On the Edge Recurrent Neural Network Approach for Ground Moving FMCW Radar Target Classification

Christian Gianoglio*, *Member, IEEE,* Ali Rizik*, Emanuele Tavanti, *Member, IEEE,* Daniele D. Caviglia, *Life Member, IEEE,* Andrea Randazzo, *Senior Member, IEEE*

***Abstract*—In this paper, an approach for ground-moving target classification with an FMCW radar is proposed. In particular, data are collected using a low-cost 24 GHz off-the-shelf FMCW radar, combined with an embedded Raspberry Pi device for data acquisition and processing. An FFT-based processing scheme is then applied to obtain a sequence of range-Doppler maps, which are provided in input to different convolutional neural network (CNN) architectures for classifying the targets (cars, motorcycles, or pedestrians) eventually passing in front of the radar. Specifically, two approaches have been followed and compared. In the first one, single range-Doppler maps are processed alone using a convolutional neural network, and then a voting mechanism is applied to select the target classes. In the second approach, a sequence of range-Doppler maps is processed using a time-distributed layer feeding a recurrent neural network. The CNNs are deployed on the Raspberry Pi providing the target classification on a low-cost embedded device. The obtained results show that the proposed approaches allow for effectively detecting the different types of targets running on an embedded device in less than one second.**

***Index Terms*—Deep neural networks, recurrent neural networks, radar, and target classification.**

I. Introduction

KNOWLEDGE and a good understanding of our environment are the key factors for protecting ourselves against external intrusions. In recent years, efficient defense and surveillance systems have been growing interest among civilians to protect their goods. In fact, reliable monitoring and surveillance systems have started to be installed in urban areas and near-critical zones [1]–[4]. Indeed, the recent development of autonomous driving cars has pushed engineers into designing reliable pedestrian recognition systems to ensure pedestrian safety [5]–[7].

Radar and camera sensors are nowadays considered valuable options to be integrated into such systems [8]. Radars are more reliable to be exploited in adverse weather and bad lighting conditions [9], [10], whereas optical sensors deteriorate [11]. In particular, frequency-modulated continuous-wave (FMCW) radars have been gaining more interest in several applications [12]–[23]. Moreover, FMCW radars are generally cheap and require low sampling rates and low peak-to-average-power ratio for distance and speed detection of multiple moving targets [24]. Many papers in the literature address the moving target recognition through FMCW radars [9], [10], [25]–[28] supporting machine learning (ML) algorithms, such as support vector machines (SVMs) or Deep Neural Networks (DNNs) [26], [29], [30]. The use of DNNs in FMCW radar target classification has also been reported in [31]–[34].

On the other hand, DNNs, accompanied by the use of micro-Doppler signatures [35], [36], are also an alternative approach for the classification of ground targets [37]. This approach has proven effective in several applications targeting human recognition and activity classification [38]–[41], as well as human-robot classification [42]. Nevertheless, the extraction of a micro-Doppler signature is not a simple process, as it usually requires long illumination periods of the targets. This fact makes it hard to extract such features with low-cost radar devices, especially in the presence of relatively fast targets like cars and motorcycles [43].

At the same time, it is necessary to keep in mind that the adoption of DNNs usually raises issues because of the large amount of training samples that are needed [44]. This indeed creates a problem, especially when working with low-cost radars that suffer from hardware limitations that hinder their capabilities to collect a sufficient amount of data in a limited time. However, this problem can be reduced with the adaptation of recurrent neural networks (RNNs), as they involve the time-varying information of the target in the final decision [45], [46]. This could compensate for the low amount of available data and hugely affect the accuracy of the final decision. In fact, RNNs are known for their ability to extract temporal features from the available sequence of time-varying data. This indeed is an interesting approach to be considered in radar applications since radar data is usually a collection of consecutive time-varying data frames. This temporal variability is usually caused by the movement of the target during the illumination period. Some examples of the use of RNNs in radar applications have been presented in the literature. RNNs were adopted by authors in [31], [47]–[49], for human recognition and target classification in radar security applications. In addition, synthetic aperture radar

*The authors have contributed equally.

Manuscript received XXX, 2023. This work is partially supported by the PNRR Smart Mobility project (CUP I53C22000720001).

C. Gianoglio, D. Caviglia, and A. Randazzo are with the Department of Electrical, Electronic, Telecommunications Engineering and Naval Architecture (DITEN), University of Genoa, 16145 Genoa, Italy (e-mail: {christian.gianoglio, daniele.caviglia, andrea.randazzo}@unige.it).

A.Rizik is with the Department of Environmental, Land and Infrastructure Engineering (DIATI), Polytechnic University of Turin, 10129 Turin (e-mail:ali.rizik@polito.it).

E. Tavanti is with the Department of Information Engineering (DII), University of Pisa, 56122 Pisa, Italy (e-mail: emanuele.tavanti@unipi.it)

(SAR) applications are an interesting field of study to use RNNs, as presented by the authors in [50]–[52].

This paper proposes a low-cost system for ground-moving target classification based on a DNN designed from scratch. An edge device collects the signals from an FMCW radar and transforms them into range-Doppler maps. The DNN, deployed on the edge, receives as input a sequence of range-Doppler maps treated as a series of images. The DNN consists of a convolutional neural network (CNN) that automatically extracts the features from each map and a recurrent neural network (RNN) that provides the classification of the moving targets. The performance of the system is evaluated in terms of accuracy and computational cost measured on the edge device. The computational cost is represented by the inference time and energy consumption. The evaluation is assessed on a three-class classification problem, i.e. pedestrians, cars, and motorcycles. For the sake of comparison, other three classifiers are designed and implemented on the edge device: a DNN designed upon a pre-trained CNN that classifies the sequence of range-Doppler maps, and two CNNs that classify a single range-Doppler map. One single-map classifier has the same architecture as the CNN employed in the DNN designed from scratch, while the other one has the same architecture as the pre-trained CNN. To increase the generalization accuracy of the single range-Doppler map classifiers, a voting mechanism is also applied. It consists of assigning the label to a sequence of maps classified as single inputs, based on the most frequent class. The contributions of this paper are summarized as follows:

- A low-cost system, based on an FMCW radar and an edge device, is adopted for three-class moving target classification.
- A DNN designed from scratch and deployed on the edge classifies a sequence of range-Doppler maps achieving high accuracy while providing a real-time inference measured on the edge with restrained energy consumption.

The remainder of the paper is organized as follows. Section II presents the methodology behind using a sequence of range-Doppler maps as an input for the DNN. Sections III illustrate the collected datasets. Section IV shows the adopted network architectures, and the training procedure is detailed in Section V. Finally, the obtained results are discussed in Section VI and VII. Finally, conclusions are presented in Section VIII.

II. Methodology

This paper proposes a system for the multi-class moving target classification consisting of a low-cost FMCW radar and an edge device powered by an external battery. The system is shown in [27].

The edge device, a Raspberry Pi4 (RP), hosts two stages: the pre-processing stage for the extraction of the range-Doppler (RD) maps from the signals received from the FMCW radar and the classification stage in which the moving target label is predicted. In the following, the two stages are detailed.

A. Pre-processing Stage

In an FMCW radar, an up-chirp, namely a sinusoidal signal having constantly increasing frequency, is emitted. More in detail, a burst of N_c up-chirps is transmitted using a dedicated (TX) antenna [53], [54]. The related irradiated electromagnetic wave bounces off a target present in the monitored area and the related echo is gathered by a receiving (RX) antenna. The resulting signal at the input of the receiving chain is a copy of the transmitted burst delayed by a time $\tau = 2R/c$, where R is the target range, and c is the speed of light. The short-range FMCW radars usually adopt an I/Q demodulator in the receiver chain. The signal at the output of this demodulator called the Intermediate Frequency (IF) signal, can be represented as [55]–[57]:

$$S_{IF}(t_c) = A_b e^{j2\pi\left[\frac{B}{T_c}\tau t_c - f_D n_c T_{crp}\right]} \tag{1}$$

where $t_c \in [0, T_c]$ is the time variable (also known as fast-time) inside a single up-chirp, A_b is the amplitude of the IF signal, B is the sweeping bandwidth (namely the amount of change in frequency of an up-chirp), T_c is the up-chirp signal duration, v_r is the target's radial velocity ($v_r > 0$ for departing targets), n_c is the chirp index among the N_c transmitted chirps (also known as slow-time), $f_D = -2v_r/\lambda_0$ is the Doppler shift (with λ_0 free-space wavelength), and T_{crp} is the chirp repetition period (this usually includes T_c and a pause time before the firing of the following up-chirp).

For each received chirp, a number of N_s samples is considered, thus, forming a 2D data matrix of dimensions $N_c \times N_s$. Such a matrix undergoes a 2D FFT [58], by performing Fourier transforms along the fast-time and slow-time dimensions, to obtain an RD map; a hypothetical point-like target having given radial velocity v_r and range R will appear as a peak of intensity in the RD map, at coordinates strictly related to v_r and R. Beyond the indication of the range and radial velocity of a target, the importance of the RD map is that particular features of the geometry and/or movement of a target appear on this map as time-varying patterns [43], [59], [60]. This encourages the adoption of the machine learning algorithms mentioned in Section I.

To adopt standard deep networks for image processing and classification, which assume that the input is a monochromatic or color image, 2D images are obtained by considering the amplitude of the RD maps. Consequently, such images can be represented as 3D digital tensors, i.e., $\mathcal{RD} \in \mathbb{N}^{N_R \times N_D \times C}$, where N_R and N_D are the number of samples outputted from the range and Doppler FFTs, and C represents the number of channels used for representing the RD map as an image.

