a contactless breathing disorder recognition system using 2.4-GHz Doppler radar and based on a linear SVM classifier is developed in ref. [117].

The real-time implementation of radar sensing methods for HGR, health monitoring and fall detection can be computationally intensive. This problem becomes more relevant in all those applications in which multiple persons have to be monitored in the same environment; in fact, in such cases, the exploitation of the MIMO technology becomes mandatory, because of the need of localising multiple agents. This explains why an important

[22]The maximum number of distinct hand gestures considered in ref. [109] is equal to 10.

technical challenge is represented by the exploitation of hardware platforms that support parallel computing (namely, FPGAs and GPUs), require a limited power consumption and can manage a large data rate at their inputs [10].

D. Autonomous driving

Automotive radar represents one of the key enabling technologies for autonomous driving. The typical processing chain employed for target detection in a MIMO FMCW radar system for automotive applications is represented in Fig. 23. The signals acquired through multiple receive antennas undergo multidimensional FFT processing;

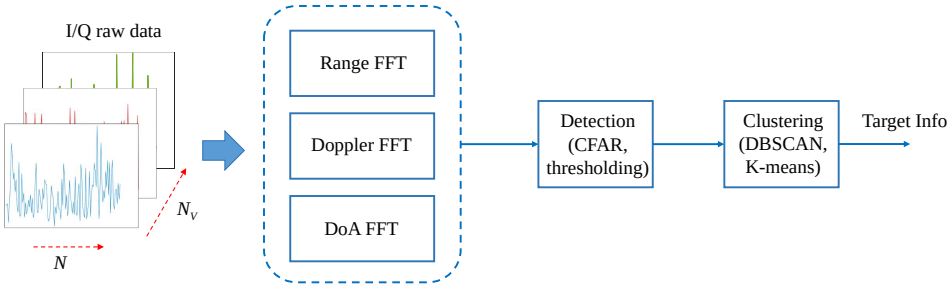


Figure 23: Typical processing chain employed in automotive radar systems [4].

this allows to extract range, Doppler and DoA information. The data generated by the FFT blocks are processed by a detection algorithm, whose objective is identifying the presence of multiple extended targets, and estimating their spatial coordinates and their radial velocity (i.e., the Doppler shift characterizing them). Each of the detected targets (e.g., pedestrians, cars or bicycles) usually appears as a cloud of point targets; the association of each point to a given extended target is called *clustering*. The simplest unsupervised strategy that can be adopted for target detection is *thresholding*; in this case, a target is detected if the amplitude spectrum of the acquired signals exceeds a fixed threshold, as illustrated in Paragraphs III-A and III-B4 for specific FMCW radar systems. A more refined alternative is represented by the *constant false alarm rate* (CFAR) technique [132]. This method consists in estimating the level of interference in each cell in the range domain of interest and in exploiting these information for the detection of the presence of a target in each cell of a radar image. Unluckily, due to the high resolution achieved by automotive radars, a single target can occupy multiple adjacent cells; when this occurs, the CFAR technique undergoes performance degradation because of the contamination affecting the estimated interference level. Clustering techniques rely on the key idea that each cluster of points is a region containing a group of detected targets, whose center typically corresponds to the point target characterized by the strongest reflectivity (see Par. III-C). This means that each cluster has a density (in terms of targets per region) which is considerably larger than that outside it; for this reason, a given point is expected to be part of a cluster if the number of its neighbours is greater than a proper threshold. Learning methods for unsupervised clustering include the *density based clustering algorithm* (DBSCAN) [118], [119], and the K-means algorithm [120]. The main difference between these two methods consists in the fact that the former method, unlike the latter one, does not require prior knowledge of the number of clusters and their shape.

All the techniques described above (namely, thresholding, CFAR and clustering) allow to detect multiple point targets and to cluster them. In general, learning methods can be adopted to improve detection performance. A number of technical problems have been identified in this area; most of them require the development of sophisticated signal processing algorithms. Specific contributions about the use of ML methods in target detection can be found in refs. [121]–[123]. In particular, a K-NN classifier is proposed as an alternative to robust CFAR detection in ref. [121], whereas the use of the SVM and PCA techniques for improving angular resolution is investigated in ref. [122]. The use of DL methods for target classification in a 2D space, instead, have been studied in refs. [15], [124]–[131]. It is worth mentioning that, in the technical literature, the first results about the use of DL methods in automotive radar systems appeared after 2015, when it was found that DCNNs were able to simultaneously detect, localize and classify multiple targets by simply analysing 2D range-azimuth (or range-Doppler) maps. Networks originally developed for computer vision applications, like *AlexNet* [133] or *ImageNet* [134], have inspired the architecture of various networks for automatically extracting features from automotive radar images [15], [124], [125]. Despite this, the CNNs usually devised for automotive applications are not as deep as those employed in computer vision. This difference is mainly due to the fact that: a) the information provided by range-azimuth or range-Doppler maps are not as rich as traditional RGB images; b) the employed inference procedure has to be as fast as possible [126]. These ideas are exemplified by the CNN proposed in ref. [127] for the classification of automotive targets, like motorcycles, cars, bicycles and pedestrians;

its architecture, illustrated in ref. [127, Fig. 2], consists of three convolutional layers and filters of size 3×3 (whose depths are equal to 32, 64 and 128, respectively). Moreover, each convolutional layer is followed by a 2×2 average-pooling layer, two fully-connected layers and a softmax layer, which is used at the end for classification. A relevant novelty introduced in this work (and in ref. [128] too) consists in considering a certain *region of interest* (ROI) around the desired targets in the analysed scene as prior information to be used during training, in order to improve the learning procedure.

Deep learning methods can be also employed to solve the problem of *scene understanding*, i.e. of correctly interpreting the events occurring around it (e.g., the event of a vehicle passing near a pedestrian that crosses a road). In this case, improving the prediction accuracy of the employed NN requires exploiting the information contained in the frames preceding and following the frame under test because of the high variability of the data provided by MIMO radar systems. An architecture based on the cascade of a LSTM module with a CNN has been proposed in ref. [129]; this exploits the temporal information provided by radar signals and is able to capture the dynamics of the surrounding scene.

Finally, it is worth mentioning that learning methods can be also employed to detect the fatigue of the driver's eyes [123] and to mitigate the interference orginating from the transmission of multiple MIMO radars in the same area. In general, the interference affecting a MIMO radar system can be due to the system itself (*self-interference*) or from other radar systems placed on the same vehicle or on other vehicles (*cross or mutual interference*); in both cases, this phenomenon results in an increase of the observed noise floor and, consequently, affects the detectability of targets. The use of RNNs for interference mitigation has been investigated in refs. [130] and [131].

VII. Current Trends in Research on MIMO Radars

In this section, a short description of three research trends in the field of DL techniques for MIMO radars is provided. More specifically, we first focus on *transfer learning*, and recent DL methods for object detection and classification. Then, we discuss the role that *explainable artificial intelligence* (XAI) may play in the radar field.

A. Transfer learning

The minimization procedure accomplished by a deep NN trained from scratch (through random initialization) may lead to a local minimum which is far from the globally optimal solution if the involved cost function is highly non-convex. Moreover, if the dataset employed in network training is not large enough, the risk of over-fitting is quite high. These problems are likely to arise in radar applications. When this occurs, *transfer learning* could represent a tool to solve them; in fact, this method often allows to achieve a good generalization capability even if the available dataset is limited [135], [136]. Transfer learning is based on the idea of exploiting the knowledge gained from a different domain to solve other related classification problems. Two approaches to the exploitation of this method in radar applications have been recently proposed. The first approach, developed for the classification of human activities, is based on training an *unsupervised network*, characterized by an encoder-decoder structure and employed to learn specific patterns appearing in the available dataset [137]. When the decoder becomes able to reconstruct the input data with a reasonable accuracy, it is removed, and fully connected and softmax layers are added in cascade to the associated encoder. Finally, the resulting network is trained in a supervised manner with a smaller, but labelled dataset: this procedure is called *fine-tuning*.

The second approach is based on the architecture represented in Fig. 24 and developed in ref. [138]. In this case, a DCNN network trained on a large dataset of RGB images is combined with fully connected and softmax layers initialized from scratch; this results in a new network, which is fine-tuned on a small dataset.

The decision about which type of transfer learning has to be preferred is based on the size of the available dataset and on the similarity of the last dataset with the one used for pre-training the selected network architecture. It has been shown that the final score of a DCNN-based classifier can be improved either by exploiting a pre-training procedure based on a simulated radar dataset [139] or by employing a pre-trained DCNN on a separate large scale RGB dataset [138].

B. Object detection and classification

The aims of object detection and classification are the labelling of all the objects appearing in a given image and the generation of a bounding box identifying their position. The *fast* R-CNN [140] and *faster* R-CNN [141] are examples of *region-based* CNNs for object detection based on bounding boxes. Another relevant solution of this type is represented by the solo called *You only look once* (YOLO) network[23]. When building up a dataset for

[23]The name of this network has been inspired by the human ability of looking once at an image and instantly recognizing the objects it contains.

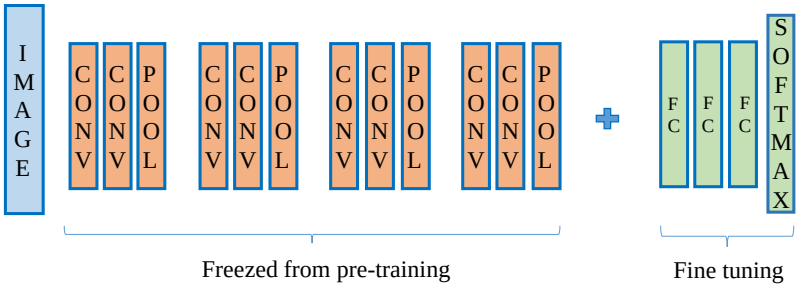


Figure 24: Architecture of a DCNN pre-trained on dataset of RGB images and fine-tuned on a small dataset of radar images.

training this network, each detectable target is bounded with a box characterized by specific size and position in the whole image. If an object detection problem in which different targets can be associated to several (say, K) classes is considered, the YOLO network should be preferred to the other methods mentioned above because of: a) its ability to predict not only the size and the position of the bounding box associated with a given target, but also the probability that the target inside a given box belongs to a certain class; b) its architecture which, being based on a CNN, is simple and fast; c) its ability to learn very general representations of objects. The results illustrated in ref. [142] for various applications evidence that a YOLO network outperforms a R-CNN in terms of detection ability, since it produces a lower number of false negatives. However, it is important to remember that a YOLO network usually makes a significant number of localization errors and, consequently, achieves a limited accuracy. Better results are obtained if an improved architecture, known as YOLO v2 and originally proposed in ref. [143], is adopted. This new version of the YOLO network is still based on a convolutional architecture, but employs *anchor boxes*[24] in predicting the position of objects. The use of anchor boxes makes the learning procedure easy, since the network has only to adjust and refine their size in order to fit an object detected in the processed image. A specific application of the YOLO v2 network to a MIMO radar system is illustrated in Paragraph VIII-B.

A recent research topic in the field of target detection and classification is represented by the use of *semantic segmentation*, that represents a powerful technique adopted for classifying the pixels of an image (a fixed set of classes is assumed in this case). The state of the art in semantic segmentation for image processing is represented by: a) *fully convolutional networks* (FCNs) [144], in which a convolutional network endowed with a pixel classification layer (instead of a fully connected layer) is used; b) SegNet [145] and U-Net [146], both based on a symmetrical encoder-decoder architecture. A more complicated method is represented by *instance segmentation*, whose aim is not only detecting and classifying all the objects appearing in an image, but also generating the segmentation of each instance appearing in the bounding box associated with each detected object. To accomplish the last task, the Facebook AI research group has proposed a new method called, *Mask*-R-CNN, that extends a Faster R-CNN by adding a branch for the prediction of the segmentation mask in each ROI [147].

It is important to note that the application of the above mentioned DL techniques to object detection and localization in radar images is still at an early stage. Despite this, specific DL methods inspired by FCNs and U-Net have been already implemented for detecting and estimating the position of different targets (like cars and other automotive targets) on the basis of range-Doppler-azimuth radar maps [148]–[150]. Moreover, the use of semantic segmentation in the radar field has been already investigated for the classification and localization of 3D point clouds of automotive targets, like cars, tractors and pedestrians; various results referring to automotive MIMO radars that operate at 77 GHz can be found in refs. [151] and [152]. The experimental results shown in these manuscripts evidence that the performance of the NNs employed for semantic segmentation substantially improves if radar data are fused with those one provided by optical sensors. It should not be forgotten that radar information can be augmented by an highly dense point cloud generated by a lidar device and that lidar data can be replaced by radar data in case of adverse weather or lighting conditions. An example of radar-centric automotive dataset based on radar, lidar and camera data for is described in ref. [153]; this dataset has been exploited in ref. [154] to test DL algorithms for 3D object detection.

