Deep Learning for Accurate Indoor Human Tracking with a mm-Wave Radar

Jacopo Pegoraro[†*], Domenico Solimini[‡], Federico Matteo[‡],
Enver Bashirov[†], Francesca Meneghello[†] and Michele Rossi[†]

***Abstract*—We address the use of backscattered mm-wave radio signals to track humans as they move within indoor environments. The common approach in the literature leverages the extended Kalman filter (EKF) method, which however undergoes a severe performance degradation when the system evolution model is highly non-linear or presents long-term time dependencies among the system states. In this work, we propose an original model-free tracking procedure based on denoising autoencoders and sequence-to-sequence neural networks, showing its superior performance with respect to state-of-the-art methods. Our architecture can be trained in either a supervised or unsupervised manner, trading tracking accuracy for flexibility. The proposed system is tested on our own measurements, obtained with a 77 GHz radar on single and multiple subjects simultaneously moving in an indoor space. The results are compared against the ground truth trajectories from a motion tracking system, obtaining average tracking errors as low as 12 cm.**

***Index Terms*—mm-wave radar, indoor sensing, human tracking, denoising autoencoders, sequence-to-sequence autoencoders**

I. Introduction

RADAR devices for indoor environments are gaining a growing interest. Recent studies have demonstrated the possibility of exploiting the properties of the reflected radar signal to infer the presence, position, and activity of human targets in indoor spaces [1]–[3]. This approach is a sound alternative to traditional camera-based sensing systems, as it preserves the privacy of the users, i.e., no visual representation of the scene is collected, and is robust to poor light conditions [4]. The use of radio waves in the mm-wave frequency band allows the estimation of the target distance with high resolution, in the order of a few centimeters. Although this increased sensitivity makes millimeter-wave (mm-wave) prone to disturbances and clutter effects from the radio environment, the use of data-driven deep learning methods has recently emerged as a viable solution to these problems, enabling person identification [2] and activity recognition [3] tasks.

This work has been supported, in part, by MIUR (Italian Ministry of Education, University and Research) through the initiative "Departments of Excellence" (Law 232/2016) and by the EU MSCA ITN project MINTS "MIllimeter-wave NeTworking and Sensing for Beyond 5G" (grant no. 861222).

† These authors are with the Department of Information Engineering, University of Padova.

‡ These authors are with the Department of Mathematics, University of Padova.

* Corresponding author e-mail: pegoraroja@dei.unipd.it

978-1-7281-8942-0/20/$31.00 ©2020 IEEE

In the present article, we focus on the problem of tracking people as they move within an indoor environment, using the backscattered signal from a mm-wave frequency-modulated continuous-wave (FMCW) radar. Our aim is to obtain accurate positioning information of the targets in the physical space. So far, the few available solutions to this problem [1], [5], have relied on a Bayesian approach using the extended Kalman filter (EKF) method [6]. Kalman filter (KF), however, is suitable for systems that follow a linear evolution model with Gaussian noise. The extension to the non-linear and non-Gaussian case (i.e., the EKF or the unscented Kalman filter (UKF), [6]) is often problematic, especially in highly non-linear models.

In this work, we alternatively use deep neural network (NN) architectures to sequentially estimate the location of human targets in indoor spaces: we leverage denoising autoencoders (DAE) [7] and sequence-to-sequence denoising autoencoders (S2S) [8] to sequentially learn the best parameters from the data, not requiring any preliminary assumptions on the nature of the system evolution, nor on the noise process. S2S architectures, moreover, are capable of modeling long time dependencies.

Our main contributions are summarized next.

1) We propose two novel deep learning architectures for the task of tracking human targets in indoor spaces with a mm-wave FMCW radar, based on a DAE and a S2S, respectively. The average tracking error is as low as 0.12 m for the single target case and 0.21 m for the multi-target one.
2) We evaluate our position tracking system on a challenging and realistic dataset collected in a room including furniture, metallic objects, and other people, emulating real-life conditions.
3) We train the proposed tracking system in supervised and unsupervised manners. For the former, the ground truth positions of the targets are provided at training time, while in the latter only the radar measurements are used. In both cases, our approach outperforms Bayesian methods such as EKF and UKF under several metrics.