B. Classification Stage

Besides the stage for the extraction of the RD maps, the RP hosts a DNN designed from scratch to classify a moving target by a sequence of RD maps. This sequence is time-dependent and can be formalized as a 4D tensor $\mathcal{X} \in \mathbb{N}^{N_R \times N_D \times C \times T}$, where T represents the number of RD maps in the sequence. The aim of collecting T RD maps is to increase the classification accuracy concerning a single RD maps classifier as shown in [31]. The proposed DNN consists of a CNN that extracts automatically the features from each RD map employing a time-distributed layer (TDL). The TDL applies the same layers or architecture to every time step of the input. In this work,

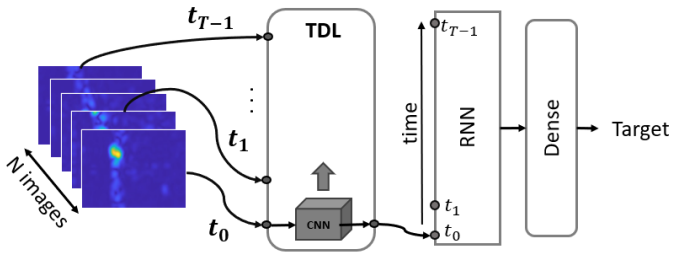


Fig. 1: Deep neural network architecture.

TABLE I: Radar Sensor Parameters

Symbol	Description	Value
B	Sweeping bandwidth	200 MHz
f_0	Starting frequency	24.025 GHz
f_s	ADC sampling rate	42.7 kHz
R_{max}	Maximum unambiguous range	25 m
V_{max}	Maximum unambiguous velocity	5.4 km/hr
ΔR	Range resolution	0.75 m
ΔV	Velocity resolution	0.4 km/hr
N_s	Number of samples/chirp	64
N_c	Number of chirps/frame	21
T_c	Up-chirp time	1.5 ms
T_{crp}	Chirp repetition time	2.1 ms

TDL wraps the CNN to extract features from the sequence of RD maps, producing a sequence of feature maps. A recurrent neural network (RNN) learns the time dependency between the outputs of the TDL. Finally, a Dense layer provides the moving target label. The DNN architecture is shown in Fig. 1. To demonstrate the effectiveness of our design, three other DNNs have been employed for classifying the moving target. The first DNN classifies the sequence of RD maps and it is built upon the pre-trained MobileNetV2 CNN architecture [61]. The other two networks predict the class of the moving target receiving as input only a single RD map as a 3D tensor. The first network is a 2D CNN having the same architecture as the one wrapped by the TDL in the proposed DNN, while the second corresponds to the MobileNetV2 CNN. To increase the generalization accuracy of the two single RD map classifiers, a voting mechanism is also applied: the eventual label is assigned based on the most frequent class in a sequence of classified data. In summary, six DNNs are compared both in terms of generalization accuracy and computational cost measured on the edge device:

1) a DNN design from scratch receiving a 4D tensor as input;
2) a DNN encapsulating the MobileNet V2 CNN receiving a 4D tensor as input;
3) a CNN having the same architecture as the one employed in point 1) receiving a 3D tensor as input;
4) a MobileNet V2 CNN receiving a 3D tensor as input;
5) same network of point 3) and applying the voting mechanism;
6) same network of point 4) and applying the voting mechanism.

In the following, the DNNs will be named $S-DNN$, $MN-DNN$, $S-CNN$, $MN-CNN$, $S-CNN_{Vote}$, and $MN-CNN_{Vote}$, respectively.

III. Datasets Collection

The Distance2Go radar module [62], developed by Infineon, is used to collect the datasets described in [27], [28]. Table I shows the complete set of radar sensor parameters. The number of points for the range and Doppler FFTs was chosen as $N_R = N_D = 256$. The outcome of the pre-processing stage on a moving target, described in Section II-A, is an RD map represented as a 3D tensor $\mathcal{RD} \in \mathbb{N}^{256\times256\times3}$. To be compliant with the input size of the DNNs described in the following sections, the RD maps were resized as $\mathcal{RD} \in \mathbb{N}^{224\times224\times3}$.

The RD maps were collected in two cluttered environments on three kinds of moving targets (i.e., pedestrians, cars, and motorcycles) by the FMCW radar-based system. The two environments, described in previous works [27], [28], involved different environmental conditions for clutter interference, side obstacles, and electromagnetic wave scattering. The number of RD maps of each target motion depended on the speed of the motion itself in the FoV of the radar. In this work, 12 datasets were extracted from the original data of [27], [28]: six concern the single RD maps, and six are the sequences of RD maps. The datasets are detailed in the two following subsections.

A. Single RD Maps Datasets

Two datasets were derived by RD maps collected by the system mounted on a pole at a 1.5 m height, near an internal road of the University of Genoa, Italy [27], [28]. Concerning the previous works, more samples for each target were considered in this proposal. A set of 300 moving targets was gathered, which was equally divided among three classes, i.e., cars, pedestrians, and motorcycles. In [27], 60 cars (of which 30 were trucks), 30 pedestrians, and 30 motorcycles were adopted while, in [28], 95 cars (of which 7 were trucks), 31 pedestrians, and 56 motorcycles were considered.

The first dataset contains 3 RD maps for each target motion while the second includes 5 RD maps. Motions with a greater number than three or five maps were sub-sampled. The sub-sampling consisted of taking the first and the last maps and extracting the remaining maps randomly. As a result, the related datasets, namely $\mathcal{SNG}1_T$ with $T \in \{3,5\}$, contain 900 and 1500 data (i.e., 300 and 500 RD maps per class), respectively. The datasets can be formalized as:

$$\mathcal{SNG}1_T = \Big\{(\mathcal{X},y)_i,\ \mathcal{X}_i \in \mathbb{N}^{224\times224\times3},\ y_i \in \{ped, car, moto\};\ i = 1,...N_{1,T}^{sng}\Big\}.$$

where $N_{1,T}^{sng} = \{900, 1500\}$ represents the number of data in the first scenario depending on the choice of T.

In the second scenario [28], the radar was mounted on a pole near a different road in the city of Genoa. The radar was placed at a higher position (3 m) than the previous one. This set comprises 115 targets (i.e. 60 pedestrians, 43 cars, and 12 motorcycles). Also, in this case, more samples were included. In [28] only 10 cars (of which 5 were trucks), 5 pedestrians, and 5 motorcycles were tested. Other two datasets were built, namely $\mathcal{SNG}2_T$ with $T \in \{3,5\}$, containing 3 and 5 RD maps

TABLE II: Single RD Maps Datasets

Dataset Names	Class	Num of Motions per Class	Num of Total Images $T=3$	Num of Total Images $T=5$
$\mathcal{SNG}1_T$	Mot	100	300	500
	Car	100	300	500
	Ped	100	300	500
$\mathcal{SNG}2_T$	Mot	12	36	60
	Car	43	129	215
	Ped	60	180	300
$\mathcal{SNG}M_T$	Mot	112	336	560
	Car	143	429	715
	Ped	160	480	800

TABLE III: RD Maps Sequences Datasets

Dataset Names	Class	Num of Motions/Data per Class
$\mathcal{SEQ}1_{3/5}$	Mot	100
	Car	100
	Ped	100
$\mathcal{SEQ}2_{3/5}$	Mot	12
	Car	43
	Ped	60
$\mathcal{SEQ}M_{3/5}$	Mot	112
	Car	143
	Ped	160

for each target, respectively. The datasets can be formalized as:

$$\mathcal{SNG}2_T = \big\{(\mathcal{X}, y)_i,\ \mathcal{X}_i \in \mathbb{N}^{224\times224\times3}, \\ y_i \in \{ped, car, moto\};\ i = 1, ...N^{sng}_{2,T}\big\}$$

where $N^{sng}_{2,T} = \{345, 575\}$ represents the number of data in the second scenario depending on the choice of T.

A third couple of datasets stems from the merge of $\mathcal{SNG}1_T$ with $\mathcal{SNG}2_T$. The resulting datasets are named $\mathcal{SNG}M_T$ and $\mathcal{SNG}M_T$. The two datasets include 160 pedestrians, 143 cars, and 112 motorcycles, and each target motion is a collection of 3 and 5 RD maps respectively. Hence, they can be formalized as:

$$\mathcal{SNG}M_T = \big\{(\mathcal{X}, y)_i,\ \mathcal{X}_i \in \mathbb{N}^{224\times224\times3}, \\ y_i \in \{ped, car, moto\};\ i = 1, ...N^{sng}_{M,T}\big\}$$

where $N^{sng}_{M,T} = \{1245, 2075\}$ represents the number of data in the merged scenarios depending on the choice of T.

Table II summarizes the six single RD maps-based datasets. The first column reports the dataset names, the second the labels of the three targets, the third the number of motions for each class, and the last column the total number of RD maps collected per class based on the value of T.

B. Sequence RD Maps Datasets

Following a similar approach as in [31], to assess the effect of including the time-varying components of the radar signals on the moving target classification, the RD maps of each target are stacked into a 4D tensor which contains all the information related to the moving target dynamics. As a result, six datasets containing sequences of RD maps for each target were obtained by the single RD maps datasets. Hence, the RD maps of each $\mathcal{SNG}$ dataset were stacked generating a sequence of maps for each target. In particular, in the dataset $\mathcal{SEQ}1_T$ with $T = \{3, 5\}$, three and five RD maps, respectively, collected for each target in the first scenario were stacked generating a sequence of RD maps. The datasets can be formalized as:

$$\mathcal{SEQ}1_T = \big\{(\mathcal{X}, y)_i,\ \mathcal{X}_i \in \mathbb{N}^{224\times224\times3\times T}, \\ y_i \in \{ped, car, moto\};\ i = 1, ...N^{seq}_1\big\}.$$

where $N^{seq}_1 = 300$. Straightforwardly, the other datasets can be represented as:

$$\mathcal{SEQ}2_T = \big\{(\mathcal{X}, y)_i,\ \mathcal{X}_i \in \mathbb{N}^{224\times224\times3\times T}, \\ y_i \in \{ped, car, moto\};\ i = 1, ...N^{seq}_2\big\}$$
$$\mathcal{SEQ}M_T = \big\{(\mathcal{X}, y)_i,\ \mathcal{X}_i \in \mathbb{N}^{224\times224\times3\times T}, \\ y_i \in \{ped, car, moto\};\ i = 1, ...N^{seq}_M\big\}$$

where $N^{seq}_2 = 115$ and $N^{seq}_M = 415$.