C. Explainable artificial intelligence

Neural networks and sophisticated decision methods are currently employed in a number of applications to solve complicated tasks. The requirement of *transparency* is becoming more and more important in AI, especially when it is employed in autonomous systems. Unluckily, understanding which features are evaluated

[24] Anchor boxes are a set of predefined bounding boxes having certain height and width.

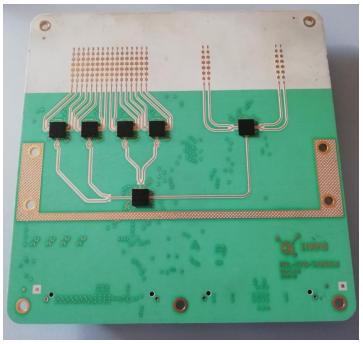
a)  b) 

Figure 25: a) Colocated MIMO radar system and b) *pico-flexx camera* employed in our experiments.

by a DNN in taking its decision is a complicated problem. Explainable artificial intelligence is a new branch of AI and concerns the problem of how the effectiveness of a deep network can be guaranteed [155]. An interesting method to improve the transparency of a DNN is based on the visualization of the features learned by each layer of the network [156]. The first layers of a DCNNs tested on radar images typically learn basic features, that depend on the size of their convolution filters. In fact, large (small) filters memorize general shapes (more specific properties), whereas some filters are also able to learn noise and clutter [105]. An alternative method to get some insight on the learning process of a CNN is based on the idea of identifying the parts of a radar image that are relevant for the classification of the object under test; such parts are also known as *spatial supports*. This approach allows to assess if a specific network is robust in taking its decision on the basis of a correct analysis of the given image. A specific technique, called *saliency extraction*, is based on this idea and, in particular, on the evaluation of the so called *saliency map*, as illustrated in ref. [157].

VIII. Experimental Results

In this section we show how specific ML and DL methods can be employed in a commercial colocated MIMO radar system to: a) classify three different human activities; b) estimate the range and DoA (azimuth) of a single target in a 2D propagation scenario. In both case, such methods are compared, in terms of accuracy and processing time; moreover, in case b), a comparison with deterministic methods is also made.

It is worth stressing that, unlike the previous sections, the results illustrated below do not originate from a synthetically generated dataset. In fact, the following tools have been exploited to generate them:

1) A colocated FMCW MIMO radar manufactured by *Inras GmbH* [158]. This radar device, shown in Fig. 25-a) and employed to acquire all our measurements, operates in the E-band (the center frequency of its transmitted signal is $f_0 = 77$ GHz) and is equipped with a TX ULA and an RX ULA, consisting of $N_T = 2$ and $N_R = 16$ antennas, respectively (see Fig. 26-a)); even if, in principle, $2 \cdot 16 = 32$ virtual channels are available (see Paragraph II-C), only $N_V = 31$ of them are exploited in our work, since two elements of the virtual array overlap.
2) A *pico-flexx camera* manufactured by *PMD Technologies Inc.* [159]. This time-of-flight camera, shown in Fig. 25-b) and employed as a reference sensor in our experiments, is based on a near-infrared vertical cavity surface emitting laser, and is able to provide a depth map or, equivalently, a three-dimensional point-cloud of a small region of the observed environment (its maximum depth is equal to 4 m, whereas its FOV is $62° \times 45°$).
3) A desktop computer equipped with a single i7 processor. All our software has been developed in the MATLAB and/or Python environment and run on this computer.

In the following two paragraphs, we provide various details about the experiments accomplished for the two specific applications mentioned above and illustrate the most relevant results we obtained.

A. Human activity classification

Our first experiment concerns the classification of following three different human activities: *walking*, *running* and *jumping*. The following choices have been made in the acquisition of our measurements:

1) The person whose activity has to be classified is alone and is in front of the employed radar device.
2) A single pair of TX-RX antennas is used (since angular information is not required).
3) The transmitted waveform is characterized by the following parameters: $N_c = 128$, $T = 128$ μs, $T_R = 32$ μs and $B = 1$ GHz (consequently, $\mu = 7.8 \cdot 10^{12}$ GHz/s; see eq. (4)).

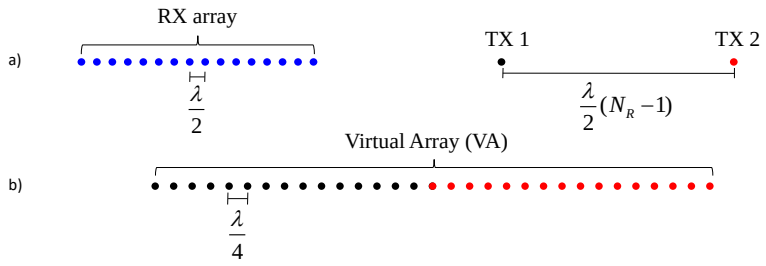


Figure 26: Geometry of a) the physical TX and RX arrays and b) the corresponding virtual array of the radar device shown in Fig. 25-a).

4) At the receive side, analog-to-digital conversion is accomplished at the sampling frequency $f_s = 80$ MHz and $N = 1024$ samples are acquired over each chirp period and an oversampling factor $M_r = 4$ is considered for our processing.

Different classification methods have been tested for this application. First of all, we took into consideration the following five ML methods: a linear SVM technique, the K-NN technique (with $K = 4$), an Adaboost classifier with decision stumps as weak learners (see Paragraph III-B4), a *customised* double stage SVM binary classifier (CSVM) and a specific version of the Adaboost, called *Stagewise Additive Modeling using a Multi-class Exponential loss function* (SAMME) [160]. As far as DL methods are concerned, we have taken into consideration a specific CNN only, since, as shown below, the preprocessed data feeding it can be interpreted as 2D images.

All these methods are fed by the matrices $\mathbf{E}$ and $\mathbf{G}$ defined at the end of Paragraph II-C (see eqs. (31) and (32)) and whose sizes are $N_f \times N_0'$ and $N_f' \times N_0'$, respectively (in all the experiments made for the considered application, $N_f = 143$, $N_0' = 256$ and $N_f' = 512$ have been selected). It is also worth remembering that the former (the latter) matrix is used to generate the spectrogram (the CVD) of the received signal. Examples of the spectrograms associated with the three possible activities are shown in Figs. 27-(a), -(b) and -(c) (note that the same time scale is used in all these figures), whereas an example of CVD is illustrated in Fig. 28. Moreover, in the last figure, two additional plots, one referring to the cadence frequency of the observed motion (left), the other one to its velocity (bottom), are also given for completeness. From Figs. 27-28 it is easily inferred that:

a) The period of the spectrogram (i.e., the distance between its consecutive peaks) is inversely proportional to the speed of the observed motion.

b) The shape of the spectrogram is influenced by the type of motion.

c) The CVD diagram contains important information regarding the motion and it is strictly related to the shape of the spectrogram. In fact, the principal components characterizing the observed motion can be identified in the CVD diagram in correspondence of the so-called *cadence frequencies*; each of these frequencies indicates how frequently a specific velocity component repeats in the observation interval.

An experimental campaign has been accomplished to build up an experimental dataset, that collects $N_t = 150$ observations equally divided among the three classes. Each observation refers to N_f consecutive frames, each consisting of N_c chirps, and is acquired over an observation interval whose duration is $T_O = 3$ s (each frame lasts $T_F = T_O/N_f = 21$ ms). Moreover, the q-th entry of the dataset $\mathcal{D}_o$ processed by the above mentioned ML methods is represented by the couple $(\mathbf{r}_q, t_q)$ (see eq. (74)), where

$$\mathbf{r}_q = [r_{q,0}, r_{q,1}, r_{q,2}, r_{q,3}] \tag{227}$$

is a 4D feature vector (so that $D_r = 4$) and t_q is a integer label identifying the specific activity which the vector $\mathbf{r}_q$ is associated with ($t_q = 0$, 1 and 2 if the observed person is walking, running or jumping, respectively). The first three elements of the vector $\mathbf{r}_q$ (227) depend on the value $\mathbf{G}_q = [G_{l,m}^{(q)}]$ of the matrix $\mathbf{G}$ computed for the q-th observation, since

$$r_{q,0} \triangleq \frac{\hat{l}_q}{N_f' \, T_F}, \tag{228}$$

$$r_{q,1} \triangleq \frac{1}{N_0'} \sum_{m=0}^{N_0'-1} \left(G_{\hat{l}_{q,0},m}^{(q)} - \mu_0^{(q)} \right) \cdot \left(G_{\hat{l}_{q,1},m}^{(q)} - \mu_1^{(q)} \right) \tag{229}$$

and

$$r_{q,2} \triangleq \frac{1}{N_0'} \sum_{m=0}^{N_0'-1} \left(\left(G_{\hat{l}_{q,0},m}^{(q)} \right)^2 + \left(G_{\hat{l}_{q,1},m}^{(q)} \right)^2 \right); \tag{230}$$

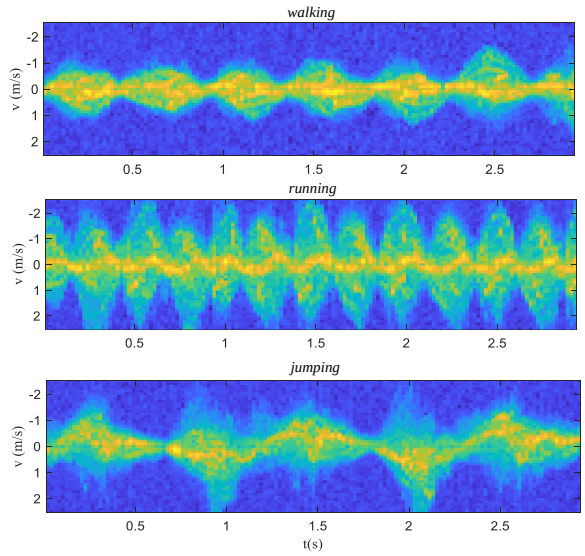


Figure 27: Spectrograms observed for the following three different activities: walking (top), running (center) and jumping (bottom).

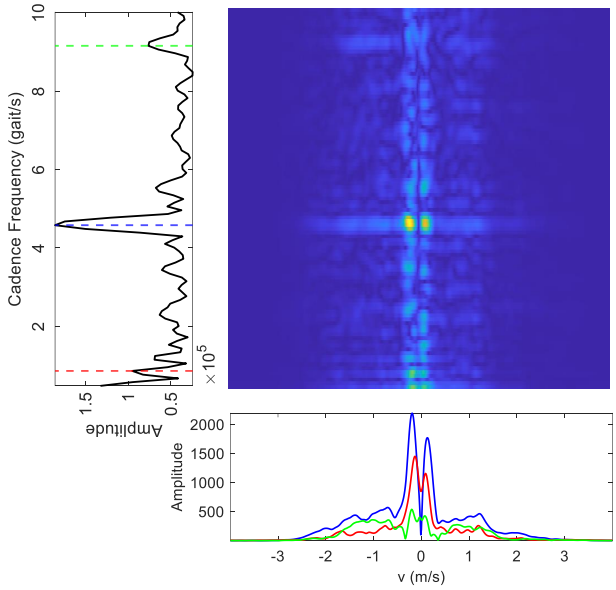


Figure 28: Representation of a CVD and of two diagrams extracted from it (one providing information about cadence frequencies, the other one about velocities). In the diagram appearing on the left, the three strongest frequency components are identified by blue, red and green dashed lines; each line is associated with the velocity profile shown in the other diagram and having the same colour.

here,

$$\hat{l}_q \triangleq \arg \max_{l \in \{0,1,\ldots,N'_f-1\}} V_l^{(q)}, \tag{231}$$

$\hat{l}_{q,k}$ is the index identifying the k-th largest peak appearing in the sequence $\{V_l^{(q)};\ l = 0,\ 1,\ ...,\ N'_f - 1\}$ (with $k = 0$ and 1),

$$V_l^{(q)} \triangleq \sum_{m=0}^{N'_0-1} G_{l,m}^{(q)}, \tag{232}$$

and $\mu_k^{(q)}$ is the mean of the elements of the $\hat{l}_{q,k}$-th row of the matrix $\mathbf{G}_q$, i.e. of the vector

$$\mathbf{G}_{\hat{l}_{q,k}}^{(q)} \triangleq \left[G_{\hat{l}_{q,k},0}^{(q)}, G_{\hat{l}_{q,k},1}^{(q)}, ..., G_{\hat{l}_{q,k},N'_0-1}^{(q)} \right]^T, \tag{233}$$

with $k = 0$ and 1. The last feature of $\mathbf{r}_q$ (227) (namely, the quantity $r_{q,3}$) depends on the value $\mathbf{E}_q$ of the matrix $\mathbf{E}$ computed for the q-th observation, since it represents the period of the spectrogram, i.e. the distance between two consecutive peaks observed along the time dimension. It is important to point out that:

1) The parameter $r_{q,0}$ (228) represents the strongest frequency component detected in the CVD diagram (see Fig. 28). The value of this parameter is expected to increase with the speed of the observed person.
2) The parameters $\hat{l}_{q,0}$ and $\hat{l}_{q,1}$ identify the two strongest frequencies (denoted $\hat{f}_{q,0}$ and $\hat{f}_{q,1}$, respectively) detected in the CVD referring to the q-th observation; such frequencies are evaluated as

$$\hat{f}_{q,k} = \frac{\hat{l}_{q,k}}{N'_f \, T_F}, \tag{234}$$

with $k = 0$ and 1.
3) The parameter $\mu_k^{(q)}$ is the mean of the *velocity profile* expressed by the N'_0-dimensional vector $\mathbf{G}^{(q)}_{\hat{l}_{q,k}}$ (233).
4) The parameter $r_{q,1}$ (229) represents the covariance between the velocity profiles $\mathbf{G}^{(q)}_{\hat{l}_{q,0}}$ and $\mathbf{G}^{(q)}_{\hat{l}_{q,1}}$, whereas $r_{q,2}$ is the overall energy associated with both profiles; our experimental data have evidenced that the value of $r_{q,1}$ ($r_{q,2}$) decreases (increases) as the speed of the observed person gets larger (smaller).
5) The value of $r_{q,3}$ is inversely proportional to the speed of the observed person, since an increase of the speed shortens the period of the spectrogram.