The rest of the article is organized as follows. In Section II, the FMCW radar signal model and the detection procedure are described. In Section III we present the tracking problem discussing the novelty of our approach. The signal processing workflow is presented in Section IV. In Section V, experimental results are discussed, while concluding remarks are given in Section VI.

II. FMCW RADAR MODEL

A multiple-input multiple-output (MIMO) FMCW radar allows the joint estimation of the distance, the angular position and the radial velocity of the target with respect to the radar device. This is achieved by transmitting sequences of *chirps*, i.e., sinusoidal waves with frequency that varies in time, and measuring the frequency shift of the backscattered signal at the multiple receiver antennas.

In this article, we use a linear FMCW (LFMCW) radar (the INRAS RadarLog) with one transmitting antenna and $M = 16$ receiving antennas. The frequency of the transmitted chirp signal (TX) is linearly increased from a base value of $f_o = 77$ GHz to a maximum $f_1 = 81$ GHz in $T = 180$ μs. We define the bandwidth of the chirp as $B = f_1 - f_o = 4$ GHz. The sinusoidal signals are transmitted every $T_{\text{rep}} = 250$ μs in sequences of $P = 256$ chirps each.

At the receiver, for each of the 16 antenna elements, a mixer combines the received signal (RX) with the transmitted one, generating the intermediate frequency (IF) signal, i.e., a sinusoid whose instantaneous frequency amounts to the difference between the frequencies of the TX and RX signals.

Then, the IF signal is sampled along three different dimensions. First, *fast time* sampling allows obtaining $N = 1024$ points from each chirp. For the *slow time* sampling, P samples, one per chirp from adjacent chirps, are taken with period T_{rep}. Finally, the *spatial* sampling is related to the M receiving channels, spaced apart by a distance d, and enables the localization of the targets in the physical space.

A discrete Fourier transform (DFT) is applied along each sampling dimension to extract the frequency components. In the resulting signal, referred to as range-Doppler-azimuth (RDA) map, the position of the peak along the fast time dimension reveals the *beat* frequency of the IF signal, f_b, and the peak along the slow time gives the Doppler frequency, f_d. The DFT along the spatial dimension reveals the phase shift due to the angular displacement of the target, f_a. Denoting the speed of light by c, the distance, velocity and angular position of the target by R, v and θ, respectively. Their estimates are [9]

$$\hat{R} = \frac{f_b T c}{2B}, \quad \hat{v} = \frac{f_d c}{2 f_o}, \quad \hat{\theta} = \sin^{-1}\left(\frac{f_a c}{2\pi d f_o}\right). \tag{1}$$

Using the parameters described above, our radar has a nominal range resolution of 3.75 cm.

III. PROBLEM OUTLINE

A. Notation

We consider a discrete time system, where time steps have a fixed duration of Δt seconds, corresponding to the radar frame period. Boldface lowercase letters refer to vectors, e.g., $\boldsymbol{x}$. Symbol $\odot$ denotes the elementwise product between vectors and $\boldsymbol{x}^T$ denotes the transpose of vector $\boldsymbol{x}$. The concatenation between two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ is denoted by $[\boldsymbol{x}, \boldsymbol{y}]$. A sequence of vectors in subsequent frames from time n to time m is indicated with the subscript $\boldsymbol{x}_{n:m}$. $||\boldsymbol{x}||$ is the Euclidean norm of vector $\boldsymbol{x}$.

B. Tracking

Indoor person tracking can be described as *sequentially* estimating the position of a person as she/he moves in the environment, tracing the trajectory of the subject and possibly predicting her/his future location [10]. We denote by $\boldsymbol{x}_k = [x_k, y_k]^T$ the state of a target at time step k, with x and y being respectively the coordinates along the reference system axes, centered in the radar device location. The information provided by the radar at step k is called an *observation* and is referred to as $\boldsymbol{z}_k = [R_k, \theta_k, v_k]^T$, where R_k, θ_k and v_k are respectively the distance, the angular position and the radial velocity of the target with respect to the device at time k. A key problem in the tracking procedure is the *filtering*, i.e., the estimation of the current (unknown) target state $\boldsymbol{x}_k$ given a sequence of present and past observations $\boldsymbol{z}_{1:k}$.