Table III summarizes the six sequence RD maps-based datasets. The first column reports the dataset names, the second the labels of the three targets, and the third the number of motions for each class corresponding to the number of RD maps sequences.

Figure 2 shows some examples of the RD maps for the three classes car, motorcycle, and pedestrian, respectively. The first row of each class corresponds to the pictures captured in three instants by a camera mounted next to the radar, while the second row contains the RD maps corresponding to the pictures. The x-axis and y-axis of the RD maps represent the range and target radial velocity quantities, respectively. Since the maximum unambiguous radial velocity measured by the radar is limited to 5.4 km/h, as reported in Table I, an aliasing phenomenon is present in the Doppler spectrum, resulting in Doppler peaks around 0 km/h for vehicles. By comparison, the vehicle's RD maps can change depending on the position of the moving vehicle with respect to the radar, whose position is fixed. When the vehicle shows its larger side to the radar, the contact surface between the radar beam and the vehicle is higher, corresponding to more reflections on the radar. Several of these reflections are received at different range and Doppler bins, resulting in multiple peaks in the RD maps. The greater the contact surface the higher the possibility of RD maps presenting multiple peaks, in fact in the figure the car presents multiple peaks in $t = t_2$ and $t = t_3$.

A different pattern can be observed from pedestrians' RD maps: their motions include the movements of different body parts. This implies that each part will generate its own Doppler frequency, causing a Doppler spread. The proposed approach based on recurrent neural networks employs the sequence of RD maps to learn the time-varying components of consecutive maps, significantly to differentiate between cars and motorcycles that can present more similarities in a single map. The results will prove that taking into account the RD maps sequences the classification accuracy outperforms the one achieved by predicting moving targets based on a single map.

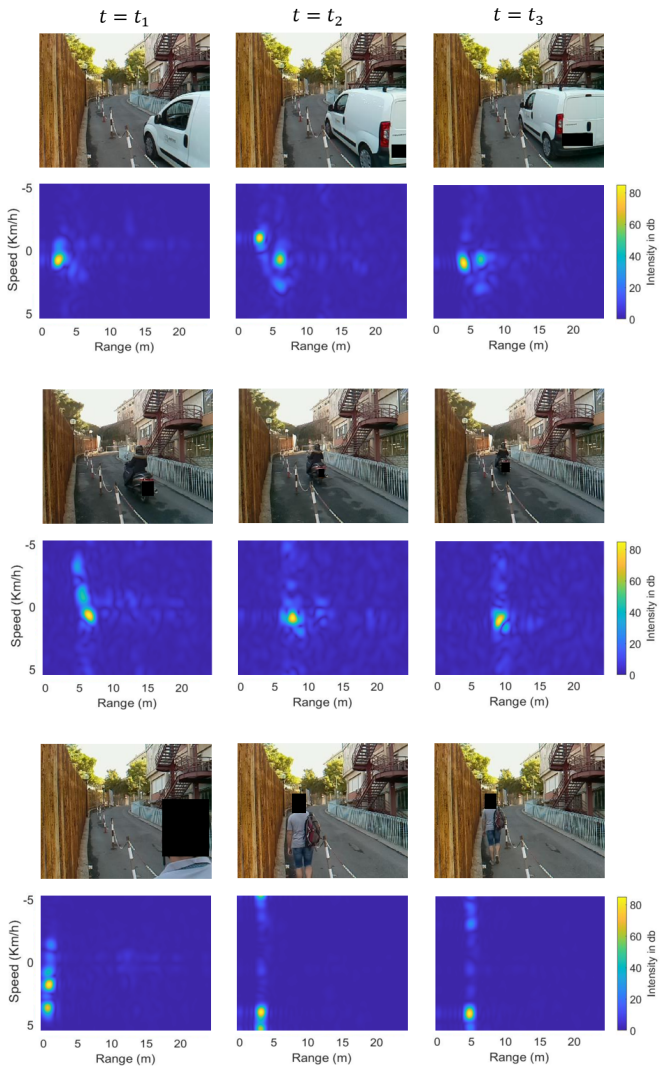


Fig. 2: Example of time-varying radar range-Doppler maps of a moving car, motorcycle, and pedestrian.

C. Statistical Analysis on the RD maps

To justify the importance of merging the datasets collected from two distinct environments, we conducted the following statistical analysis. For each scenario, we randomly selected 50 RD maps from each class. For each RD map, we computed the histogram horizontal projection and the vertical one (a common approach in computer vision, e.g. [63]), related to the range and Doppler dimensions, respectively. From these, we extracted the standard deviations on the range and the Doppler dimensions, indicated as σ_r and σ_D, respectively. Thereafter, within each class in each scenario, we computed the mean and the standard deviation of both σ_r and σ_D, indicated as $E(\sigma_r)$, $E(\sigma_D)$, $SD(\sigma_r)$, and $SD(\sigma_D)$, where $E(\cdot)$ and $SD(\cdot)$ are the mean and standard deviation operators, respectively. Table IV illustrates the computed values. This indicates significant statistical differences between the two scenarios and between the classes within a scenario. Consequently, a training set built with observations coming only from a single scenario will lead to a classifier that probably poorly generalizes to test data coming from the other scenario.

TABLE IV: Statistical analysis of a random sample coming from the two scenarios.

	$\mathcal{SNG}1$			$\mathcal{SNG}2$		
	Car	**Mot**	**Ped**	**Car**	**Mot**	**Ped**
$E(\sigma_r)$	1.86	2.02	1.07	4.11	2.77	2.00
$SD(\sigma_r)$	0.55	0.97	0.86	2.38	1.93	2.05
$E(\sigma_D)$	1.55	2.21	1.83	2.11	2.31	2.93
$SD(\sigma_D)$	0.59	0.83	1.25	1.03	1.05	1.23

IV. Deep Neural Networks for Targets Classification

This section describes the architectures of the six networks designed for the moving target classification. All the networks were designed in Python, using the Keras library provided by Tensorflow.

A. Single RD Map Classifiers

Two CNNs were employed to classify the single RD maps, the first designed from scratch and the second based on the MobileNetV2 architecture [61]. As mentioned, they are named $S-CNN$ and $MN-CNN$, respectively. Table V presents the $S-CNN$ architecture, while Table VI the $MN-CNN$ architecture. In both tables, the first column represents the size of the input tensor for each layer, while the second shows the operation applied to that tensor. As an example, in the second row of Table V five 2D-Convolutional layers with 3×3 kernels are applied to a $224\times224\times8$ tensor that is the output of the first 2D-Convolutional layer. The input dimension of the next row represents the output of the current one. All the convolutional layers used the ReLU as an activation function. The dimension of the CNNs output, not reported in the table, is equal to the number of predicted classes (i.e., three in this paper). The last row of the tables represents the total number of parameters of the networks. The $S-CNN_{Vote}$ and $MN-CNN_{Vote}$ networks, which exploit the voting mechanism, present the same architectures of $S-CNN$ and $MN-CNN$, respectively. With the voting mechanism, the label of the moving target is estimated on a sequence of RD maps, elaborated by the classifiers as single inputs (i.e. as 3D tensors): the moving target label is assigned based on the majority of labels assigned to the RD maps of the sequence. Procedure 1 depicts the voting mechanism. In particular, in the case of two or more most frequent classes, the label is retrieved by summing the T probabilities of the most frequent classes from the softmax layer and taking the highest one.

B. Sequence RD Maps Classifiers

Two DNNs, i.e. $S-DNN$ and $MN-DNN$, were designed to classify the sequences of RD maps as 4D tensors. A time-distributed layer (TDL) wraps a CNN to extract the features from each RD map of the input sequence, thus obtaining a sequence of feature tensors. A recurrent neural network (RNN) learns the dependencies between the feature tensors. Figure 3 shows an example of a CNN wrapped by TDL and applied over T maps of the same sequence to feed an RNN. The two CNNs wrapped by the TDL are the $S-CNN$ and $MN-CNN$ architectures previously presented, excluding the two

TABLE V: $S-CNN$ architecture

Input	Operator	Input	Operator
$224^2 \times 3$	3×3 Conv2D	$28^2 \times 64$	BatchNorm
$224^2 \times 8$	5 3×3 Conv2D	$28^2 \times 64$	2×2 MaxPool2D
$224^2 \times 8$	BatchNorm	$14^2 \times 64$	3×3 Conv2D
$224^2 \times 8$	2×2 MaxPool2D	$14^2 \times 128$	3×3 Conv2D
$112^2 \times 8$	3×3 Conv2D	$14^2 \times 128$	BatchNorm
$112^2 \times 16$	4 3×3 Conv2D	$14^2 \times 128$	2×2 MaxPool2D
$112^2 \times 16$	BatchNorm	$7^2 \times 128$	3×3 Conv2D
$112^2 \times 16$	2×2 MaxPool2D	$7^2 \times 256$	BatchNorm
$56^2 \times 16$	3×3 Conv2D	$7^2 \times 256$	GlobMaxPool2D
$56^2 \times 32$	3 3×3 Conv2D	256	Dense, ReLU
$56^2 \times 32$	BatchNorm	64	Dense, Softmax
$56^2 \times 32$	2×2 MaxPool2D		
$28^2 \times 32$	3×3 Conv2D	**Num Of Params**	673,595
$28^2 \times 64$	2 3×3 Conv2D		

TABLE VI: $MN-CNN$ architecture

Input	Operator
$224^2 \times 3$	MobileNetV2 [61]
$7^2 \times 1280$	GlobalMaxPool2D
1280	Dense, ReLU
64	Dense, Softmax
Num Of Params	2,340,163

dense layers at the bottom of the networks. The Long Short-Term Memory (LSTM) layer is chosen as RNN because it proved to be suitable in many applications based on time-series [64]. Two dense layers represent the output for both networks. The first one is a fully connected layer with the ReLU activation function, while the second presents 3 neurons, i.e. the number of types of moving targets, with the Softmax activation to assign the label. Tables VII and VIII summarize the two architectures.