As far as the adopted ML methods are concerned, the following choices have been made:

a) The K-NN classifier is structured as illustrated in Paragraph III-B4.

b) The classifiers based on the SVM and the Adaboost methods exploit the pairwise classification approach illustrated at the end of Paragraph III-B4. For this reason, they combine $L = K(K-1)/2 = 3$ *identical* binary classifiers (i.e., base learners).

c) The CSVM method is obtained by cascading two linear SVM binary classifiers (whose behaviour is described in Paragraph III-B4). The first SVM classifier (denoted SVM #1) distinguishes jumping from the rest of the activities and is fed by the feature vector $\mathbf{r}'_q = [r_{q,0}\,,\, r_{q,1}]$ (in this case, the scalar labels $t'_q = 1$ and $t'_q = -1$ are associated with jumping, and with walking and running, respectively). The second classifier (SVM #2) processes the observations related to running and walking only and is fed by the feature vector $\mathbf{r}''_q = [r_{q,2}\,,\, r_{q,3}]$ (in this case, the scalar labels $t''_q = 1$ and $t''_q = -1$ are associated with walking and running, respectively). The final predictions of the CSVM are generated on the basis of the SVM #1 (SVM #2) predictions for jumping (running and walking).

d) The employed version of the SAMME method is the one implemented in the Python library Scikit-learn [161] (namely, `sklearn.ensemble.AdaBoostClassifier`) and represents a specific version of the Adaboost technique for solving multi-class problems; in practice, it is based on a decision tree classifier characterized by two nodes (instead of a simple decision stump). This methods outperforms a classical Adaboost technique by simply emphasizing the weights assigned to misclassified points.

In our experiment, a N-fold cross-validation, with $N = 5$, has been employed. The accuracy achieved by the considered ML methods and the processing time they have required for training and prediction are listed in Table V.

From these results, it is easily inferred that:

a) The accuracy is reasonably good in all cases (slightly above the 90%).
b) The Adaboost performs marginally better than the K-NN and SVM methods, at the price of substantially larger computation time.
c) The best trade-off in terms of performance and computation time is achieved by the K-NN technique.
d) The CSVM method requires a lower computational effort (especially in training) with respect to the method based on SVM and round-robin binarization. This is mainly due to the fact that the former approach employs only two learners, whereas the latter one three binary classifiers.
e) The SAMME algorithm achieves the same accuracy as the round-robin binarization of the classic Adaboost, even if its computation time (in both training and prediction) is approximately ten times smaller.

The ML methods tested in the first part of our experiment exploit a dataset of manually extracted features (see eqs. (227)-(230)). On the contrary, the CNN employed in the second part of our experiment is able to classify human activities by recognizing specific patterns directly in the matrix $\mathbf{E}$. A description of its architecture is provided in Table VI. The first three layers of the employed network are represented by three convolutional 2D filters, having size 15×5 and depths 4, 8 and 16; moreover, each filter feeds a linear rectifier, followed by a max pooling layer. Each max pooling layer allows to halve the size of the image made available by the previous layer, so that a significant dimensionality reduction is obtained. The first three layers are followed by another 2D convolutional filter with a batch normalization layer. The last layers are represented by a *fully-connected* (FC) and a *softmax* (Soft) layer transforming the residual 2D image in a vector of size 3, since three classes

	SVM	K-NN	ADA	CSVM	SAMME
Accuracy (%)	89	90	91	90	91
Training time (s)	0.1	0.03	4.5	0.06	0.45
Prediction rime (s)	0.01	0.01	0.5	0.01	0.05

Table V: Accuracy, training time and prediction time evaluated for each of the ML methods considered for human activity classification.

Layers	Filters	Size	Stride	Output
Convolutional + ReLu	4	15×5	1	$143 \times 53 \times 4$
Max pooling	-	15×5	2	$65 \times 25 \times 4$
Convolutional + ReLu	8	15×5	1	$65 \times 25 \times 8$
Max pooling	-	15×5	2	$26 \times 11 \times 8$
Convolutional + ReLu	16	15×5	1	$26 \times 11 \times 16$
Max pooling	-	3×3	2	$12 \times 5 \times 16$
Convolutional + BN + ReLu	3	3×3	1	$12 \times 5 \times 3$
FC + Soft	3	-	-	$1 \times 1 \times 3$

Table VI: Architecture of the CNN employed for the classification of three human activities.

are considered. It is worth noting that the adoption of a CNN having a small depth is justified by the fact that spectrograms referring to the three activities are quite different, as exemplified by Fig. 27.

The q-th entry of the dataset $\mathcal{D}_o$ processed by the employed CNN is represented by the couple $(\mathbf{r}_q, t_q)$, where, however, the observation $\mathbf{r}_q$ is represented by the value $\mathbf{E}_q$ taken on by matrix $\mathbf{E}$ in the q-th acquisition (the label t_q, instead, has the same meaning as in the ML case). Moreover, $N_f = 143$ and $\bar{N}_0' = 53$, and $\hat{N}_t = 150$ are assumed for the size of the matrix $\mathbf{E}$ and for the dataset $\mathcal{D}_o$, respectively. Network training is based on 60% of the whole dataset (the remaining part of the dataset has been equally divided to generate a validation set and a test set); moreover, it has been accomplished by an SGD minimization procedure, which is characterized by a subset S of 4 training data samples, a learning rate $\gamma^{(i)} = 10^{-3}$ for any i and an overall number of epochs $N_E = 50$ (see eq. (106)). A 96% classification accuracy has been achieved in this case; therefore, the proposed DL method achieves a substantially better generalization capability that the ML counterparts described above. We should not forget, however, that this result is achieved at the price of a training time of about 25 s; this is substantially larger than that required by the considered ML methods (see Table V). Finally, it is important to mention that the computation time required by the employed CNN for evaluation a new prediction is about 0.03 s and, consequently, is reasonably short and comparable with the one characterizing the considered ML methods.

B. Estimation of the range and azimuth of a single target

The second application we have investigated concerns the detection of a specific target moving on a 2D multi-target scenario, and the estimation of its range and azimuth. In our experiment, the target to be detected is an omnidirectional reflector, obtained by putting together eight corner reflectors (and inspired by the architecture of the *echo-master* corners used for maritime applications). This target is mounted, through a vertical carton support, on a Propeller Scribbler 3 mobile robot manufactured by Parallax Inc [162]. This robot has been programmed to move randomly inside a square white region delimited by four opaque black lines and whose side is equal to 2.5 m, as shown in Fig. 29; note that two corner reflectors have placed on the borders of this region in order to build a multi-target scenario. The following choices have been made in the acquisition of our measurements:

1) The whole antenna array shown in Fig. 26-a) is exploited, so that $N_V = 31$ distinct virtual channels are available at the receive side.
2) The waveform radiated by each TX antenna is characterized by the following parameters: $N_c = 1$, $T = 64$ μs, $T_R = 32$ μs and $B = 2$ GHz (consequently, $\mu = 3.13 \cdot 10^{13}$ GHz/s; see eq. (4)).
3) At the receive side, analog-to-digital conversion is accomplished at the sampling frequency $f_s = 40$ MHz and $N = 2048$ samples are acquired over each chirp period.

4) The reference position of the target with respect to a three-dimensional reference system is evaluated by means of the pico-flexx camera. This sensor is aligned with the radar system, being mounted on the same plastic support of the radar device and at a fixed distance from it (about 10 cm) along the vertical direction.

The following two supervised DL methods have been tested: a) a feed-forward NN exploiting some manually extracted features (further details about this method are provided below); b) a YOLO v2 NN for object detection (see Paragraph VII-B).

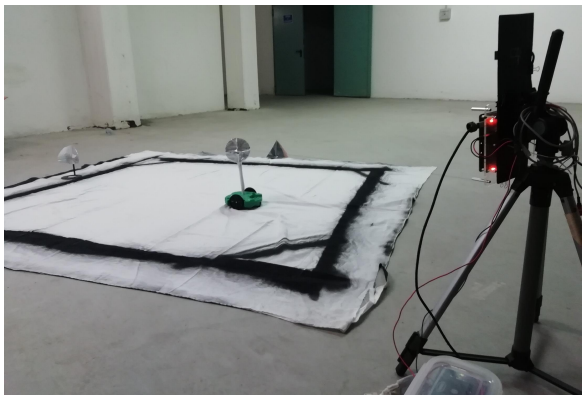


Figure 29: Experimental-setup developed for our second application. The region of interest is delimited by an opaque and black line; two coner reflectors are located on its border. A robot, equipped with corner reflectors, moves randomly inside that area. The employed radar system and pico-flexx camera are placed on the tripod visible on the right.

The q-th entry $(\mathbf{r}_q, \mathbf{t}_q)$ of the dataset $\mathcal{D}_o$ processed by the employed feed-forward NN is generated as follows. The label $\mathbf{t}_q$ associated with the q-th observation $\mathbf{r}_q$ is defined as

$$\mathbf{t}_q \triangleq [\hat{R}_q, \hat{\phi}_q] \tag{235}$$

where $\hat{R}_q$ and $\hat{\phi}_q$ represent the estimates of the target range R_q and azimuth ϕ_q, respectively, evaluated on the basis of the point-cloud made available by our pico-flexx camera. Such a camera generates the $N_p \times 3$ matrix

$$\mathbf{P} \triangleq [\mathbf{x}\ \mathbf{y}\ \mathbf{z}], \tag{236}$$

collecting the 3D coordinates of $N_p = 38304$ distinct points; here, $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ are N_p-dimensional column vectors. The deterministic algorithm developed for the estimation of the target range and azimuth involves the computation of the estimates $(\hat{x}_q, \hat{y}_q, \hat{z}_q)$ of the target coordinates (x_q, y_q, z_q) in the q-th observation; note that z_q (i.e., the target height) is assumed to be approximately known ($z_q \cong 0.4$ m). This algorithm consists of the following three consecutive steps:

1) The size of the search space for the couple $(\hat{x}_q, \hat{y}_q)$ is reduced by extracting the set[25]

$$\mathcal{S}_q \triangleq \{(x_{q,n}, y_{q,n}, z_{q,n}) | z_{\min} \leq z_{q,n} \leq z_{\max}; n \in \Delta_q\}, \tag{237}$$

from the matrix $\mathbf{P}$ (236); here, Δ_q is a proper subset of the set of integers $\{0, 1, ..., N_p - 1\}$ and consists of $\bar{N}_q$ elements, whereas $z_{\min} = 0.3$ m and $z_{\max} = 0.5$ m represent two thresholds.

2) The estimates

$$\hat{x}_q = 1/\bar{N}_q \sum_{n=0}^{\bar{N}_q - 1} x_{q,n}, \tag{238}$$

$$\hat{y}_q = 1/\bar{N}_q \sum_{n=0}^{\bar{N}_q - 1} y_{q,n} \tag{239}$$

and

$$\hat{z}_q = 1/\bar{N}_q \sum_{n=0}^{\bar{N}_q - 1} z_{q,n} \tag{240}$$

are computed. The estimate $\hat{z}_q$ (240) is exploited only to check if the vector $(\hat{x}_q, \hat{y}_q, \hat{z}_q)$ is meaningful, i.e. if the condition $\hat{z}_q \approx 0.4$ m is satisfied; if this does not occur, the thresholds appearing in the RHS

[25]Note that, in this step, our prior knowledge about the target height is exploited.

of eq. (237) should be properly adjusted (i.e., $z_{\min}$ should be increased and/or $z_{\max}$ reduced) in order to improve the obtained accuracy.