Non-linear Bayesian tracking methods, e.g., EKF and UKF, provide an approximate solution to this filtering problem using the assumptions that *(i)* the state at time k only depends on the state at time $k-1$ (first-order Markov assumption) and *(ii)* the observations are conditionally independent of one another, given the corresponding state value at a specific frame (conditional independency assumption). Bayesian tracking requires one to specify a transition and observation model, in the form of the probability distributions $p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1})$ (modeling transitions) and $p(\boldsymbol{z}_k|\boldsymbol{x}_k)$ (observations). The model is used in a recursive fashion, starting from a prior $p(\boldsymbol{x}_0)$, to form a predictive distribution on the state at the next time step as [10]

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) = \int p(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}) p(\boldsymbol{x}_{k-1}|\boldsymbol{z}_{1:k-1}) d\boldsymbol{x}_{k-1}, \tag{2}$$

and then compute the filtering distribution as

$$p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k}) = \frac{p(\boldsymbol{z}_k|\boldsymbol{x}_k) p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1})}{\int p(\boldsymbol{z}_k|\boldsymbol{x}_k) p(\boldsymbol{x}_k|\boldsymbol{z}_{1:k-1}) d\boldsymbol{x}_k}. \tag{3}$$

From Eq. (3), an estimate of the current state $\hat{\boldsymbol{x}}_k$ can be obtained using, for instance, the minimum mean-square error (MMSE) criterion, returning the expected value of the state under the filtering distribution.

In contrast with the above approach, in this article we use deep NNs capable of modeling long term and non-linear dependencies among the samples of a time sequence, approximating the posterior filtering distribution in Eq. (3) and the state estimates through a deterministic function $\hat{\boldsymbol{x}}_k = F_{\boldsymbol{\vartheta}}(\boldsymbol{z}_{1:k})$. This function is parameterized by the NN weights, $\boldsymbol{\vartheta}$, which are learned directly from the data (observations) by minimizing a loss function, and without requiring the definition of a system model. We stress that no assumptions, neither on the nature of the system (e.g., first order Markov process for the state evolution), nor on the noise distribution (e.g., Gaussian noise) are necessary. In particular, we use two types of NN architectures that have recently become the state-of-the-art in time series analysis and forecasting, i.e., the denoising autoencoders and the sequence-to-sequence denoising autoencoders with long short-term memory (LSTM) neural networks.

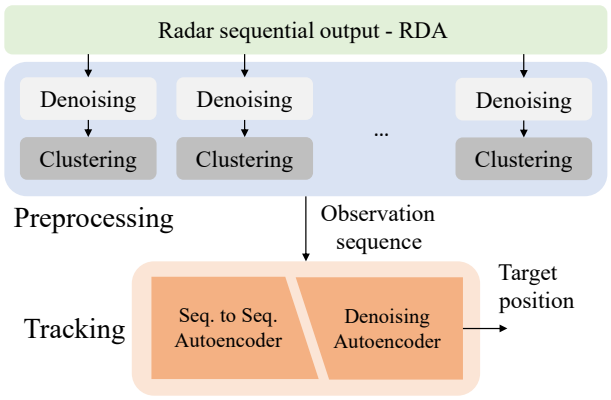


Fig. 1: Proposed workflow.

C. Neural networks for time series analysis

In the following, a brief introduction on LSTM networks and DAEs is provided.

1) Denoising autoencoder: The autoencoder (AE) is a model trained to encode the input into a lower dimensional or sparse representation (called the *code* or *latent* representation). Hence, the code is inputted into a decoder, from which the input is reconstructed with the lowest possible error. In order to learn more meaningful latent representations, *denoising* AEs are often employed, where the training input is corrupted with artificial noise before being fed to the DAE [7]. We implement the DAE using a cascade of two fully connected (FC) NNs: the first acts as an encoder by extracting relevant features (code) form the input, the second performs the decoding by reconstructing the original input starting from the code. The encoder and the decoder are *jointly trained* to minimize the reconstruction error. The encoder-decoder structure of the DAE is shared by the S2S model (see Section IV-B2). However, the latter is capable of representing the temporal features of the input more effectively thanks to the use of LSTM layers, instead of FC layers.