V. Experimental Setup

This section provides an extensive description of the training setup for the classifiers described in Sec. IV.

Procedure 1 Procedure of the voting mechanism

Input: $\hat{\boldsymbol{y}} \leftarrow T$ predicted labels of an RD maps sequence, $P \leftarrow$ predicted probabilities ($Softmax$ layer output) of the T RD maps

1. **Voting**

$freq \leftarrow$ count the labels in $\hat{\boldsymbol{y}}$
$ind_{max} = \text{argmax}(freq)$
if len(ind_{max}) == 1 **then**
  $\hat{y}_{vote} = ind_{max}$
else
  $p = 0$, $ind_{best} = None$
  for i in ind_{max} **do**
    $p_{temp} = \sum_{j=1}^{T} P[i][j]$
    if $p_{temp} > p$ **then**
      $ind_{best} = i$, $p = p_{temp}$
    end if
  end for
  $\hat{y}_{vote} = ind_{best}$
end if

2. **Return:** $\hat{y}_{vote}$

(a) RD map in $t = t_0$ filtered by the CNN (b) RD map in $t = t_1$ filtered by the CNN

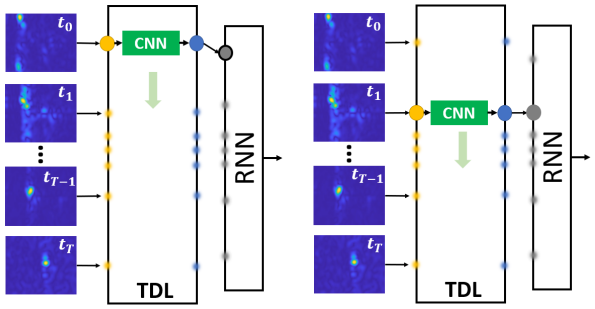


Fig. 3: Example of CNN wrapped by the TDL and applied to the first (a) and second (b) frames of the input sequence. Each output feeds the RNN.

TABLE VII: $S-DNN$ architecture

Input	Operator
$224^2 \times 3 \times T$	TDL ($S-CNN$ excluding Denses)
$256 \times T$	LSTM
32	Dense, ReLU
64	Dense, Softmax
Num Of Params	696,251

A. Single RD Map Classifiers Training Setup

The $S-CNN$ and $MN-CNN$ classifiers were trained on $\mathcal{SNG}1_T$ and $\mathcal{SNG}M_T$ datasets, with $T = \{3, 5\}$. The $MN-CNN$ networks were tuned by unfreezing the last four layers of the pre-trained network corresponding to 412,800 tunable parameters. When the networks were trained on $\mathcal{SNG}1_T$, the datasets $\mathcal{SNG}2_T$ were used as test sets. The first part of Table IX reports the hyper-parameters adopted during the training of the two models for the training datasets. In particular, the first column represents the hyper-parameters (i.e., batch size Bs, number of epochs Ep, and learning rate Lr), and from the second to the last, the table lists their values. For the sake of good visualization, the datasets are grouped as follows: the $\mathcal{SNG}1_{3/5}$ column contains the hyper-parameters values of $\mathcal{SNG}1_3$ and $\mathcal{SNG}1_5$, and $\mathcal{SNG}M_{3/5}$ shows the hyper-parameters of $\mathcal{SNG}M_3$ and $\mathcal{SNG}M_5$. A '/' symbol separates, if any, the differences in the hyper-parameter configurations between the grouped datasets.

In addition, the four training datasets were randomly split into the training/validation/test sets. The validation set was used for an early stopping criterion, setting the patience value to five for all the configurations on the validation loss. All the networks were trained by using the Adam optimizer [65]. The number of data of the splits for each class and each pair model/dataset is shown in Table X. Specifically, the first half of the table shows the number of samples for each split and class

TABLE VIII: $MN-DNN$ architecture

Input	Operator
$224^2 \times 3 \times T$	TDL ($MN-CNN$ excluding Denses)
$256 \times T$	LSTM
32	Dense, ReLU
64	Dense, Softmax
Num Of Params	2,428,355

TABLE IX: Hyper-parameters of $S-CNN$, $MN-CNN$, $S-DNN$ and $MN-DNN$ with respect to the employed training dataset

	$\boldsymbol{S-CNN}$		$\boldsymbol{MN-CNN}$	
Hyp-par	$\mathcal{SNG}1_{3/5}$	$\mathcal{SNG}M_{3/5}$	$\mathcal{SNG}1_{3/5}$	$\mathcal{SNG}M_{3/5}$
Bs	15/25	15/25	15/25	15/25
Ep	100	100	100	100
Lr	$5/7e^{-4}$	$10e^{-4}$	$5/10e^{-4}$	$5e^{-4}$
	$\boldsymbol{S-DNN}$		$\boldsymbol{MN-DNN}$	
Hyp-par	$\mathcal{SEQ}1_{3/5}$	$\mathcal{SEQ}M_{3/5}$	$\mathcal{SEQ}1_{3/5}$	$\mathcal{SEQ}M_{3/5}$
Bs	5	5	5	5
Ep	100	100	100	100
Lr	$2/7e^{-4}$	$2e^{-4}$	$2/10e^{-4}$	$2e^{-4}$

TABLE X: Number of samples in each class for the training, validation, and test splits

Images					
Training sets		$\mathcal{SNG}1_3$		$\mathcal{SNG}1_5$	
Splits	**Class**	$\mathcal{SNG}1_3$	$\mathcal{SNG}2_3$	$\mathcal{SNG}1_5$	$\mathcal{SNG}2_5$
Train/Val	Mot	210/30	-	350/50	-
	Car	210/30	-	350/50	-
	Ped	210/30	-	350/50	-
Test	Mot	60	36	100	60
	Car	60	129	100	215
	Ped	60	180	100	300
Training sets		$\mathcal{SNG}M_3$		$\mathcal{SNG}M_5$	
Splits	**Class**	$\mathcal{SNG}1_3$	$\mathcal{SNG}2_3$	$\mathcal{SNG}1_5$	$\mathcal{SNG}2_5$
Train/Val	Mot	255	15	425	25
	Car	240	30	400	50
	Ped	225	45	375	75
Test	Mot	45	21	75	35
	Car	60	99	100	165
	Ped	75	135	125	225
Videos					
Training sets		$\mathcal{SEQ}1_3$		$\mathcal{SEQ}1_5$	
Splits	**Class**	$\mathcal{SEQ}1_3$	$\mathcal{SEQ}2_3$	$\mathcal{SEQ}1_5$	$\mathcal{SEQ}2_5$
Train/Val	Mot	70/10	-	70/10	-
	Car	70/10	-	70/10	-
	Ped	70/10	-	70/10	-
Test	Mot	20	12	20	12
	Car	20	43	20	43
	Ped	20	60	20	60
Training sets		$\mathcal{SEQ}M_3$		$\mathcal{SEQ}M_5$	
Splits	**Class**	$\mathcal{SEQ}1_3$	$\mathcal{SEQ}2_3$	$\mathcal{SEQ}1_5$	$\mathcal{SEQ}2_5$
Train/Val	Mot	85	5	85	5
	Car	80	10	80	10
	Ped	75	15	75	15
Test	Mot	15	7	15	7
	Car	20	33	20	33
	Ped	25	45	25	45

of the datasets $\mathcal{SNG}1_T$ and $\mathcal{SNG}2_T$, when adopting $\mathcal{SNG}1_T$ and $\mathcal{SNG}M_T$ as training sets. The first column represents the split set, the second column the target labels, and from the third to the last the table displays the number of samples for each class in each split of the datasets. The training and validation sets are grouped and separated by the '/' symbol. For the merged datasets, 30 and 50 random samples for each class were extracted for the validation splits when $T=3$ and $T=5$, respectively. As can be noticed, each class of the merged datasets, considering data from both scenarios, contains the same number of data for the training/validation splits, avoiding biased training due to class unbalancing.

B. Sequence RD Maps Classifiers Training Setup

Similarly to the single RD maps classifiers, the $S-DNN$ and $MN-DNN$ architectures were trained on the four datasets containing the sequences of RD maps (i.e., $\mathcal{SEQ}1_T$ and $\mathcal{SEQ}M_T$, with $T=\{3,5\}$). As previously, the last four layers of the pre-trained networks were unfrozen.

The second half of Table IX represents the hyper-parameters of the RD sequences classifiers. The RD maps sequence classifiers were trained by using the Adam optimizer as well. The second half of Table X, similarly to the first part, shows the number of samples of the datasets $\mathcal{SEQ}1_T$ and $\mathcal{SEQ}2_T$ when adopting $\mathcal{SEQ}1_T$ and $\mathcal{SEQ}M_T$ as training sets, for the classes and the splits. As before, each class of the merged datasets contains the same number of data for the training/validation splits.