3) The estimates

$$\hat{R}_q = \sqrt{\hat{x}_q + \hat{y}_q} \tag{241}$$

and

$$\hat{\phi}_q = \arctan\ (\hat{y}_q / \hat{x}_q) \tag{242}$$

are evaluated.

The observation $\mathbf{r}_q$ labelled by $\mathbf{t}_q$ (235) is defined as[26]

$$\mathbf{r}_q \triangleq \left[\hat{\psi}_{q,0}, \hat{\psi}_{q,1}, ..., \hat{\psi}_{q,N_V-1}, \hat{f}_q\right]^T, \tag{243}$$

where $\hat{f}_q$ is the frequency associated with the detected target (and estimated on the whole array) and $\hat{\psi}_{q,v}$ is the phase of the signal spectrum computed at the frequency $\hat{f}_q$ for the v-th virtual element (with $v = 0, 1, ..., N_V - 1$); note that the size of the vector $\mathbf{r}_q$ (243) is $D_r = N_V + 1 = 32$. The deterministic algorithm employed for the computation of the frequency $\hat{f}_q$ and the phases $\{\hat{\psi}_{q,v}\}$ forming $\mathbf{r}_q$ (243) consists of the following two steps:

1) *Coarse estimation of the target position* - The N-dimensional vector of the time domain samples acquired over the v-th virtual antenna (see eq. (51)) undergoes zero padding and FFT processing of order $N_0 = N \cdot M_r$ (in our experiment, $N_0 = 8192$, since $M_r = 4$). This produces the N_0-dimensional vector $\mathbf{X}_v^{(q)}$ (see eq. (53)), which is employed to compute the *power spectrum* $\mathbf{P}_v^{(q)} = [P_{v,0}^{(q)}, P_{v,1}^{(q)}, ..., P_{v,N_0-1}^{(q)}]$ on the basis of eq. (56). Then, given (see eq. (58))

$$\hat{l}_v^{(q)} \triangleq \arg \max_{\tilde{l} \in \{b_m, ..., b_M\}} P_{v,\tilde{l}}^{(q)}, \tag{244}$$

a target is detected on the v-th antenna if $P_{v,\hat{l}_v^{(q)}}^{(q)} > P_d$, where P_d is a proper threshold; here, the integer parameter b_m (b_M) identifies the frequency bin corresponding to the minimum (maximum) measurable range R_m (R_M). In our experiment, $P_d = 0.9$, and

$$b_m = \left\lfloor \frac{2\mu N_0 T_s R_m}{c} \right\rfloor = 42 \tag{245}$$

and

$$b_M = \left\lfloor \frac{2\mu N_0 T_s R_M}{c} \right\rfloor = 147, \tag{246}$$

since $R_m = 1.0$ m and $R_M = 3.5$ m have been assumed. The procedure illustrated above is accomplished for each virtual channel (i.e., for $v = 0, 1, ..., N_V - 1$) and is employed to generate the set

$$\mathcal{S}_{\hat{l}} \triangleq \{\hat{l}_{v_k}^{(q)}; k = 0, 1, ..., \bar{N}_V - 1\}, \tag{247}$$

with $v_k < v_{k+1}$ for any k; the size $\bar{N}_V$ of this set is usually smaller that N_V, since: a) the target may be missed on one or more virtual channels (this occurs when the condition (244) is not satisfied); b) the elements of $\mathcal{S}_{\hat{l}}$ are required to be distinct. The elements of $\mathcal{S}_{\hat{l}}$ are collected in the vector $\hat{\mathbf{l}}_q = [\hat{l}_{v_0}^{(q)}, \hat{l}_{v_1}^{(q)}, ..., \hat{l}_{v_{\bar{N}_V-1}}^{(q)}]^T$. Then, the following vectors are computed: a) the $\bar{N}_V$-dimensional vector $\hat{\mathbf{f}}_q = [\hat{f}_{v_0}^{(q)}, \hat{f}_{v_1}^{(q)}, ..., \hat{f}_{v_{\bar{N}_V-1}}^{(q)}]^T$ and $\hat{\mathbf{R}}_q = [\hat{R}_{v_0}^{(q)}, \hat{R}_{v_1}^{(q)}, ..., \hat{R}_{v_{\bar{N}_V-1}}^{(q)}]^T$, that collect $\bar{N}_V$ estimates of the target frequency and range, respectively (these quantities computed on the basis of eqs. (60) and (62), respectively); b) the set of $\bar{N}_V$ vectors $\{\hat{\mathbf{A}}_{v_k}^{(q)}\ ; k = 0, 1, ..., \bar{N}_V - 1\}$, where $\hat{\mathbf{A}}_{v_k}^{(q)} = [\hat{A}_{v_k,0}^{(q)}, \hat{A}_{v_k,1}^{(q)}, ..., \hat{A}_{v_k,N_V-1}^{(q)}]^T$ is made of the complex amplitudes evaluated over the whole virtual array on the basis of eq. (61) under the assumption that $\hat{l} = \hat{l}_{v_k}^{(q)}$ for any k; d) the set of $\bar{N}_V$ vectors $\{\hat{\boldsymbol{\psi}}_{v_k}^{(q)}; k = 0, 1 , ..., \bar{N}_V - 1\}$, where

$$\hat{\boldsymbol{\psi}}_{v_k}^{(q)} = [\hat{\psi}_{v_k,0}^{(q)}, \hat{\psi}_{v_k,1}^{(q)}, ..., \hat{\psi}_{v_k,N_V-1}^{(q)}]^T \tag{248}$$

and $\hat{\psi}_{v_k,l}^{(q)}$ is equal to the phase of the complex gain $\hat{A}_{v_k,l}^{(q)}$ for any k and l (see eq. (40)). Finally, each of the vectors $\{\hat{\mathbf{A}}_{v_k}^{(q)}\}$ undergoes zero padding, that increases their size to $\bar{N}_0 = 128$, and $\bar{N}_0$-th order FFT processing for azimuth estimation (see eqs. (70)-(72)). This produces the vector $\hat{\boldsymbol{\phi}}_q = [\hat{\phi}_{v_0}^{(q)}, \hat{\phi}_{v_1}^{(q)}, ..., \hat{\phi}_{v_{\bar{N}_V-1}}^{(q)}]^T$, collecting $\bar{N}_V$ different estimates of the target azimuth. Therefore, this step produces $\bar{N}_V$

[26] *Unwrapped* phases are employed in this case, since they ease network training

distinct estimates $\{(\hat{f}_{v_k}^{(q)}, \hat{R}_{v_k}^{(q)}, \hat{\phi}_{v_k}^{(q)}); k = 0, 1, ..., \bar{N}_V - 1\}$ of the target frequency, range and azimuth, respectively.

2) *Fine estimation of the target position* - A *single* estimate of the target frequency, range and azimuth is evaluated in this step on the basis of the $\bar{N}_V$ estimates $\{(\hat{f}_{v_k}^{(q)}, \hat{R}_{v_k}^{(q)}, \hat{\phi}_{v_k}^{(q)})\}$ available at the end of the previous step. This estimate is computed as follows. First, we compute

$$\hat{v}_q = \min_{\tilde{l} \in \mathcal{S}_{\hat{l}}} \left| \hat{\phi}_q - \hat{\phi}_{v_{\tilde{l}}}^{(q)} \right|, \tag{249}$$

under the constraint

$$\left| \hat{R}_q - \hat{R}_{v_{\tilde{l}}}^{(q)} \right| < R_{th}, \tag{250}$$

with $R_{th} = 0.3$ m; here, the quantities $\hat{R}_q$ and $\hat{\phi}_q$ are expressed by eq. (241) and eq. (242), respectively, and $\mathcal{S}_{\hat{l}}$ is the set defined by eq. (247). Then, the vector $\mathbf{r}_q$ (243) is evaluated as

$$\mathbf{r}_q = [\hat{\boldsymbol{\psi}}_{\hat{v}_q}^{(q)}, \hat{f}_{\hat{v}_q}^{(q)}]^T \tag{251}$$

where the vector $\hat{\boldsymbol{\psi}}_{\hat{v}_q}^{(q)}$ is expressed by eq. (248) with $v_k = \hat{v}_q$.

The entire dataset is generated by accomplishing the feature selection procedure expressed by eqs. (249)-(250) for any q. It is important to stress that, in our experiment, only a specific target must be selected for each observation. In fact, in a multiple target scenario like the one we are considering, it is hard to understand which elements of the set $\mathcal{S}_{\hat{l}}$ (247) are associated with the target of interest.

The vector $\mathbf{r}_q$ (251) generated by the deterministic procedure described above represents the input of our feed-forward NN, whose response is the bidimensional vector $\hat{\mathbf{t}}_q \triangleq [\hat{t}_{q,0}, \hat{t}_{q,1}]$; the elements of this vector represent the estimates of the range and the azimuth, respectively, of the target detected on the basis of the q-th observation (see eq. (235)). This network contains three hidden layers, consisting of $M_1 = 30$, $M_2 = 20$ and $M_3 = 10$ neurons (see Fig. 13). Each of them employs a ReLu, characterized by the transfer function

$$h(x) = x\, \mathrm{u}(x), \tag{252}$$

where $\mathrm{u}(\cdot)$ denotes the *unit step function*. The estimates of the target range and azimuth are computed by the output layer, that contains two neurons only.

The size of the whole dataset acquired in our experiment is $\hat{N}_t = 1438$; 80% of it has been exploited for training the considered NN and the remaining part for its test (therefore, the size of the training set and that of the test are $N_t = 1150$ and $\bar{N}_t = 288$, respectively). Moreover, training has been accomplished by an *adam* optimizer; the batch size, the (constant) learning rate and the number of epochs selected for this procedure are $N_S = 4$, $\gamma = 10^{-3}$ and $N_E = 50$, respectively (see eq. (106)). The elements of the feature vector $\mathbf{r}_q$ (with $q = 0, 1, ..., N_t - 1$) have been scaled before applying it to the network (more specifically, a min-max normalization has been employed [163]); this ensures that the absolute value of such elements belongs to the interval $[0, 1]$ and makes the training procedure more effective. The accuracy achieved by the network over the test set has been assessed by evaluating the RMSEs

$$\hat{\varepsilon}_R = \frac{1}{\sqrt{\bar{N}_t}} \left\| \hat{\mathbf{R}} - \hat{\mathbf{R}}_{NN} \right\| \tag{253}$$

and

$$\hat{\varepsilon}_\phi = \frac{1}{\sqrt{\bar{N}_t}} \left\| \hat{\boldsymbol{\phi}} - \hat{\boldsymbol{\phi}}_{NN} \right\|, \tag{254}$$

where $\hat{\mathbf{R}}$ ($\hat{\boldsymbol{\phi}}$) is the $\bar{N}_t$-dimensional vector collecting the values of the target range (azimuth) estimated by means of the pico-flexx camera over the test set and $\hat{\mathbf{R}}_{NN}$ ($\hat{\boldsymbol{\phi}}_{NN}$) is the corresponding prediction computed by our NN ($||\boldsymbol{x}||$ denotes the Euclidean norm of the vector $\boldsymbol{x}$). The network performance has been also assessed by evaluating its *detection score*

$$A_c = \frac{N_C}{N_C + N_W}, \tag{255}$$

where N_C (N_W) is the number of trials in the test set in which both target azimuth and range have been correctly (wrongly) estimated (note that $N_C + N_W = \bar{N}_t$). In the q-th trial, estimation is deemed correct if $|\hat{R}_q - \hat{t}_{q,0}| \leq \Delta R$ and $|\hat{\phi}_q - \hat{t}_{q,1}| \leq \Delta\phi$, where $\Delta R = 20$ cm and $\Delta\phi = 5.5°$. It is worth pointing out that the values selected for the parameters ΔR and $\Delta\phi$ account for the limited resolution of the employed camera and radar system. Actually, the value selected for ΔR may look larger than expected, because of the high resolution that can be potentially achieved by both our radar device and pico-flexx camera. However, readers should not forget that the algorithm employed for the computation of $\hat{R}_q$ is not error free (see eq. (241) and

Methods	$\hat{\varepsilon}_R$ (m)	$\hat{\varepsilon}_\theta$ (°)	*AC* (%)	*Training* (sec)	*Prediction* (msec)
FFT based	0.09	3.0	88	-	5
ANN	0.07	3.5	92	8	10
YOLO v2	0.03	1.5	98	398	20

Table VII: Accuracy, detection score, training and prediction time of a deterministic estimation algorithm, a feed-forward NN and a YOLO v2 network.

(242)), especially when the cluster of points Δ_q (237) is not so dense or when the size $\bar{N}_q$ in eq. (240) is large. A low density in the set $\mathcal{S}_q$ could be observed when, for instance, the robot reaches the corners of the delimited area or in presence of optical disturbances.