2) Long short-term memory: the LSTM [11] is a recurrent neural network (RNN) architecture suited to efficiently learn long term dependencies in time series. It has been successfully applied to different fields, such as natural language processing and speech recognition. The LSTM basic processing unit is the *memory cell*, which is characterized by two state vectors called the *cell* state $\boldsymbol{c}_k$ and the *hidden* state, $\boldsymbol{h}_k$. The cell state propagates through different time steps, carrying long term information about the process, while the hidden state controls the behavior of the cell in the current step and is renewed at every frame. $\boldsymbol{h}_k$ is also the output of the cell at time k.

At each time step, the input vector $\boldsymbol{z}_k$, corresponding to the current observation, is processed by the cell together with $\boldsymbol{h}_k$ through four FC layers, whose parameters are shared across the time steps. The LSTM network training is carried out using the back-propagation through time (BPTT) algorithm [12].

IV. Proposed approach

In Fig. 1, we show a high-level overview of the proposed signal processing workflow. The system operates on sequences of K RDA maps, obtained from consecutive radar frames. The maps are first preprocessed individually (and in parallel), without taking into account their temporal relation. A denoising procedure is carried out to reduce the effects of unwanted random disturbances and of reflections from static objects. Hence, a clustering algorithm is applied to identify the reflections of the target user(s) among the residual points after denoising. The centroids of the obtained clusters are observations of the real positions of the target(s) in the RDA space across time, and are denoted by $\boldsymbol{z}_k$. The observation sequence is then fed to the tracking block, where either a DAE or a S2S autoencoder is utilized to filter the observed time series, by jointly processing sequences of length K, $\boldsymbol{z}_{1:K}$, and outputting the estimated target positions at time $1, \dots, K$, namely, $\hat{\boldsymbol{x}}_{1:K}$. The process is iteratively repeated by applying a sliding window that selects the last K RDA maps that are to be processed each time a new map is acquired from the radar. During training, the whole sequence $\hat{\boldsymbol{x}}_{1:K}$ is compared with the target sequence $\boldsymbol{y}_{1:K}$ (see Section IV-B) and the error is backpropagated trough the network. When trained, the network provides a position estimate (i.e., $\hat{\boldsymbol{x}}_K$) for each new map acquired from the radar.

In the following sections we detail each processing block.

A. Pre-processing

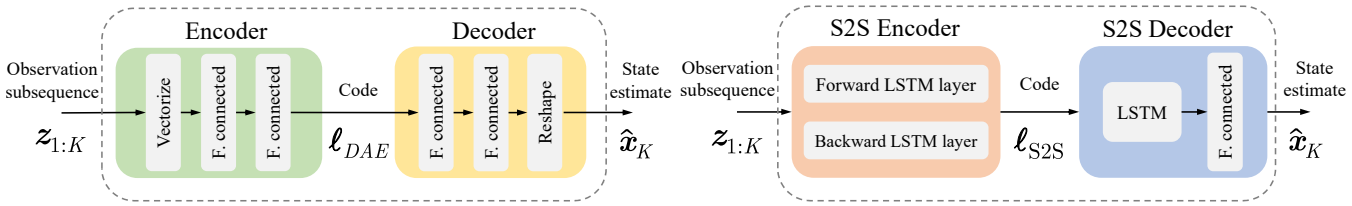
The pre-processing involves two different phases, namely the removal of static reflections/denoising and the clustering.

1) Denoising and removal of static reflections: The contribution to the reflected signal of the static objects in the environment is removed by deleting the samples with an estimated velocity close to 0 in the RDA maps. Specifically, we remove the values in the interval $[-0.135, 0.135]$ m/s.

For the noise removal, we apply two thresholds. The first one, along the range dimension, is decreased linearly in the logarithmic domain as the range increases, going from -70 dBm at minimum range to -95 dBm at maximum range. The second, along the angular dimension, is set at a level of -15 dB with respect to the peak value. Only the points with received power greater than both thresholds are maintained and represent candidate reflections from the targets.

2) Target clustering in the RDA space – DBSCAN: To identify the reflections from the targets, we group the residual points into clusters, using the *density-based spatial clustering for applications with noise* (DBSCAN) algorithm [13]. The algorithm requires one to specify two input parameters, ϵ and m_{pts}, respectively representing the range around each point and the minimum number of other points inside this range that must satisfy a given density condition. In this work, we use $\epsilon = 0.04$ and $m_{\text{pts}} = 40$.