VI. Generalization Performance Results

When introducing edge AI systems, it is crucial to assess two complementary aspects, i.e. classification performance in terms of accuracy and F1-score and the efficiency of the edge device. The F1-score serves as a metric that balances between precision and recall, making it a robust choice for addressing the class imbalance. In this study, the weighted metric was calculated by inversely weighting the class frequencies, thereby assigning greater significance to the classes with fewer instances, such as motorcycles and cars.

In this section, the results in terms of generalization accuracy and F1-score are presented.

A. Generalization performance on $S-CNN$ and $MN-CNN$ networks

The first experiment regards the evaluation of the generalization performance in terms of accuracy and F1-score of the $S-CNN$ and $MN-CNN$ classifiers trained with $\mathcal{SNG}1_T$ and $\mathcal{SNG}M_T$ datasets, according to Sec. III-A and Sec. V-A. Table XI shows the results. The table is divided into four parts based on the training sets and whether the voting mechanism has been applied. The first part reports the scores of the two networks trained with $\mathcal{SNG}1_T$ and tested on the test split of $\mathcal{SNG}1_T$ and the whole $\mathcal{SNG}2_T$. The second part shows the scores of the networks trained with $\mathcal{SNG}M_T$ and tested over the test splits of the merged datasets, $\mathcal{SNG}1_T$, and $\mathcal{SNG}2_T$. The third and the fourth present the results of the networks trained and tested on the datasets of the first two parts and adopting the voting mechanism.

The results highlight that the network designed from scratch, i.e. $S-CNN$, exhibits a higher accuracy and F1-score than the $MN-CNN$ architecture for all the cases. This is likely related to the larger number of parameters in these networks, which can increase the risk of overfitting. Additionally, it's possible that the frozen layers may not adapt optimally to the new task, resulting in slower convergence compared to networks built from scratch. Ultimately, the architectural design of the pre-trained networks may not be the best fit for the new task. A network designed from scratch can be tailored to the specific requirements of the task, potentially leading to faster convergence. Additionally, when the networks are trained with

$\mathcal{SNG}1_T$, they poorly generalize on $\mathcal{SNG}2_T$, even adopting the voting mechanism. This can be explained considering that the two scenarios suffer from different distributions of the clutter. Moreover, the relative positions between the targets and the radar are different among the scenarios, and this affects the patterns on the RD maps. On the other hand, including a small amount of data from $\mathcal{SNG}2_T$ during training, the architectures achieve good performance across all the datasets. These performance are further improved by applying the voting mechanism. The $S-CNN$ networks always exceed the accuracies and F1-scores achieved by $MN-CNN$ models. When the voting mechanism is not adopted, the $S-CNN$ architecture trained with $\mathcal{SNG}M_3$ exhibits the best generalization performance, even though the F1-score on $\mathcal{SNG}2_3$ is lower than that achieved with $\mathcal{SNG}2_5$. This difference in F1-scores can be attributed to the varying class weights resulting from the larger sample size in $\mathcal{SNG}2_5$. While, when the mechanism is used, the best generalization performance are achieved by the $S-CNN_{Vote}$ network trained with $\mathcal{SNG}M_5$, meaning that with a higher number of RD maps collected for each moving target, the voting mechanism is more effective.

TABLE XI: Generalization performances of $S-CNN$ and $MN-CNN$

	$\boldsymbol{S-CNN}$		$\boldsymbol{MN-CNN}$	
Train / Test	$\mathcal{SNG}1_3$	$\mathcal{SNG}1_5$	$\mathcal{SNG}1_3$	$\mathcal{SNG}1_5$
$\mathcal{SNG}1_{3/5}$	90.6 (90.5)	91.7 (91.7)	77.2 (76.5)	80.7 (80.2)
$\mathcal{SNG}2_{3/5}$	67.5 (70.2)	70.1 (75.2)	63.9 (66.9)	65.6 (65.2)
Train / Test	$\mathcal{SNG}M_3$	$\mathcal{SNG}M_5$	$\mathcal{SNG}M_3$	$\mathcal{SNG}M_5$
$\mathcal{SNG}M_{3/5}$	90.3 (89.1)	87.7 (87.5)	81.1 (78.4)	79.7 (76.8)
$\mathcal{SNG}1_{3/5}$	91.7 (90.7)	88.7 (87.9)	83.9 (81.3)	79.7 (77.0)
$\mathcal{SNG}2_{3/5}$	89.4 (86.8)	87.1 (88.4)	79.2 (77.2)	79.8 (79.0)
	$\boldsymbol{S-CNN_{Vote}}$		$\boldsymbol{MN-CNN_{Vote}}$	
Train / Test	$\mathcal{SNG}1_3$	$\mathcal{SNG}1_5$	$\mathcal{SNG}1_3$	$\mathcal{SNG}1_5$
$\mathcal{SNG}1_{3/5}$	93.3 (93.3)	98.3 (98.3)	83.3 (82.1)	83.3 (82.7)
$\mathcal{SNG}2_{3/5}$	70.6 (73.1)	77.6 (82.8)	69.4 (73.5)	72.9 (72.5)
Train / Test	$\mathcal{SNG}M_3$	$\mathcal{SNG}M_5$	$\mathcal{SNG}M_3$	$\mathcal{SNG}M_5$
$\mathcal{SNG}M_{3/5}$	94.5 (94.4)	95.2 (95.9)	88.3 (86.1)	92.4 (90.5)
$\mathcal{SNG}1_{3/5}$	96.7 (96.7)	96.7 (96.7)	88.3 (86.4)	91.7 (90.0)
$\mathcal{SNG}2_{3/5}$	92.9 (90.8)	94.1 (95.5)	88.2 (87.3)	94.1 (95.2)

B. Generalization performance on $S-DNN$ and $MN-DNN$ networks

The second experiment regards the evaluation of the generalization performance in terms of accuracy and F1-score of the $S-DNN$ and $MN-DNN$ classifiers trained with $\mathcal{SEQ}1_T$ and $\mathcal{SEQ}M_T$ datasets, according to Sec. III-B and Sec. V-B.

The results show a similar trend as in Table XI: the $S-DNN$ network achieves better accuracy and F1-score than the $MN-DNN$ model across all the tested datasets. Both architectures, when the networks are trained with $\mathcal{SEQ}1_T$, poorly generalize on $\mathcal{SEQ}2_T$. On the other hand, by training the networks with the $\mathcal{SEQ}M_T$ datasets, the classifiers generalize well across all the datasets. In particular, $S-DNN$ achieves the best performance concerning the $MN-DNN$ network and the single RD map classifiers, i.e., $S-CNN$ trained with the $\mathcal{SNG}M_T$ datasets. Comparing the $S-DNN$ classifiers with $S-CNN_{Vote}$ trained with the merged datasets, the outcomes highlight that, when collecting 3 RD maps for each target, the voting mechanism is more effective than adopting the sequence-based classifier. Instead, when employing 5 maps, the $S-DNN$ network outperforms all the other classifiers in terms of accuracy. However, the F1-score on $\mathcal{SEQ}2M_5$ is lower than the achieved with the voting mechanism even though the accuracy remains the same. As mentioned earlier, this difference can be attributed to the variation in class weights. Specifically, the $S-DNN$ misclassified one motorcycle (the less numerous class) as a car, three cars as motorcycles, and one car as a pedestrian. On the other hand, the $S-CNN_{Vote}$ misclassified two cars as motorcycles, one car as a pedestrian, and two pedestrians as motorcycles. Importantly, $S-CNN_{Vote}$ correctly predicted all the motorcycles, which is the class with the highest weight in the computation of the F1-score.

In general, except one case, considering the time dependency between a sufficient number of RD maps extracted from a moving target leads to a higher generalization performance for single-map classifiers, even though the voting mechanism increases their performance.

TABLE XII: Generalization performance of $S-DNN$ and $MN-DNN$

	$\boldsymbol{S-DNN}$		$\boldsymbol{MN-DNN}$	
Train / Test	$\mathcal{SEQ}1_3$	$\mathcal{SEQ}1_5$	$\mathcal{SEQ}1_3$	$\mathcal{SEQ}1_5$
$\mathcal{SEQ}1_{3/5}$	95.0 (94.7)	95.0 (95.0)	85.0 (84.2)	81.6 (80.8)
$\mathcal{SEQ}2_{3/5}$	71.3 (72.9)	70.4 (74.2)	68.9 (72.1)	67.8 (70.6)
Train / Test	$\mathcal{SEQ}M_3$	$\mathcal{SEQ}M_5$	$\mathcal{SEQ}M_3$	$\mathcal{SEQ}M_5$
$\mathcal{SEQ}M_{3/5}$	93.8 (91.8)	96.6 (96.0)	84.1 (80.3)	87.6 (87.4)
$\mathcal{SEQ}1_{3/5}$	93.3 (92.8)	100 (100)	88.3 (85.4)	93.3 (92.8)
$\mathcal{SEQ}2_{3/5}$	94.1 (87.6)	94.1 (91.1)	81.2 (78.0)	83.5 (80.8)

C. Train and Validation Losses and Accuracies

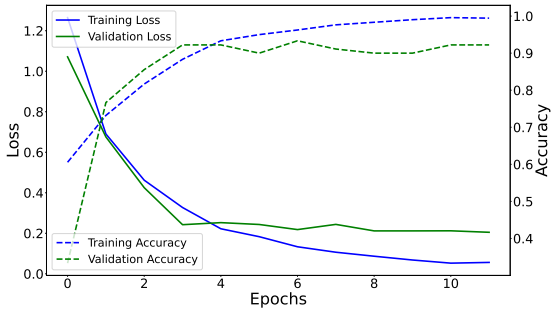
Figure 4 displays the training and validation accuracies and losses for the $S-CNN$ and $S-DNN$ networks, which were trained on the $\mathcal{SNG}M_T$ and $\mathcal{SEQ}M_T$ (with $T=\{3,5\}$), respectively, due to their highest performance in the previous analysis. In each plot, accuracy is represented by dashed lines, while loss is indicated by continuous lines. The training set data is depicted in blue, and the validation set data is in red, with the number of epochs shown on the x-axis. As outlined in Section V, we implemented an early stopping criterion with the patience of five epochs based on the validation loss.