The estimated accuracy and precision achieved by the adopted NN together with the time required for its training and testing are listed in Table VII. In the same table, the values of the same parameters evaluated on the basis of the deterministic algorithm employed for feature extraction are also provided; note that, for any q, this algorithm can be exploited to generate the estimates $\hat{R}_{\hat{v}_q}^{(q)}$ and $\hat{\phi}_{\hat{v}_q}^{(q)}$ of the target range and azimuth, on the basis of $\hat{v}_q$ (249) (note that $\hat{R}_{\hat{v}_q}^{(q)}$ and $\hat{\phi}_{\hat{v}_q}^{(q)}$ represent the $\hat{v}_q$-the element of the vectors $\hat{\mathbf{R}}_q$ and $\hat{\boldsymbol{\phi}}_q$, respectively). From these results it is easily inferred that: a) the NN is able to accurately predict the position of the target; b) it outperforms the deterministic algorithm in terms of both accuracy and precision; c) its prediction time is comparable with the computation time required by the deterministic algorithm.

In general, feed-forward NNs require a clever selection of their feature vector; for this reason, some expertise in radar systems is desirable when applying them to target detection and estimation. This problem can be circumvented by applying the YOLO v2 network (see Paragraph VII-B). Let us illustrate now how this network can be employed to solve the target detection and estimation problem taken into consideration in this paragraph. The q-the element of the collected dataset

$$\mathcal{D}_o \triangleq \{(\mathbf{r}_q, \mathbf{b}_q, t_q)\,; q = 0, 1, ..., \hat{N}_t - 1\} \tag{256}$$

consists of the following three components:

1) The noisy observation $\mathbf{r}_q = \mathbf{J}_q$, where $\mathbf{J}_q = [J_{l,m}^{(q)}]$ is a *range-azimuth matrix* having size $N_0 \times \bar{N}_0$ and computed on the basis of the measurements acquired in the q-th trial. The element on the l-th row and the m-th column of $\mathbf{J}_q$ is defined as

$$J_{l,m}^{(q)} \triangleq \frac{1}{N_0 \bar{N}_0} \left| \sum_{v=0}^{\bar{N}_0 - 1} \sum_{n=0}^{N_0 - 1} S_{l,m} \right|, \tag{257}$$

where

$$S_{l,m} = \hat{r}_{v,n}^{(ZP)} \exp\left(-j2\pi n\, \hat{f}_l T_s\right) \exp\left(-j2\pi v \frac{d}{\lambda} s_m\right) \tag{258}$$

with $l = 0$, 1, ..., $N_0 - 1$ and $m = 0$, 1, ..., $\bar{N}_0 - 1$; here, $\hat{r}_{v,n}^{(ZP)}$ is the n-tn element of the N_0-dimensional vector $\mathbf{r}_v^{(ZP)}$that results from zero padding N-dimensional vector of time domain signal samples acquired over the v-th virtual antenna, T_s the sampling period, $f_l \triangleq l/(N_0 T_s)$ is the center frequency of the l-th frequency bin and $s_m \triangleq 2\,(m - \bar{N}_0/2)/\bar{N}_0$ is the m-th *normalized spatial frequency*. Note that the matrix $\mathbf{J}_q$ can be computed through a $N_0 \times \bar{N}_0$-order 2D FFT, and that the range and azimuth associated with $J_{l,m}^{(q)}$ (257) are (see eqs. (62) and (72), respectively)

$$\bar{R}_l = f_l \frac{c}{2\mu} \tag{259}$$

and

$$\bar{\phi}_m = \arcsin s_m, \tag{260}$$

respectively.

2) The vector

$$\mathbf{b}_q = [l_q, m_q, w_q, h_q] \tag{261}$$

describing the bounding box associated with the detected target; here, the couple of integers (l_q, m_q) identifies the frequency bin and the normalised spatial frequency, respectively, corresponding to the center of the box and w_q (h_q) represents the width (height) of the box itself.

Layers	Filters	Size	Stride	Output
Convolutional + BN + ReLu	16	5×5	2	$52 \times 52 \times 16$
Max pooling	-	2×2	2	$26 \times 26 \times 16$
Convolutional + BN + ReLu	32	5×5	2	$12 \times 12 \times 32$
Max pooling	-	2×2	2	$6 \times 6 \times 32$
Convolutional + BN + ReLu	64	3×3	1	$6 \times 6 \times 64$
Max pooling	-	2×2	2	$3 \times 3 \times 64$
Convolutional + BN + ReLu	128	3×3	1	$3 \times 3 \times 128$
Convolutional + BN + ReLu	256	3×3	1	$3 \times 3 \times 256$
Convolutional + BN + ReLu	512	3×3	1	$3 \times 3 \times 512$
Convolutional	6	1×1	1	$1 \times 1 \times 6$

Table VIII: Architecture of the CNN employed for target detection and estimation.

3) The label t_q; this equal to 1 (-1) if a target is detected (absent).

In our experiment, we have selected $N_0 = 8192$ and $\bar{N}_0 = 128$ in the computation of the elements of the matrix $\mathbf{J}_q$. However, since $R_m = 1.0$ m ($R_M = 3.5$ m) and $\phi_m = -55^\circ$ ($\phi_M = 55^\circ$) have been assumed for the minimum (maximum) range, the $N_0 \times \bar{N}_0$ matrix $\mathbf{J}$ has been resized to an $\bar{N}_l \times \bar{N}_m$ matrix, where $\bar{N}_l = b_M - b_m + 1 = 106$ (the values of the parameters b_m and b_M are expressed by eqs. (245) and (246), respectively), $\bar{N}_m = d_M - d_m + 1 = 106$, with

$$d_m = \left\lfloor \frac{\bar{N}_0}{2} (s_m + 1) \right\rfloor = 11 \tag{262}$$

and

$$b_m = 42. \tag{263}$$

In addition, a square shape with $w_q = h_q = 12$ has been always assumed for the bounding box; its parameters l_q and m_q have been computed as

$$l_q = \arg \min_{b_m \leq \tilde{l} \leq b_M} \left| \hat{R}_q - R_{\tilde{l}} \right| \tag{264}$$

and as

$$m_q = \arg \min_{d_m \leq \tilde{m} \leq d_M} \left| \hat{\phi}_q - \phi_{\tilde{m}} \right| \tag{265}$$

where $\hat{R}_q$ ($\hat{\phi}_q$) is expressed by eq. (241) (eq. (242)) and $R_{\tilde{l}}$ ($\phi_{\tilde{m}}$) by eq. (259) (eq. (260)) with $l = \tilde{l}$ ($m = \tilde{m}$). The size of the dataset $\mathcal{D}_o$ (256) is $\hat{N}_t = 1438$; 80% of its elements are used for training and the remaining part for testing; consequently, the sizes of the training set and the test set are $N_t = 1150$ and $\bar{N}_t = 288$, respectively. *Data augmentation* has been performed on the training and test set in order to reduce network overfitting, since their sizes are not so large; this procedure consists in randomly flipping and scaling the input image and the associated box.

The architecture of the employed network is summarized in Table VIII. It consists of a cascade of 22 layers and is fed by a normalized version of the resized range-azimuth matrix generated through the procedure illustrated above and having size $\bar{N}_l \times \bar{N}_m = 106 \times 106$. Each of its first two convolutional layers has stride $S = 2$ and is followed by a max pooling layer for dimensionality reduction. The use of a *batch normalization* (BN) layer after each convolutional layer allows to avoid overfitting, since the dataset size is not so large; consequently, other forms of regularization (as dropout) are not required. The activation function at the end of each convolutional layer is ReLu (see eq. (252)). The filter depth in the last convolutional layer must be proportional to $N_A \cdot (N_{P_A} + K)$, where N_A is number of anchor boxes, N_{P_A} is the number of predictions per each anchor and K is the number of classes (see refs. [142] and [143]). Since, in our test, $N_A = 1$, $N_{P_A} = 5$ and $K = 1$ (if the background is ignored), the selected filter depth is equal to 6. A *transform* layer and an *output* layer are also included in the architecture of the adopted network. The former layer improves network stability in predicting the possible locations for the bounding box, whereas the latter one refines the estimate of the bounding box location.

If the N_K candidate boxes $\{\mathbf{b}_q[k] = [l_q[k], m_q[k], w_q[k], h_q[k]]^T;\ k = 0, 1, ..., N_K - 1\}$ (all labelled by $t_q = 1$) are identified by network, the index $\hat{k}_q$ of the bounding box

$$\hat{\mathbf{b}}_q = \left[l_q[\hat{k}_q], m_q[\hat{k}_q], w_q[\hat{k}_q], h_q[\hat{k}_q] \right] \tag{266}$$

best fitting the ground truth box is evaluated as

$$\hat{k}_q = \arg \max_{\tilde{k} \in \{0,1,\ldots,N_K-1\}} I_{\tilde{k}}^{(q)}, \tag{267}$$

where

$$I_{\tilde{k}}^{(q)} = \frac{A_{BG}^{(q)} \cap A_{BP}^{(q)}}{A_{BG}^{(q)} \cup A_{BP}^{(q)}} \tag{268}$$

is the *intersection over union* (IOU) associated with the k-th candidate box; here, $A_{BG}^{(q)}$ ($A_{BP}^{(q)}$) represents the surface of the ground truth (predicted) bounding box referring to the q-th observation. In our experiment, a target is detected if $I_{\tilde{k}}^{(q)} > I_{th}$, where $I_{th} = 0.1$ is a properly selected threshold. Once the predicted bounding box $\hat{\mathbf{b}}_q$ (266) is known, the estimate of the target range (azimuth angle) is evaluated by setting $l = l_q[\hat{k}_q]$ ($m = m_q[\hat{k}_q]$) in eq. (259) (eq. (260)); note that the values selected for the parameters l and m identify the center of the predicted bounding box.

The training procedure of the adopted network has been carried out through the SGD algorithm; a batch size $N_S = 10$, a learning rate $\gamma^{(i)} = 10^{-3}$ and a number of epochs $N_E = 25$ have been assumed (see eq. (106)). The testing procedure has evidenced that this network is able to predict the bounding boxes characterized by $I_{\tilde{k}}^{(q)} > 0.1$ over 98% of the test set. A realization of range-azimuth map associated with the matrix $\mathbf{J}$ and of the associated ground truth and the predicted bounding boxes around the detected target is illustrated in Fig. 30, where the position of the two corner reflectors placed on the border of the area of interest is also identified (see Fig. 29). These results deserve the following comments:

a) The network is able to detect the target on the basis of the value of range and azimuth obtained through the pico-flexx camera.

b) In the considered case, the IOU between the ground truth bounding box (red line) and the predicted one (green line) is quite large, being equal to 0.73. Consequently, the estimate of the position of the target (green circle) is very accurate and certainly much better than the one used as reference (red cross) (note that $|\hat{R}_q - R_{l_q[\hat{k}_q]}| = 0.001$ m and $|\hat{\phi}_q - \phi_{m_q[\hat{k}_q]}| = 0.9°$ in this case).

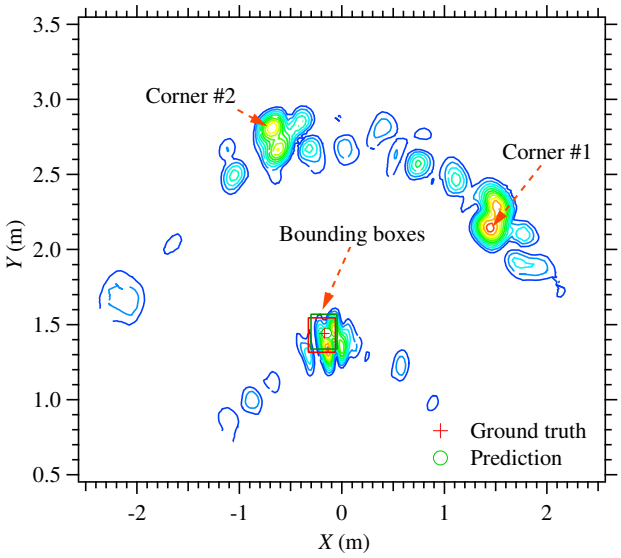


Figure 30: Range-azimuth map referring to the scenario illustrated in Fig. 29. The ground truth bounding box and the position of the target are identified by a red square and a red cross, respectively. The prediction of the network, together with the estimated bounding box, are identified by a green circle and a green square, respectively.

The values of the achieved accuracy (evaluated in terms of the RMSEs $\hat{\varepsilon}_R$ (253) and $\hat{\varepsilon}_\phi$ (254)), the detection score (255), and the computational time required for training and testing are listed in Table VII. From these results it is easily inferred that:

a) The YOLO network outperforms our (deterministic) FFT-based method in target detection and estimation.

b) The value of the YOLO detection score A_c (255) is really high and better than that provided by the feed-forward NN.

c) The YOLO RMSE $\hat{\varepsilon}_R$ ($\hat{\varepsilon}_\phi$) is smaller than (close to) the one characterizing the feed-forward NN.