We select the centroid of the cluster, $\boldsymbol{z}_k$, with the highest number of points as a noisy observation of the true coordinates (range, velocity and angle) of the person. This centroid is computed from a weighted average of the cluster points, where the weight assigned to each point is the corresponding normalized reflected power value. In this way, the centroid



(a) Denoising autoencoder (b) Sequence-to-sequence autoencoder

Fig. 2: Proposed DAE and S2S architectures.

moves towards those points with higher power, as they are more likely to represent the actual target position.

In the case of multi-target tracking, there are multiple retained clusters at each time step, which produce multiple observations. We use the Hungarian algorithm with Euclidean distance scores to assign observations to tracks [14]. As our current focus is on the tracking step rather than on the data association procedure, we refer to [15] for a detailed description of the multi-target association technique.

B. Target tracking – NN

Next, we present two different approaches to the tracking problem using the DAE and the S2S models. The first one is **supervised** (sDAE, sS2S) and consists of training the network by setting the target $\boldsymbol{y}_{1:K}$ as the true state sequence $\boldsymbol{x}_{1:K}$. This method requires one to know the ground truth values for the state sequence. To this end, we employ a motion-tracking system based on infra-red cameras that can measure the position of the targets in the physical space with an error in the order of the millimeter (see Section V). The values obtained from the motion tracking system are used as the ground truth $\boldsymbol{x}_{1:K}$ during training. The second approach is **unsupervised** (uDAE, uS2S), and entails a much more challenging learning task. In this case, the target values $\boldsymbol{y}_{1:K}$ are set to be the observations $\boldsymbol{z}_{1:K}$ and the network is trained to reconstruct the input introducing smoothness and regularity in the sequence, by adding a specifically designed term to the loss function (this will be detailed shortly in Eq. (4)). The reconstructed sequence is then mapped onto the estimated state sequence $\hat{\boldsymbol{x}}_{1:K}$ by transforming the polar coordinate representation into the Cartesian space. The encoder-decoder structure retains only the meaningful properties of the input, discarding variations due to noise. The unsupervised method is appealing as it does not require labeled data, which is usually difficult and costly to obtain.

Both the supervised and the unsupervised approaches rely on the same NN architectures, detailed in the following.

1) Denoising autoencoder: The proposed DAE is shown in Fig. 2a. The input sequence of observations, $\boldsymbol{z}_{1:K}$, is flattened onto a one-dimensional input vector and passed to the encoder. The encoder block has two FC layers with 20 and 10 units respectively and outputs a code, $\boldsymbol{\ell}_{\mathrm{DAE}}$, of dimension 10×1. The code is inputted into the decoder with a hidden layer of 20 units and an output layer of 30 units. All units use the hyperbolic tangent activation function. The output, one-dimensional vector, is reshaped into the output time sequence $\hat{\boldsymbol{x}}_{1:K}$ and compared against the sequence of target values $\boldsymbol{y}_{1:K}$, measuring the error in the reconstruction through the mean absolute error (MAE) loss, $L = \sum_{k=1}^{K} ||\hat{\boldsymbol{x}}_k - \boldsymbol{y}_k|| / K$.

2) S2S LSTM autoencoder: The S2S includes an encoder and a decoder, both based on LSTM layers, see Fig. 2b.