It is worth highlighting that, despite the relatively small size of the training dataset, these figures demonstrate a noteworthy convergence of training and validation scores across all plots.

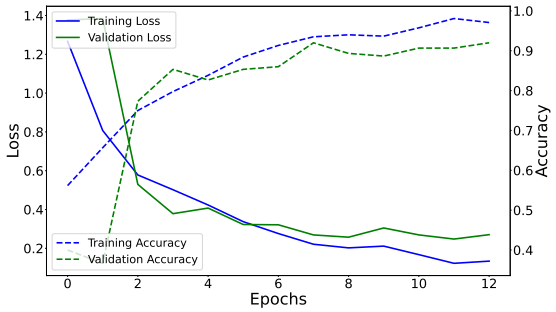
D. Comparison with other approaches

Recently, other techniques have been proposed to classify moving targets in similar operational scenarios. In [66], authors proposed an SVM to classify the single targets achieving 95% accuracy in the three-class classification problem in the

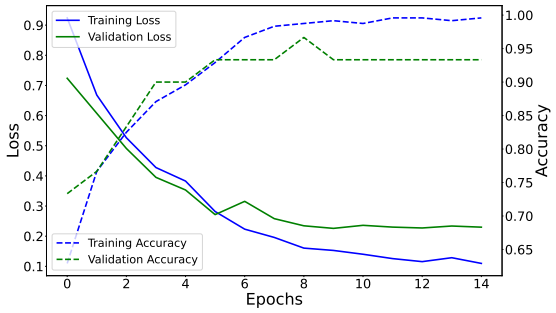
(a) S-CNN on $\mathcal{SNGM}_3$



(b) S-CNN on $\mathcal{SNGM}_5$



(c) S-DNN on $\mathcal{SEQM}_3$



(d) S-DNN on $\mathcal{SEQM}_5$

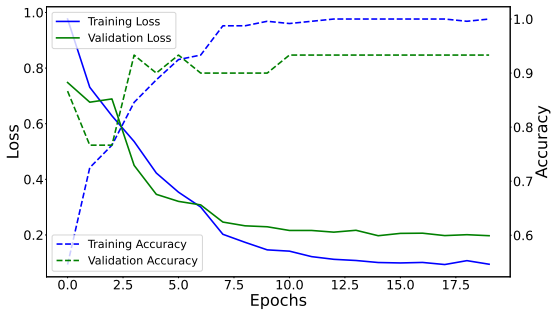


Fig. 4: Train/Validation losses and accuracy of $S-CNN$ and $S-DNN$ trained on $\mathcal{SNGM}_T$ and $\mathcal{SEQM}_T$, respectively.

first scenario, misclassifying a motorcycle as a pedestrian. In [28], the authors obtained an overall accuracy slightly lower than 92% in the merged scenarios adopting a K-NN, misclassifying two cars and a pedestrian as motorcycles, and one motorcycle as a pedestrian. For both approaches, the trucks that represent the fourth class have been considered cars. Table XIII compares the performance of [28], [66] with the proposal in the first scenario and the merged ones. For the proposed approach, the best model has been chosen, i.e., $S-DNN$ trained with $\mathcal{SEQM}_5$. The columns report the number of correctly classified samples per class for the total number of data in the class. The last column reports the accuracies and the F1-scores computed over the three classes. The proposed approach, in the first scenario, achieved 100% accuracy outperforming the SVM. In the merged scenarios, the DNN presents a slight deterioration with respect to K-NN in the classification of the cars, whereas it misclassifies a motorcycle as a car, and correctly predicts all the pedestrians, outperforming the K-NN overall accuracy. Finally, it is worth remarking that the proposed approach does not require defining and extracting a set of features to be used for classification, but instead, the DNNs computed them automatically through the input convolutional layers.

TABLE XIII: Performance comparison with other techniques

Approaches	Cars	Moto	Ped	Overall
SVM [66] (first scenario)	10/10	4/5	5/5	95 (93.3)
K-NN [28] (merged scenarios)	22/24	11/12	11/12	91.7 (91.8)
DNN (first scenario)	**20/20**	**15/15**	**25/25**	**100 (100)**
DNN (merged scenarios)	**49/53**	**21/22**	**70/70**	**96.6 (95.9)**

E. Generalization in a new environment

We employed the proposed network architectures for classifying RD maps collected in another environment [67]. Specifically, RD maps from drones, cars, and people, have been acquired in real outdoor scenarios using an FMCW radar. From the original dataset, which contains more than 17000 data, we collected 1500 maps from each class maintaining the time correlation between the data. Subsequently, we created new datasets tailored to our network architectures. To adapt the $S-CNN$ and $S-DNN$ models, which were originally trained on $\mathcal{SNGM}_{3/5}$ and $\mathcal{SEQM}_{3/5}$ to new data, we performed fine-tuning. The new datasets will be denoted as $\mathcal{SNGN}_{3/5}$ and $\mathcal{SEQN}_{3/5}$. For the fine-tuning process, we partitioned the new datasets into training and testing subsets, allocating 70% of the data for training and the remaining 30% for testing. Table XIV provides a breakdown of the number of data points in each split within the new datasets. Table XV presents the results, where the highest performance is attained by the $S-DNN$ model in conjunction with $\mathcal{SEQN}_3$, showing an accuracy and F1-score of 96.9%. It's worth noting that, in [67], the authors achieved a remarkable 99.5% accuracy and F1-score by employing a network with over 3.8 million parameters. On the other hand, the proposed network achieves strong generalization on new problems while still maintaining a low computational cost (i.e., 696K parameters).

VII. Edge Deployment Results

For a real-time application, the classifier deployed on the edge must not only achieve the highest possible accuracy but rather present a trade-off between classification accuracy and computational cost. In this paper, the computational cost is measured as inference time and energy consumption. The models were deployed on a Raspberry Pi4 through the TFLite

TABLE XIV: Number of samples in each class for the training, validation, and test splits for the new datasets

Splits	Class	$\mathcal{SNGN}_3$	$\mathcal{SNGN}_5$	$\mathcal{SEQN}_3$	$\mathcal{SEQN}_5$
Train	Car	1050	1050	350	210
	Dro	1050	1050	350	210
	Peo	1050	1050	350	210
Test	Car	450	450	150	90
	Dro	450	450	150	90
	Peo	450	450	150	90

TABLE XV: Generalization performance of $S-CNN$, $S-CNN_{Vote}$, and $S-DNN$ on the new dataset

Architectures	Dataset	Accuracy	F1-Score
$S-CNN$	$\mathcal{SNGN}_3$	88.8	88.7
	$\mathcal{SNGN}_5$	92.1	92.1
$S-CNN_{Vote}$	$\mathcal{SNGN}_3$	92.9	92.8
	$\mathcal{SNGN}_5$	95.2	95.1
$S-DNN$	$\mathcal{SEQN}_3$	96.9	96.9
	$\mathcal{SEQN}_5$	95.9	95.9

toolbox provided by Tensorflow. The energy consumption was estimated using a USB multimeter that was plugged into the power supply of the edge device while running the inference and it was averaged on the number of tested data. According to the generalization accuracy results presented in the previous section, only the networks based on the networks designed from scratch and trained with the merged datasets were taken into consideration, i.e. $S-CNN$ and $S-CNN_{Vote}$ trained with $\mathcal{SNGM}_{3/5}$, and $S-DNN$ trained with $\mathcal{SEQM}_{3/5}$. Table XVI shows the results. The first column represents the classifiers, the second whether the voting mechanism is applied or not, the third the names of the tested datasets, the fourth the accuracies achieved by the classifiers on the tested datasets, the fifth the number of parameters of the networks, the sixth the inference time, and the last the energy consumption.

As a result, if the computational cost is a hard constraint, the best choice relies on the $S-CNN$ models trained with the $\mathcal{SNGM}_3$ dataset, achieving higher accuracy than the $S-CNN$ trained with $\mathcal{SNGM}_5$. On the contrary, if the accuracy is more relevant for the application, the $S-DNN$ network trained with the $\mathcal{SEQM}_5$ dataset is the best option.

TABLE XVI: System assessment on Raspberry Pi4

Best Classifier	Vote	Tested Dataset	Acc.	Num of Params	Time (ms)	Energy (J)
$S-CNN$ Train: $\mathcal{SNGM}_3$		$\mathcal{SNGM}_3$	90.3	673K	175	0.94
		$\mathcal{SNG1}_3$	91.7			
		$\mathcal{SNG2}_3$	89.4			
	✓	$\mathcal{SNGM}_3$	94.5		523	2.86
		$\mathcal{SNG1}_3$	96.7			
		$\mathcal{SNG2}_3$	94.1			
$S-DNN$ Train: $\mathcal{SEQM}_3$		$\mathcal{SEQM}_3$	93.8	696K	415	2.25
		$\mathcal{SEQ1}_3$	93.3			
		$\mathcal{SEQ2}_3$	94.1			
$S-CNN$ Train: $\mathcal{SNGM}_5$		$\mathcal{SNGM}_5$	87.7	673K	174	0.94
		$\mathcal{SNG1}_5$	88.7			
		$\mathcal{SNG2}_5$	87.1			
	✓	$\mathcal{SNGM}_5$	95.9		914	4.96
		$\mathcal{SNG1}_5$	96.7			
		$\mathcal{SNG1}_5$	94.1			
$S-DNN$ Train: $\mathcal{SEQM}_5$		$\mathcal{SEQM}_5$	96.6	696K	678	3.65
		$\mathcal{SEQ1}_5$	100			
		$\mathcal{SEQ2}_5$	94.1			

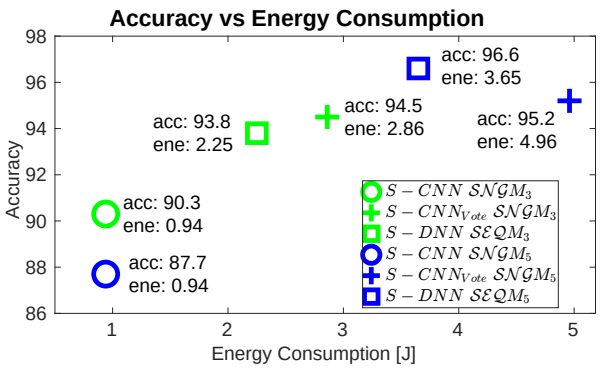


Fig. 5: Accuracy vs. Energy Consumption of the models trained on the merged dataset.