These results lead to the conclusion that the YOLO network is more robust than the feed-forward NN. Note also that, even if the complexity of this network is higher than those of the other two methods, the time it employs for computing its prediction is not too long, being in the order of few milliseconds.

Since the YOLO v2 network tries to solve also a binary classification problem, other two important parameters for evaluating its performance are its *precision*

$$P = \frac{T_P}{T_P + T_N} \tag{269}$$

and its *recall*

$$R = \frac{T_P}{T_P + F_N}, \tag{270}$$

where T_P (T_N) represents the overall number of *true positives* (*true negatives*), i.e. the number of targets (false targets) classified correctly, and F_N is the overall number of false targets classified as targets. The *precision versus recall* plot evaluated in the considered experiment is shown in Fig. 31. These results lead to the conclusion that, in this case, the precision remains high for large values of the recall and drops steeply only when the recall exceeds 0.9. The area under the curve shown in Fig. 31 represents the so called *mean average precision* (mAP); in this case, we have found that $mAP = 93\%$ (note that the value of this parameter is expressed as a percentage since the precision P (269) and the recall R (270) are defined in the range $[0\,,\,1]$).

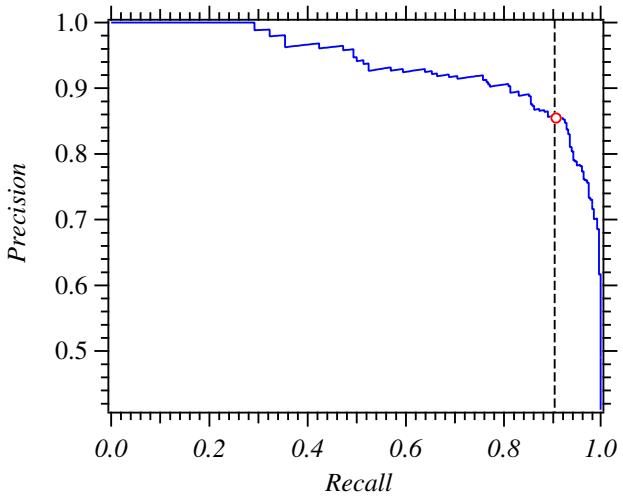


Figure 31: Representation of the precision versus recall plot referring to the YOLO v2 network employed in the second application. Note that, if the recall exceeds the threshold identified by the vertical dashed line, the precision decreases steeply.

IX. Conclusions

Thanks to recent developments of electronic technology and advances in signal processing algorithms, colocated MIMO radar systems have reached a stage of maturity that allows their adoption in a number of applications. Existing algorithms developed for target detection and estimation in radar systems equipped with antenna arrays do not always provide satisfactory performance in such applications, because of the complexity of colocated MIMO radar devices and of the propagation scenario in which they operate. This motivates the adoption of machine learning and deep learning techniques, since these are able to extract relevant information from the available data in the absence of an accurate mathematical description of the behaviour of radar devices and of the mechanisms of electromagnetic propagation. Even if important steps have been made in this field in the last years, significant research efforts are still required to make the adoption of these techniques in commercial systems a reality. In this manuscript, after providing essential information about MIMO radars and the deterministic algorithms they employ for target detection and estimation, we have shown how some learning methods can be exploited to solve simple classification and regression problems in FMCW radar systems operating in a 2D propagation scenario and in the presence of point targets. This allows readers to become familiar with some basic concepts and tools originating from the fields of radar systems and learning methods. Then, various applications of learning methods to specific technical problems have been illustrated and relevant trends in research on MIMO radars have been identified. Finally, the application of machine learning and deep learning

methods to two specific problems, namely human activity classification and range azimuth estimation, has been investigated. Our numerical results, based on experimental datasets acquired through a colocated MIMO device operating at 77 GHz, allow readers to grasp how such methods can be exploited to solve real world problems. A pervasive use of such methods should be expected in the near future, as understanding of the learning methods described in this manuscript is becoming deeper and deeper, and MIMO technology is continuously evolving.

ACKNOWLEDGEMENT

The authors would like to thank CNH Industrial Italia S.p.A. and CNH Industrial Belgium NV for funding this research work.

REFERENCES

[1] H. Van Trees, ***Optimum Array Processing – Part IV of Detection, Estimation, and Modulation Therory***. New York: Wiley, May 2002.
[2] J. Li and P. Stoica, ***MIMO Radar Signal Processing***. New York: Wiley, Mar. 2008.
[3] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Trans. Antennas Propag.*, vol. 34, no. 3, pp. 276–280, Mar. 1986.
[4] I. Bilik, O. Longman, S. Villeval, and J. Tabrikian, "The Rise of Radar for Autonomous Vehicles: Signal Processing Solutions and Future Research Directions," *IEEE Signal Process. Mag.*, vol. 36, no. 5, pp. 20–31, Sept. 2019.
[5] G. Hakobyan and B. Yang, "High-Performance Automotive Radar: A Review of Signal Processing Algorithms and Modulation Schemes," *IEEE Signal Process. Mag.*, vol. 36, no. 5, pp. 32–44, Sept. 2019.
[6] E. Sirignano, A. Davoli, G. M. Vitetta, and F. Viappiani, "A Comparative Analysis of Deterministic Detection and Estimation Techniques for MIMO SFCW Radars," *IEEE Access*, vol. 7, pp. 129 848–129 861, 2019.
[7] F. Engels, P. Heidenreich, A. M. Zoubir, F. K. Jondral, and M. Wintermantel, "Advances in Automotive Radar: A framework on computationally efficient high-resolution frequency estimation," *IEEE Signal Process. Mag.*, vol. 34, no. 2, pp. 36–46, Mar. 2017.
[8] S. Saponara, M. S. Greco, and F. Gini, "Radar-on-Chip/in-Package in Autonomous Driving Vehicles and Intelligent Transport Systems: Opportunities and Challenges," *IEEE Signal Process. Mag.*, vol. 36, no. 5, pp. 71–84, Sept. 2019.
[9] S. Patole, M. Torlak, D. Wang, and M. Ali, "Automotive Radars: A review of signal processing techniques," *IEEE Signal Process. Mag.*, vol. 34, pp. 22–35, Mar. 2017.
[10] J. Le Kernec, F. Fioranelli, C. Ding, H. Zhao, L. Sun, H. Hong, J. Lorandel, and O. Romain, "Radar Signal Processing for Sensing in Assisted Living: The challenges associated with real-time implementation of emerging algorithms," *IEEE Signal Process. Mag.*, vol. 36, no. 4, pp. 29–41, July 2019.
[11] J. Yu and J. Krolik, "MIMO adaptive beamforming for nonseparable multipath clutter mitigation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 50, no. 4, pp. 2604–2618, 2014.
[12] O. Simeone, "A Very Brief Introduction to Machine Learning With Applications to Communication Systems," *IEEE Trans. Cogn. Commun. Netw.*, vol. 4, no. 4, pp. 648–664, 2018.
[13] N. Shlezinger, R. Fu, and Y. C. Eldar, "Deep Soft Interference Cancellation for MIMO Detection," in *Proc. 2020 IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2020, pp. 8881–8885.
[14] S. Z. Gurbuz and M. G. Amin, "Radar-Based Human-Motion Recognition With Deep Learning: Promising applications for indoor monitoring," *IEEE Signal Process. Mag.*, vol. 36, no. 4, pp. 16–28, July 2019.
[15] J. Lombacher, M. Hahn, J. Dickmann, and C. Wöhler, "Potential of radar for static object classification using deep learning methods," in *Proc. 2016 IEEE MTT-S Int. Conf. Microw. Intell. Mobility (ICMIM)*, May 2016, pp. 1–4.
[16] J. Winters, "On the Capacity of Radio Communication Systems with Diversity in a Rayleigh Fading Environment," *IEEE J. Sel. Areas Commun.*, vol. 5, no. 5, pp. 871–878, June 1987.
[17] G. J. Foschini, "Layered Space-Time Architecture for Wireless Communication in a Fading Environment When Using Multi-Element Antennas," *Bell Labs Tech. J.*, pp. 41–59, Oct. 1996.
[18] G. Foschini and M. Gans, "On Limits of Wireless Communications in a Fading Environment when Using Multiple Antennas," *Wireless Pers. Commun.*, vol. 6, no. 3, pp. 311–335, Mar. 1998.
[19] E. Telatar, "Capacity of multi-antenna gaussian channels," *Eur. Trans. Telecommun.*, vol. 10, no. 6, pp. 585–595, 1999.
[20] G. Vitetta, D. Taylor, G. Colavolpe, F. Pancaldi, and P. Martin, ***Wireless Communications: Algorithmic Techniques***. New York: Wiley, May 2013.
[21] E. Fishler, A. Haimovich, R. Blum, D. Chizhik, L. Cimini, and R. Valenzuela, "MIMO radar: an idea whose time has come," in *Proc. 2004 IEEE Radar Conf.*, 2004, pp. 71–78.
[22] D. W. Bliss and K. W. Forsythe, "Multiple-input multiple-output (MIMO) radar and imaging: degrees of freedom and resolution," in *Proc. 37th Asilomar Conf. Signals, Sys. Comput.*, vol. 1, 2003, pp. 54–59.
[23] J. Li and P. Stoica, "MIMO Radar with Colocated Antennas," *IEEE Signal Process. Mag.*, vol. 24, no. 5, pp. 106–114, Sept. 2007.
[24] A. M. Haimovich, R. S. Blum, and L. J. Cimini, "MIMO Radar with Widely Separated Antennas," *IEEE Signal Process. Mag.*, vol. 25, no. 1, pp. 116–129, 2008.
[25] E. Fishler, A. Haimovich, R. Blum, R. Cimini, D. Chizhik, and R. Valenzuela, "Performance of MIMO radar systems: advantages of angular diversity," in *Conf. Rec. 38th Asilomar Conf. Signals, Sys. Comput.*, vol. 1, 2004, pp. 305–309.
[26] M. A. Richards, ***Fundamentals of radar signal processing***. New York: McGraw-Hill Education, 2005.
[27] C. Pfeffer, R. Feger, C. Wagner, and A. Stelzer, "FMCW MIMO Radar System for Frequency-Division Multiple TX-Beamforming," *IEEE Trans. Microw. Theory Tech.*, vol. 61, no. 12, pp. 4262–4274, 2013.
[28] R. Feger, C. Pfeffer, and A. Stelzer, "A frequency-division MIMO FMCW radar system using delta-sigma-based transmitters," in *Proc. 2014 IEEE MTT-S Int. Microw. Symp. (IMS)*, 2014, pp. 1–4.
[29] D. Schindler, B. Schweizer, C. Knill, J. Hasch, and C. Waldschmidt, "MIMO-OFDM Radar Using a Linear Frequency Modulated Carrier to Reduce Sampling Requirements," *IEEE Trans. Microw. Theory Tech.*, vol. 66, no. 7, pp. 3511–3520, July 2018.
[30] H. Griffiths, P. Knott, and W. Koch, "Christian Hülsmeyer: Invention and Demonstration of Radar, 1904," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 34, no. 9, pp. 56–60, 2019.
[31] I. C. Society, ***A Brief History of Communications****: IEEE Communications Society - a Fifty-year Foundation for the Future*. The Society, 2002.
[32] N. Wiener, ***Extrapolation, Interpolation, and Smoothing of Stationary Time Series: with Engineering Applications***. Technology Press of the Massachusetts Institute of Technology, 1949.

[33] A. Fenn, D. Temme, W. Delaney, and W. Courtney, "The Development of Phased-Array Radar Technology," *Lincoln Lab. J.*, vol. 12, Jan. 2000.

[34] P. Barton, "Digital beam forming for radar," *IEE Proc. F - Commun., Radar Signal Process.*, vol. 127, no. 4, pp. 266–277, 1980.

[35] S. H. Talisa, K. W. O'Haver, T. M. Comberiate, M. D. Sharp, and O. F. Somerlock, "Benefits of Digital Phased Array Radars," *Proc. IEEE*, vol. 104, no. 3, pp. 530–543, 2016.

[36] Luzhou Xu, Jian Li, and P. Stoica, "Radar imaging via adaptive MIMO techniques," in *Proc. 14th Eur. Signal Process. Conf.*, 2006, pp. 1–5.

[37] M. Kronauge and H. Rohling, "New chirp sequence radar waveform," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 50, no. 4, pp. 2870–2877, 2014.

[38] T. Spreng, U. Prechtel, B. Schönlinner, V. Ziegler, A. Meusling, and U. Siart, "UWB near-field MIMO radar: Calibration, measurements and image reconstruction," in *Proc. 2013 Eur. Radar Conf. (EuRAD)*, 2013, pp. 33–36.

[39] C. Sturm and W. Wiesbeck, "Waveform Design and Signal Processing Aspects for Fusion of Wireless Communications and Radar Sensing," *Proc. IEEE*, vol. 99, no. 7, pp. 1236–1259, 2011.