Two solutions present a valuable trade-off between accuracy and computational cost: $S-CNN_{Vote}$ trained with the $\mathcal{SNGM}_3$ dataset, and $S-DNN$ trained with the $\mathcal{SEQM}_3$ dataset. As last consideration, $S-CNN_{Vote}$ trained with the $\mathcal{SNGM}_5$ presents the highest computational cost both in terms of inference time and energy consumption, and the second-best accuracy, thus not representing a suitable solution for the classification of moving targets on edge. The results are also visualized in Fig. 5, considering only the test split of the merged datasets. The figure shows the accuracy vs. energy consumption of the models deployed on edge. All the considerations made for the table holds also for the figure.

Eventually, Figure 6 presents the confusion matrices of the four models that, respectively, achieve: the lowest computational cost (i.e., $S-CNN$ trained with the $\mathcal{SNGM}_3$ dataset and represented by a red circle in Fig. 5), the highest accuracy (i.e., $S-DNN$ trained with $\mathcal{SEQM}_5$ and represented by a blue square in Fig.5), and the two models that present the best trade-off between accuracy and computational cost (i.e., $S-CNN_{Vote}$ trained with the $\mathcal{SNGM}_3$ dataset and represented with a red '+' in Fig. 5, $S-DNN$ trained with the $\mathcal{SEQM}_3$ dataset and represented with a red square in Fig. 5).

Based on the results of Figures 6, it is worth highlighting that car and motorcycle classes have, in general, a higher miss-classification rate compared to pedestrians. This is logical as both classes present a similar motion model with comparable speeds (at least in this study) and a higher metallic reflective cross-sectional area. On the opposite, pedestrians' gait is considered different [68] because the movement of the legs and arms can produce different Doppler frequencies in the RD maps [35]. This phenomenon highly affects the temporal signature resulting from the movement of pedestrians, and consequently makes it much different from the one of rigid bodies.

VIII. Conclusion

In this paper, a range-Doppler (RD) maps sequence-based DNN architecture for radar ground-moving targets' classification on edge has been proposed. The classifier, designed by scratch, combined a convolutional neural network (CNN) with a recurrent neural network (RNN) to classify moving targets

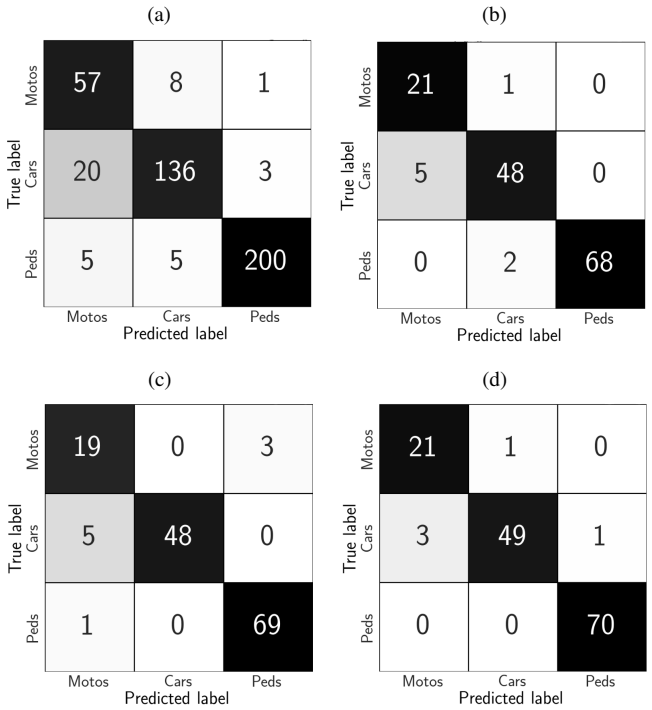


Fig. 6: Confusion matrices of a) $S-CNN$ architecture trained on $\mathcal{SNG}M_3$, b) $S-CNN_{Vote}$ architecture trained on $\mathcal{SNG}M_3$, c) $S-DNN$ architecture trained on $\mathcal{SEQ}M_3$, and d) $S-DNN$ architecture trained on $\mathcal{SEQ}M_5$.

depending on their time-varying signatures. Two datasets were used to train the DNN module representing two real and diverse cluttered environments. In particular, the radar range-Doppler maps collected on three kinds of moving targets, i.e., pedestrians, motorcycles, and cars were used as inputs to the DNN. The data have been collected by a low-cost FMCW radar plugged into a Raspberry Pi powered by an external battery. The device performed data preprocessing to transform the signal from the radar raw data into RD maps. The generalization accuracy of the proposed DNN was computed and compared with three different DNN models: RD maps sequence-based classifier enclosing a pre-trained CNN, and two single-RD map classifiers with the same structure as the CNN architectures used in the RD maps sequence-based classifiers. A voting mechanism was also proposed to enhance the performance of the single-map classifiers. The single map (with and without voting) and maps sequence classifiers were tested on 3 and 5 RD maps collected from the same target. Eventually, the architectures were deployed on the Raspberry Pi to find the model that achieves the best trade-off between accuracy and computational cost measured as inference time and energy consumption. The results showed that the DNN designed from scratch and trained on the sequences of 5 RD maps per target achieved the best accuracy (96.6%) but demanded 678ms of inference time and 3.65J of energy consumption. On the other hand, the CNN designed from scratch and trained on single maps (3 maps per target) presented the lowest computational cost (175ms of inference and 0.94J of energy). A good trade-off between accuracy and computational cost has been achieved by the CNN designed from scratch and trained on single maps (3 maps per target) applying the voting mechanism (523ms and 2.86J) and by the DNN designed from scratch and trained on the sequences of 3 RD maps per target (415ms and 2.25J). The developed approach based on DNNs classification is able to provide higher accuracies than other techniques based on classic ML algorithms, at least when dealing with single-target identification. The next step will be implementing the proposed methodology to solve the multi-target recognition enhancing the classification results achieved in [27], [28].

REFERENCES

[1] P. Withington, H. Fluhler, and S. Nag, "Enhancing homeland security with advanced uwb sensors," *IEEE Microw. Mag.*, vol. 4, no. 3, pp. 51–58, 2003.

[2] F. Colone *et al.*, "Wifi-based passive isar for high-resolution cross-range profiling of moving targets," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 6, pp. 3486–3501, 2014.

[3] G. Gennarelli *et al.*, "Multiple extended target tracking for through-wall radars," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 12, pp. 6482–6494, 2015.

[4] A. Randazzo *et al.*, "A two-step inverse-scattering technique in variable-exponent lebesgue spaces for through-the-wall microwave imaging: Experimental results," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 9, pp. 7189–7200, 2021.

[5] B. Pottier, L. Rasolofondraibe, and S. Kerroumi, "Pedestrian detection strategy in urban area: Capacitance probes and pedestrians' signature," *IEEE Sensors J.*, vol. 17, no. 17, pp. 5663–5668, 2017.

[6] T. Gandhi and M. M. Trivedi, "Pedestrian protection systems: Issues, survey, and challenges," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 3, pp. 413–430, 2007.

[7] A. Rasouli and J. K. Tsotsos, "Autonomous vehicles that interact with pedestrians: A survey of theory and practice," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 900–918, 2020.

[8] R. Zhang and S. Cao, "Extending reliability of mmwave radar tracking and detection via fusion with camera," *IEEE Access*, vol. 7, pp. 137 065–137 079, 2019.

[9] P.-J. Wang *et al.*, "A channel awareness vehicle detector," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 2, pp. 339–347, 2010.

[10] S. Saponara and B. Neri, "Radar sensor signal acquisition and multidimensional fft processing for surveillance applications in transport systems," *IEEE Trans. Instrum. Meas.*, vol. 66, no. 4, pp. 604–615, 2017.

[11] A. Mukhtar, L. Xia, and T. B. Tang, "Vehicle detection techniques for collision avoidance systems: A review," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 5, pp. 2318–2338, 2015.

[12] C. Ding *et al.*, "Continuous human motion recognition with a dynamic range-doppler trajectory method based on fmcw radar," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 9, pp. 6821–6831, 2019.

[13] D.-H. Jung *et al.*, "Sparse scene recovery for high-resolution automobile fmcw sar via scaled compressed sensing," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 12, pp. 10 136–10 146, 2019.

[14] Y. Nan, X. Huang, and Y. J. Guo, "A millimeter-wave gcw-sar based on deramp-on-receive and piecewise constant doppler imaging," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 1, pp. 680–690, 2020.

[15] C. Q. Mayoral *et al.*, "Water content continuous monitoring of grapevine xylem tissue using a portable low-power cost-effective fmcw radar," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 8, pp. 5595–5605, 2019.

[16] B. Vandersmissen *et al.*, "Indoor person identification using a low-power fmcw radar," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 7, pp. 3941–3952, 2018.