[40] C. Pfeffer, R. Feger, and A. Stelzer, "A stepped-carrier 77-GHz OFDM MIMO radar system with 4 GHz bandwidth," in *Proc. 2015 Eur. Radar Conf. (EuRAD)*, 2015, pp. 97–100.

[41] A. Bourdoux, U. Ahmad, D. Guermandi, S. Brebels, A. Dewilde, and W. Van Thillo, "PMCW waveform and MIMO technique for a 79 GHz CMOS automotive radar," in *Proc. 2016 IEEE Radar Conf. (RadarConf)*, 2016, pp. 1–5.

[42] D. E. Dudgeon, "Fundamentals of digital array processing," *Proc. IEEE*, vol. 65, no. 6, pp. 898–904, 1977.

[43] J. Capon, "High-resolution frequency-wavenumber spectrum analysis," *Proc. IEEE*, vol. 57, no. 8, pp. 1408–1418, 1969.

[44] P. Stoica, Zhisong Wang, and Jian Li, "Robust Capon beamforming," *IEEE Signal Process. Lett.*, vol. 10, no. 6, pp. 172–175, 2003.

[45] R. Roy and T. Kailath, "ESPRIT-estimation of signal parameters via rotational invariance techniques," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 37, no. 7, pp. 984–995, July 1989.

[46] J. Li and R. T. Compton, "Angle and polarization estimation using ESPRIT with a polarization sensitive array," *IEEE Trans. Antennas Propag.*, vol. 39, no. 9, pp. 1376–1383, 1991.

[47] F. C. Robey, S. Coutts, D. Weikle, J. C. McHarg, and K. Cuomo, "MIMO radar theory and experimental results," in *Conf. Rec. 38th Asilomar Conf. Signals, Sys. Comput.*, vol. 1, Nov. 2004, pp. 300–304.

[48] C. Duo-fang, C. Bai-xiao, and Q. Guo-dong, "Angle estimation using ESPRIT in MIMO radar," in *Electron. Lett.*, vol. 44, no. 12, 2008, pp. 770–771.

[49] F. Belfiori, W. van Rossum, and P. Hoogeboom, "Application of 2D MUSIC algorithm to range-azimuth FMCW radar data," in *Proc. 9th Eur. Radar Conf.*, 2012, pp. 242–245.

[50] D. Cohen and Y. C. Eldar, "Sub-Nyquist Radar Systems: Temporal, Spectral, and Spatial Compression," *IEEE Signal Process. Mag.*, vol. 35, no. 6, pp. 35–58, Nov. 2018.

[51] L. Zhao, L. Wang, L. Yang, A. M. Zoubir, and G. Bi, "The Race to Improve Radar Imagery: An overview of recent progress in statistical sparsity-based techniques," *IEEE Signal Process. Mag.*, vol. 33, no. 6, pp. 85–102, Nov. 2016.

[52] S. Fortunati, R. Grasso, F. Gini, M. Greco, and K. Lepage, "Single-snapshot DOA estimation by using Compressed Sensing," *EURASIP J. Adv. Signal Process. (JASP)*, Nov. 2014.

[53] J. Bock, H. Schafer, K. Aufinger, R. Stengl, S. Boguth, R. Schreiter, M. Rest, H. Knapp, M. Wurzer, W. Perndl, T. Bottner, and T. F. Meister, "SiGe bipolar technology for automotive radar applications," in *Bipolar/BiCMOS Circuits and Technol., Proc. 2004 Meeting*, 2004, pp. 84–87.

[54] A. Hajimiri, H. Hashemi, A. Natarajan, Xiang Guan, and A. Komijani, "Integrated Phased Array Systems in Silicon," *Proc. IEEE*, vol. 93, no. 9, pp. 1637–1655, 2005.

[55] K. Koh and G. M. Rebeiz, "An X- and Ku-Band 8-Element Phased-Array Receiver in 0.18-μm SiGe BiCMOS Technology," *IEEE J. Solid-State Circuits*, vol. 43, no. 6, pp. 1360–1371, 2008.

[56] H. P. Forstner, H. Knapp, H. Jager, E. Kolmhofer, J. Platz, F. Starzer, M. Treml, A. Schinko, G. Birschkus, J. Bock, K. Aufinger, R. Lachner, T. Meister, H. Schafer, D. Lukashevich, S. Boguth, A. Fischer, F. Reininger, L. Maurer, J. Minichshofer, and D. Steinbuch, "A 77GHz 4-channel automotive radar transceiver in SiGe," in *Proc. 2008 IEEE Radio Freq. Integr. Circuits Symp.*, 2008, pp. 233–236.

[57] D. Freundt and B. Lucas, "Long Range Radar Sensor for High-Volume Driver Assistance Systems Market," in *SAE Techn. Paper*. SAE International, Apr. 2008.

[58] C. M. Schmid, R. Feger, C. Wagner, and A. Stelzer, "Design of a linear non-uniform antenna array for a 77-GHz MIMO FMCW radar," in *Proc. 2009 IEEE MTT-S Int. Microw. Workshop Wireless Sens., Local Positioning, RFID*, 2009, pp. 1–4.

[59] Z. Tong, A. Stelzer, and E. Kolmhofer, "77 GHz center-fed differential microstrip antenna array," in *Proc. 5th Eur. Conf. Antennas Propag. (EUCAP)*, 2011, pp. 583–586.

[60] J. Lee, Y. Li, M. Hung, and S. Huang, "A Fully-Integrated 77-GHz FMCW Radar Transceiver in 65-nm CMOS Technology," *IEEE J. of Solid-State Circuits*, vol. 45, no. 12, pp. 2746–2756, 2010.

[61] L. Zheng, M. Lops, Y. C. Eldar, and X. Wang, "Radar and Communication Coexistence: An Overview: A Review of Recent Methods," *IEEE Signal Process. Mag.*, vol. 36, no. 5, pp. 85–99, Sept. 2019.

[62] J. Gamba, ***Radar Signal Processing for Autonomous Driving***, 1st ed. Singapore: Springer Publishing Company Inc., 2019.

[63] L. L. Scharf and C. Demeure, ***Statistical signal processing : detection, estimation, and time series analysis***. Boston: Addison-Wesley Pub. Co., 1991.

[64] S. Sun, A. P. Petropulu, and H. V. Poor, "MIMO Radar for Advanced Driver-Assistance Systems and Autonomous Driving: Advantages and Challenges," *IEEE Signal Process. Mag.*, vol. 37, no. 4, pp. 98–117, July 2020.

[65] T. Strohmer and B. Friedlander, "Compressed sensing for MIMO radar - algorithms and performance," in *Conf. Rec. 43rd Asilomar Conf. Signals, Sys. Comput.*, 2009, pp. 464–468.

[66] Y. Yu, A. P. Petropulu, and H. V. Poor, "MIMO Radar Using Compressive Sampling," *IEEE J. Sel. Topics Signal Process.*, vol. 4, no. 1, pp. 146–163, 2010.

[67] W. Roberts, P. Stoica, J. Li, T. Yardibi, and F. A. Sadjadi, "Iterative Adaptive Approaches to MIMO Radar Imaging," *IEEE J. Sel. Topics Signal Process.*, vol. 4, no. 1, pp. 5–20, 2010.

[68] V. C. Chen, F. Li, S. . Ho, and H. Wechsler, "Micro-Doppler effect in radar: phenomenon, model, and simulation study," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 42, no. 1, pp. 2–21, Jan. 2006.

[69] C. A. Balanis, ***Antenna theory: analysis and design***. Wiley-Interscience, 2005.

[70] J. Selva, "ML Estimation and Detection of Multiple Frequencies Through Periodogram Estimate Refinement," *IEEE Signal Process. Lett.*, vol. 24, no. 3, pp. 249–253, 2017.

[71] C. M. Bishop, ***Pattern Recognition and Machine Learning (Information Science and Statistics)***. Berlin, Heidelberg: Springer-Verlag, 2006.

[72] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967.

[73] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, Aug. 1996.

[74] R. E. Schapire, "A Brief Introduction to Boosting," in *Proc. 16th Int. Joint Conf. Artif. Intell. (IJCAI'99)*, vol. 2, 1999, p. 1401–1406.

[75] J. Fürnkranz, "Round Robin Classification," *J. Mach. Learn. Res.*, vol. 2, pp. 721–747, Sept. 2002.
[76] S. Escalera, O. Pujol, and P. Radeva, "On the Decoding Process in Ternary Error-Correcting Output Codes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, pp. 120–34, Jan. 2010.
[77] O. Simeone, ***A Brief Introduction to Machine Learning for Engineers***. Now Foundations and Trends, 2018.
[78] I. Jolliffe, ***Principal Component Analysis***, M. Lovric, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.
[79] S. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inf. Theory*, vol. 28, no. 2, pp. 129–137, 1982.
[80] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–44, May 2015.
[81] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.
[82] M. Møller, "A Scaled Conjugate Gradient Algorithm For Fast Supervised Learning," *Neural Networks*, vol. 6, pp. 525–533, Dec. 1993.
[83] M. Gori, ***Machine Learning: A Constraint-Based Approach***, 1st ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2017.
[84] E. Mason, B. Yonel, and B. Yazici, "Deep learning for radar," in *Proc. 2017 IEEE Radar Conf.*, May 2017, pp. 1703–1708.
[85] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct. 1990.
[86] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training Recurrent Neural Networks," *Proc. 30th Int. Conf. Mach. Learn. (ICML 2013)*, Nov. 2012.
[87] S. Hochreiter and J. Schmidhuber, "Long Short-term Memory," *Neural comput.*, vol. 9, pp. 1735–80, Dec. 1997.
[88] It.mathworks.com, "Phased Array System Toolbox," 2020. [Online]. Available: https://it.mathworks.com/products/phased-array.html
[89] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *Proc. Int. Conf. Learn. Representations*, Dec. 2014.
[90] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Networks," *Adv. in Neural Inf. Process. Sys.*, vol. 3, June 2014.
[91] S. Abdulatif, K. Armanious, F. Aziz, U. Schneider, and B. Yang, "Towards Adversarial Denoising of Radar Micro-Doppler Signatures," in *Proc. 2019 Int. Radar Conf.*, 2019, pp. 1–6.
[92] B. Vandersmissen, N. Knudde, A. Jalalvand, I. Couckuyt, A. Bourdoux, W. De Neve, and T. Dhaene, "Indoor Person Identification Using a Low-Power FMCW Radar," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 7, pp. 3941–3952, July 2018.
[93] Youngwook Kim, Sungjae Ha, and Jihoon Kwon, "Human Detection Using Doppler Radar Based on Physical Characteristics of Targets," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 2, pp. 289–293, Feb. 2015.
[94] Y. Li, Z. Peng, R. Pal, and C. Li, "Potential Active Shooter Detection Based on Radar Micro-Doppler and Range-Doppler Analysis Using Artificial Neural Network," *IEEE Sensors J.*, vol. 19, no. 3, pp. 1052–1063, Feb. 2019.
[95] F. Fioranelli, M. Ritchie, and H. Griffiths, "Classification of Unarmed/Armed Personnel Using the NetRAD Multistatic Radar for Micro-Doppler and Singular Value Decomposition Features," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 9, pp. 1933–1937, Sept. 2015.
[96] P. Molchanov, J. Astola, K. Egiazarian, and A. Totsky, "Ground moving target classification by using DCT coefficients extracted from micro-Doppler radar signatures and artificial neuron network," in *Proc. 2011 Microw., Radar, Remote Sens. Symp.*, Aug. 2011, pp. 173–176.
[97] R. J. Javier and Y. Kim, "Application of Linear Predictive Coding for Human Activity Classification Based on Micro-Doppler Signatures," *IEEE Geosci. Remote Sens. Lett.*, vol. 11, no. 10, pp. 1831–1834, Oct. 2014.
[98] C. Clemente, L. Pallotta, A. De Maio, J. J. Soraghan, and A. Farina, "A novel algorithm for radar classification based on doppler characteristics exploiting orthogonal Pseudo-Zernike polynomials," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 51, no. 1, pp. 417–430, Jan. 2015.
[99] J. Zabalza, C. Clemente, G. Di Caterina, Jinchang Ren, J. J. Soraghan, and S. Marshall, "Robust PCA micro-doppler classification using SVM on embedded systems," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 50, no. 3, pp. 2304–2310, July 2014.
[100] G. E. Smith, K. Woodbridge, and C. J. Baker, "Naive Bayesian radar micro-doppler recognition," in *Proc. 2008 Int. Radar Conf.*, Sept. 2008, pp. 111–116.
[101] S. Björklund, T. Johansson, and H. Petersson, "Evaluation of a micro-Doppler classification method on mm-wave data," in *Proc. 2012 IEEE Radar Conf.*, May 2012, pp. 0934–0939.
[102] S. Abdulatif, Q. Wei, F. Aziz, B. Kleiner, and U. Schneider, "Micro-Doppler Based Human-Robot Classification Using Ensemble and Deep Learning Approaches," *Proc. 2018 IEEE Radar Conf. (RadarConf18)*, pp. 1043–1048, Apr. 2018.
[103] Y. Kim and H. Ling, "Human Activity Classification Based on Micro-Doppler Signatures Using a Support Vector Machine," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 5, pp. 1328–1337, 2009.
[104] Y. Kim and T. Moon, "Human Detection and Activity Classification Based on Micro-Doppler Signatures Using Deep Convolutional Neural Networks," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 1, pp. 8–12, Jan. 2016.
[105] M. S. Seyfioğlu, A. M. Özbayoğlu, and S. Z. Gürbüz, "Deep convolutional autoencoder for radar-based classification of similar aided and unaided human activities," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 54, no. 4, pp. 1709–1723, Aug. 2018.
[106] S. Abdulatif, F. Aziz, K. Armanious, B. Kleiner, B. Yang, and U. Schneider, "Person Identification and Body Mass Index: A Deep Learning-Based Study on Micro-Dopplers," in *Proc. 2019 IEEE Radar Conf. (RadarConf)*, Apr. 2019, pp. 1–6.
[107] G. Malysa, D. Wang, L. Netsch, and M. Ali, "Hidden Markov model-based gesture recognition with FMCW radar," in *Proc. 2016 IEEE Global Conf. Signal Inf. Process. (GlobalSIP)*, Dec. 2016, pp. 1017–1021.
[108] S. Wang, J. Song, J. Lien, I. Poupyrev, and O. Hilliges, "Interacting with Soli: Exploring Fine-Grained Dynamic Gesture Recognition in the Radio-Frequency Spectrum," in *Proc. 29th Annu. Symp. User Interface Softw. Technol. (UIST '16)*, 2016, pp. 851–860.
[109] Y. Kim and B. Toomajian, "Hand Gesture Recognition Using Micro-Doppler Signatures With Convolutional Neural Network," *IEEE Access*, vol. 4, pp. 7125–7130, 2016.
[110] Z. Peng, C. Li, J.-M. Munoz-Ferreras, and R. Gomez-Garcia, "An FMCW radar sensor for human gesture recognition in the presence of multiple targets," in *Proc. 1st IEEE MTT-S Int. Microw. Bio Conf. (IMBIOC)*, May 2017, pp. 1–3.
[111] Z. Zhang, Z. Tian, and M. Zhou, "Latern: Dynamic Continuous Hand Gesture Recognition Using FMCW Radar Sensor," *IEEE Sensors J.*, vol. 18, pp. 3278–3289, 2018.
[112] Z. Peng, J.-M. Munoz-Ferreras, R. Gomez-Garcia, and C. Li, "FMCW radar fall detection based on ISAR processing utilizing the properties of RCS, range, and Doppler," in *Proc. 2016 IEEE MTT-S Int. Microw. Symp. (IMS)*, May 2016, pp. 1–3.
[113] B. Jokanović and M. Amin, "Fall detection using deep learning in range-doppler radars," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 54, no. 1, pp. 180–189, Feb. 2018.
[114] H. Hong, L. Zhang, C. Gu, Y. Li, G. Zhou, and X. Zhu, "Noncontact Sleep Stage Estimation Using a CW Doppler Radar," *IEEE Trans. Emerg. Sel. Topics in Circuits and Syst.*, vol. 8, no. 2, pp. 260–270, 2018.
[115] J. Saluja, J. Casanova, and J. Lin, "A Supervised Machine Learning Algorithm for Heart-Rate Detection Using Doppler Motion-Sensing Radar," *IEEE J. Electromagn., RF and Microw. in Medicine and Biol.*, vol. 4, no. 1, pp. 45–51, 2020.
[116] C. Gu, J. Wang, and J. Lien, "Deep Neural Network based Body Movement Cancellation for Doppler Radar Vital Sign Detection," in *Proc. 2019 IEEE MTT-S Int. Wireless Symp. (IWS)*, 2019, pp. 1–3.

[117] H. Zhao, H. Hong, D. Miao, Y. Li, H. Zhang, Y. Zhang, C. Li, and X. Zhu, "A Noncontact Breathing Disorder Recognition System Using 2.4-GHz Digital-IF Doppler Radar," *IEEE J. Biomed. Health Inform.*, vol. 23, no. 1, pp. 208–217, 2019.

[118] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in *Proc. 2nd Int. Conf. Knowl. Discovery, Data Mining*, 1996, p. 226–231.

[119] D. Kellner, M. Barjenbruch, J. Klappstein, J. Dickmann, and K. Dietmayer, "Wheel extraction based on micro doppler distribution using high-resolution radar," in *Proc. 2015 IEEE MTT-S Int. Conf. Microw. Intell. Mobility (ICMIM)*, June 2015.

[120] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probability*, vol. 1, 1967, pp. 281–297.

[121] A. Coluccia, A. Fascista, and G. Ricci, "Robust CFAR Radar Detection Using a K-nearest Neighbors Rule," in *Proc. 2020 IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2020, pp. 4692–4696.

[122] A. El Gonnouni, M. Martinez-Ramon, J. L. Rojo-Alvarez, G. Camps-Valls, A. R. Figueiras-Vidal, and C. G. Christodoulou, "A Support Vector Machine MUSIC Algorithm," *IEEE Trans. Antennas Propag.*, vol. 60, no. 10, pp. 4901–4910, Oct. 2012.

[123] Y. Kim, "Detection of Eye Blinking Using Doppler Sensor With Principal Component Analysis," *IEEE Antennas Wireless Propag. Lett.*, vol. 14, pp. 123–126, 2015.

[124] S. Capobianco, L. Facheris, F. Cuccoli, and S. Marinai, "Vehicle Classification Based on Convolutional Networks Applied to FMCW Radar Signals," in *Traffic Mining Appl. to Police Activities - Proc. 1st Italian Conf. Traffic Police (TRAP-2017)*, vol. 728, Oct. 2017, pp. 115–128.

[125] T. Giese, J. Klappstein, J. Dickmann, and C. Wöhler, "Road course estimation using deep learning on radar data," in *Proc. 18th Int. Radar Symp. (IRS 2017)*, June 2017, pp. 1–7.

[126] H. Dbouk, H. Geng, C. M. Vineyard, and N. R. Shanbhag, "Low-Complexity Fixed-Point Convolutional Neural Networks For Automatic Target Recognition," in *Proc. 2020 IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2020, pp. 1598–1602.

[127] K. Patel, K. Rambach, T. Visentin, D. Rusev, M. Pfeiffer, and B. Yang, "Deep Learning-based Object Classification on Automotive Radar Spectra," in *Proc. 2019 IEEE Radar Conf. (RadarConf)*, July 2019.

[128] J. M. García, D. Zoeke, and M. Vossiek, "MIMO-FMCW Radar-Based Parking Monitoring Application With a Modified Convolutional Neural Network With Spatial Priors," *IEEE Access*, vol. 6, pp. 41 391–41 398, 2018.

[129] B. Major, D. Fontijne, A. Ansari, R. T. Sukhavasi, R. Gowaikar, M. Hamilton, S. Lee, S. Grzechnik, and S. Subramanian, "Vehicle Detection With Automotive Radar Using Deep Learning on Range-Azimuth-Doppler Tensors," in *Proc. 2019 IEEE/CVF Int. Conf. Comput. Vision Workshop (ICCVW)*, Oct. 2019, pp. 924–932.

[130] J. Mun, H. Kim, and J. Lee, "A Deep Learning Approach for Automotive Radar Interference Mitigation," in *Proc. 2018 IEEE 88th Veh. Technol. Conf. (VTC-Fall)*, 2018, pp. 1–5.

[131] J. Mun, S. Ha, and J. Lee, "Automotive Radar Signal Interference Mitigation Using RNN with Self Attention," in *Proc. 2020 IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2020, pp. 3802–3806.

[132] H. Rohling, "Radar CFAR Thresholding in Clutter and Multiple Target Situations," *IEEE Trans. Aerosp. Electron. Syst.*, pp. 608–621, 1983.

[133] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Neural Inf. Process. Syst.*, vol. 25, Jan. 2012.

[134] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. 2009 IEEE Conf. Comput. Vision Pattern Recogn.*, 2009, pp. 248–255.

[135] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, 2010.

[136] Z. Zheng, T. Ruan, Y. Wei, and Y. Yang, "VehicleNet: Learning Robust Feature Representation for Vehicle Re-identification," in *Proc. IEEE/CVF Conf. Comput. Vision Pattern Recogn. (CVPR)*, June 2019.

[137] M. S. Seyfioğlu and S. Z. Gürbüz, "Deep Neural Network Initialization Methods for Micro-Doppler Classification With Low Training Sample Support," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 12, pp. 2462–2466, Dec. 2017.

[138] J. Park, R. Javier, T. Moon, and Y. Kim, "Micro-Doppler Based Classification of Human Aquatic Activities via Transfer Learning of Convolutional Neural Networks," *Sensors*, vol. 16, no. 12, p. 1990, Nov. 2016.

[139] M. S. Seyfioglu, B. Erol, S. Z. Gurbuz, and M. G. Amin, "Diversified radar micro-Doppler simulations as training data for deep residual neural networks," in *Proc. 2018 IEEE Radar Conf. (RadarConf18)*, Apr. 2018, pp. 0612–0617.

[140] R. Girshick, "Fast R-CNN," in *Proc. 2015 IEEE Int. Conf. Comput. Vision (ICCV)*, 2015, pp. 1440–1448.

[141] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, June 2017.

[142] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *Proc. 2016 IEEE Conf. Comput. Vision Pattern Recogn. (CVPR)*, 2016, pp. 779–788.

[143] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in *Proc. 2017 IEEE Conf. Comput. Vision Pattern Recogn. (CVPR)*, 2017, pp. 6517–6525.

[144] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. 2015 IEEE Conf. Comput. Vision Pattern Recogn. (CVPR)*, June 2015, pp. 3431–3440.

[145] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.

[146] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *Proc. Med. Image Comput. and Comput.-Assisted Intervention (MICCAI 2015)*, 2015, pp. 234–241.

[147] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. 2017 IEEE Int. Conf. Comput. Vision (ICCV)*, 2017, pp. 2980–2988.

[148] G. Zhang, H. Li, and F. Wenger, "Object Detection and 3d Estimation Via an FMCW Radar Using a Fully Convolutional Network," in *Proc. 2020 IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2020, pp. 4487–4491.

[149] O. Bialer, D. Shapiro, and A. Jonas, "Object Surface Estimation from Radar Images," in *Proc. 2020 IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2020, pp. 4132–4136.

[150] S. Gasperini, M. Paschali, C. Hopke, D. Wittmann, and N. Navab, "Signal Clustering With Class-Independent Segmentation," in *Proc. 2020 IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2020, pp. 3982–3986.

[151] O. Schumann, M. Hahn, J. Dickmann, and C. Wöhler, "Semantic Segmentation on Radar Point Clouds," *Proc. 21st Int. Conf. Inf. Fusion (FUSION)*, pp. 2179–2186, 2018.

[152] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," in *Proc. 2017 IEEE Conf. Comput. Vision Pattern Recogn. (CVPR)*, 2017, pp. 77–85.

[153] M. Meyer and G. Kuschk, "Automotive Radar Dataset for Deep Learning Based 3D Object Detection," in *Proc. 16th Eur. Radar Conf. (EuRAD)*, Oct. 2019, pp. 129–132.

[154] M. Meyer and G. Kuschk, "Deep Learning Based 3D Object Detection for Automotive Radar and Camera," in *Proc. 16th Eur. Radar Conf. (EuRAD)*, 2019, pp. 133–136.

[155] A. B. Arrieta, N. Díaz-Rodríguez, J. D. Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera, "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," *Inf. Fusion*, vol. 58, pp. 82 – 115, 2020.
[156] M. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks," in *Proc. Eur. Conf. Comput. Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., vol. 8689, Nov. 2013.
[157] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps," in *Proc. Workshop at Int. Conf. Learn. Representations*, 2014.
[158] "Inras," 2020. [Online]. Available: http://www.inras.at/
[159] "Pico-flexx." [Online]. Available: https://pmdtec.com/picofamily/flexx/
[160] J. Zhu, S. Rosset, H. Zou, and T. Hastie, "Multi-class AdaBoost," *Statist. and its interface*, vol. 2, Feb. 2006.
[161] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.
[162] "Parallax inc." [Online]. Available: https://www.parallax.com/
[163] L. Al Shalabi and Z. Shaaban, "Normalization as a Preprocessing Engine for Data Mining and the Approach of Preference Matrix," *Proc. 2006 Int. Conf. Dependability Comput. Sys.*, pp. 207–214, 2006.