[17] B. Yektakhah and K. Sarabandi, "All-directions through the wall imaging using a small number of moving omnidirectional transceivers," in *Proc. IEEE Inter. Geosci. Remote Sens. Symposium (IGARSS)*, 2017, pp. 2416–2419.

[18] A. Ganis *et al.*, "A portable 3-d imaging fmcw mimo radar demonstrator with a 24×24 antenna array for medium-range applications," *IEEE Trans. Geosci. Remote Sens.*, vol. 56, no. 1, pp. 298–312, 2018.

[19] D. Jasteh *et al.*, "Experimental low-terahertz radar image analysis for automotive terrain sensing," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 4, pp. 490–494, 2016.

[20] F. Fioranelli, S. Salous, and X. Raimundo, "Frequency-modulated interrupted continuous wave as wall removal technique in through-the-wall imaging," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 10, pp. 6272–6283, 2014.

[21] A. Anghel *et al.*, "Short-range wideband fmcw radar for millimetric displacement measurements," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 9, pp. 5633–5642, 2014.

[22] R. Wang *et al.*, "Motion compensation for high-resolution automobile fmcw sar," *IEEE Geosci. Remote Sens. Lett.*, vol. 10, no. 5, pp. 1157–1161, 2013.

[23] F. Alimenti *et al.*, "Noncontact measurement of river surface velocity and discharge estimation with a low-cost doppler radar sensor," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 7, pp. 5195–5207, 2020.

[24] T. Mitomo *et al.*, "A 77 ghz 90 nm cmos transceiver for fmcw radar applications," *IEEE J. Solid-State Circuits*, vol. 45, no. 4, pp. 928–937, 2010.

[25] J. M. Munoz-Ferreras *et al.*, "Traffic surveillance system based on a high-resolution radar," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 6, pp. 1624–1633, 2008.

[26] Z. Wang *et al.*, "A Review of Vehicle Detection Techniques for Intelligent Vehicles," *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–21, 2022.

[27] A. Rizik *et al.*, "Cost-efficient fmcw radar for multi-target classification in security gate monitoring," *IEEE Sensors J.*, vol. 21, no. 18, pp. 20 447–20 461, 2021.

[28] E. Tavanti *et al.*, "A short-range fmcw radar-based approach for multi-target human-vehicle detection," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–16, 2022.

[29] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[30] A. Wrabel, R. Graef, and T. Brosch, "A Survey of Artificial Intelligence Approaches for Target Surveillance With Radar Sensors," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 36, no. 7, pp. 26–43, Jul. 2021.

[31] Y. Kim *et al.*, "Human Detection Based on Time-Varying Signature on Range-Doppler Diagram Using Deep Neural Networks," *IEEE Geosci. Remote Sens. Lett.*, vol. 18, no. 3, pp. 426–430, Mar. 2021.

[32] R. Perez *et al.*, "Single-Frame Vulnerable Road Users Classification with a 77 GHz FMCW Radar Sensor and a Convolutional Neural Network," in *Proc. of the 19th Int. Radar Symposium (IRS)*. Bonn: IEEE, Jun. 2018, pp. 1–10.

[33] S. Gupta *et al.*, "Target Classification by mmWave FMCW Radars Using Machine Learning on Range-Angle Images," *IEEE Sensors J.*, vol. 21, no. 18, pp. 19 993–20 001, Sep. 2021.

[34] A. Mohanna *et al.*, "A convolutional neural network-based method for discriminating shadowed targets in frequency-modulated continuous-wave radar systems," *Sensors*, vol. 22, no. 3, p. 1048, 2022.

[35] V. Chen *et al.*, "Micro-doppler effect in radar: phenomenon, model, and simulation study," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 42, no. 1, pp. 2–21, 2006.

[36] L. Du *et al.*, "Micro-doppler feature extraction based on time-frequency spectrogram for ground moving targets classification with low-resolution radar," *IEEE Sensors J.*, vol. 16, no. 10, pp. 3756–3763, 2016.

[37] A. Angelov *et al.*, "Practical classification of different moving targets using automotive radar and deep neural networks," *IET Radar, Sonar & Navigation*, vol. 12, no. 10, pp. 1082–1089, 2018.

[38] Z. Chen *et al.*, "Personnel Recognition and Gait Classification Based on Multistatic Micro-Doppler Signatures Using Deep Convolutional Neural Networks," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 5, pp. 669–673, May 2018.

[39] 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.

[40] X. Bai *et al.*, "Radar-Based Human Gait Recognition Using Dual-Channel Deep Convolutional Neural Network," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 12, pp. 9767–9778, Dec. 2019.

[41] X. Yao, X. Shi, and F. Zhou, "Human Activities Classification Based on Complex-Value Convolutional Neural Network," *IEEE Sensors J.*, vol. 20, no. 13, pp. 7169–7180, Jul. 2020.

[42] S. Abdulatif *et al.*, "Micro-doppler based human-robot classification using ensemble and deep learning approaches," in *Proc. of the IEEE Radar Conf. (RadarConf18)*. IEEE, 2018, pp. 1043–1048.

[43] R. Prophet *et al.*, "Pedestrian classification with a 79 ghz automotive radar sensor," in *Proc. of the 19th Int. Radar Symposium (IRS)*, 2018, pp. 1–6.

[44] O. Schumann *et al.*, "Scene understanding with automotive radar," *IEEE Trans. Intell. Veh.*, vol. 5, no. 2, pp. 188–203, 2020.

[45] M. Hermans and B. Schrauwen, "Training and analysing deep recurrent neural networks," *Advances in Neural Inf. Proc. Sys.*, vol. 26, 2013.

[46] R. Ravindran, M. J. Santora, and M. M. Jamali, "Multi-Object Detection and Tracking, Based on DNN, for Autonomous Vehicles: A Review," *IEEE Sensors J.*, vol. 21, no. 5, pp. 5668–5677, Mar. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9274366/

[47] N. Scheiner *et al.*, "Radar-based Road User Classification and Novelty Detection with Recurrent Neural Network Ensembles," in *Proc. of the IEEE Intell. Vehicles Symposium (IV)*. Paris, France: IEEE, Jun. 2019, pp. 722–729.

[48] S. Chen *et al.*, "Target Classification Using the Deep Convolutional Networks for SAR Images," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 8, pp. 4806–4817, Aug. 2016.

[49] M. Pan *et al.*, "Radar HRRP Target Recognition Model Based on a Stacked CNN–Bi-RNN With Attention Mechanism," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–14, 2022.

[50] X. Bai *et al.*, "Sequence SAR Image Classification Based on Bidirectional Convolution-Recurrent Network," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 11, pp. 9223–9235, Nov. 2019.

[51] J. Ding *et al.*, "Video SAR Moving Target Indication Using Deep Neural Network," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 10, pp. 7194–7204, Oct. 2020.

[52] R. Xue, X. Bai, and F. Zhou, "Spatial–Temporal Ensemble Convolution for Sequence SAR Target Classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 2, pp. 1250–1262, Feb. 2021.

[53] S.-L. Jeng, W.-H. Chieng, and H.-P. Lu, "Estimating speed using a side-looking single-radar vehicle detector," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 2, pp. 607–614, Apr. 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6636147/

[54] E. Giusti and M. Martorella, "Range doppler and image autofocusing for FMCW inverse synthetic aperture radar," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 47, no. 4, pp. 2807–2823, 2011. [Online]. Available: http://ieeexplore.ieee.org/document/6034666/

[55] M. I. Skolnik, *Introduction to Radar Systems*. McGraw-Hill, 2001.

[56] M. A. Richards, *Fundamentals of radar signal processing*. McGraw-Hill, 2014.

[57] M. I. Skolnik, *Radar handbook*. McGraw-Hill Education, 2008.

[58] M. Kronauge and H. Rohling, "New chirp sequence radar waveform," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 50, no. 4, pp. 2870–2877, 2014.

[59] S. Heuel and H. Rohling, "Pedestrian classification in automotive radar systems," in *2012 13th Int. Radar Symposium*. IEEE, 2012, pp. 39–44.

[60] ——, "Pedestrian recognition based on 24 GHz radar sensors," in *Ultra-Wideband Radio Technologies for Communications, Localization and Sensor Applications*, R. Thomä, Ed. InTech, Mar. 2013.

[61] M. Sandler *et al.*, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proc. of the IEEE Conf. on Comp. Vision and Pattern Rec.*, 2018, pp. 4510–4520.

[62] AG, Infineon Technologies. Demo distance2go. [Online]. Available: https://www.infineon.com/cms/en/product/evaluation-boards/demo-distance2go/

[63] Y. Ma *et al.*, "Spatial Perception of Tagged Cargo Using Fused RFID and CV Data in Intelligent Storage," *IEEE Internet of Things Journal*, vol. 10, no. 2, pp. 1574–1587, Jan. 2023. [Online]. Available: https://ieeexplore.ieee.org/document/9903839/

[64] Y. Yu *et al.*, "A review of recurrent neural networks: Lstm cells and network architectures," *Neural computation*, vol. 31, no. 7, pp. 1235–1270, 2019.

[65] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[66] A. Rizik *et al.*, "Single target recognition using a low-cost fmcw radar based on spectrum analysis," in *2020 27th IEEE Int. Conf. on Electron., Circuits and Syst. (ICECS)*. IEEE, 2020, pp. 1–4.

[67] I. Roldan *et al.*, "Dopplernet: A convolutional neural network for recognising targets in real scenarios using a persistent range–doppler radar," *IET Radar, Sonar & Navigation*, vol. 14, no. 4, pp. 593–600, 2020.

[68] J. Zhang, T. E. Lockhart, and R. Soangra, "Classifying lower extremity muscle fatigue during walking using machine learning and inertial sensors," *Annals of biomedical engineering*, vol. 42, no. 3, pp. 600–612, 2014.