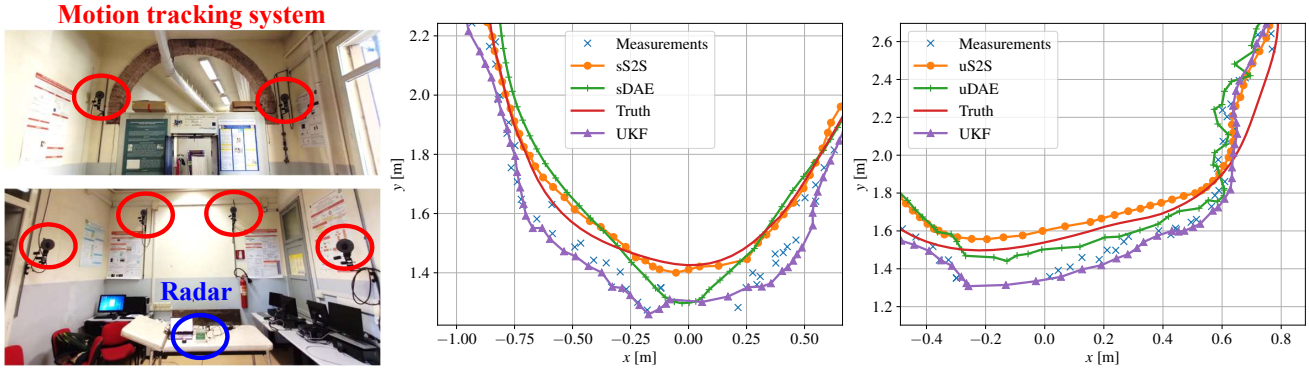
(i) The encoder takes as input the observation subsequence $\boldsymbol{z}_{1:K}$, and outputs a latent representation of the whole input sequence, $\boldsymbol{\ell}_{\mathrm{S2S}}$. To encode all the information available from the input sequence into the final state, we use a bidirectional LSTM layer that processes the observations $\boldsymbol{z}_{1:K}$ in the forward time direction, outputting the forward hidden and cell states $\overrightarrow{\boldsymbol{h}}_K$, $\overrightarrow{\boldsymbol{c}}_K$, and in the backward direction, outputting the backward hidden and cell states $\overleftarrow{\boldsymbol{h}}_1$, $\overleftarrow{\boldsymbol{c}}_1$. The FC layers in the encoder cells and the states $\overrightarrow{\boldsymbol{h}}_k$ and $\overrightarrow{\boldsymbol{c}}_k$, have dimension 8×1. The code is obtained concatenating the forward and backward states in the last processed time step $\boldsymbol{\ell}_{\mathrm{S2S}} = \left[\left[\overrightarrow{\boldsymbol{h}}_K, \overleftarrow{\boldsymbol{h}}_1 \right], \left[\overrightarrow{\boldsymbol{c}}_K, \overleftarrow{\boldsymbol{c}}_1 \right] \right]$.

(ii) The decoder is a unidirectional LSTM layer initialized with the cell and hidden states contained in $\boldsymbol{\ell}_{\mathrm{S2S}}$. The input at time step 1 is a zero vector with the same dimensionality of the system state, (i.e., 2×1). Each output $\boldsymbol{h}_k$ is processed with a FC layer with linear activation function to obtain an estimate of the target state $\hat{\boldsymbol{x}}_k$, which becomes the input for the cell at the next time step. The state estimate at the last time step, $\hat{\boldsymbol{x}}_K$, is the final output of the network and represents the value of interest for the filtering task.

The S2S model is trained by comparing the output vectors $\hat{\boldsymbol{x}}_k$ with the target values $\boldsymbol{y}_k$ in each time step $k = 1, \ldots, K$, and backpropagating the reconstruction error. The loss function has two components: the first component is a standard mean square error (MSE) term, that measures the quadratic reconstruction error between the target sequence $\boldsymbol{y}_{1:K}$ and the output subsequence $\hat{\boldsymbol{x}}_{1:K}$, while the second is a *smoothness* term, which evaluates the regularity in the predicted subsequence by computing the square norm of the first-order differences between subsequent time samples of $\hat{\boldsymbol{x}}_{1:K}$. Denoting the the first order differences by $\Delta_k = ||\hat{\boldsymbol{x}}_k - \hat{\boldsymbol{x}}_{k+1}||$, the expression of the loss function L is

$$L = \frac{1}{2K} \sum_{k=1}^{K} ||\hat{\boldsymbol{x}}_k - \boldsymbol{y}_k||^2 + \frac{\alpha}{K-1} \sum_{k=1}^{K-1} \Delta_k^2. \tag{4}$$

The tradeoff between the two components (MSE and smoothness) is regulated by the hyperparameter α. The use of the



(a) Measurement device setup. (b) Example trajectory, supervised case. (c) Example trajectory, unsupervised case.

Fig. 3: Measurement setup and example trajectories for supervised and unsupervised tracking techniques.

smoothness term is an original contribution of this work. Its effectiveness is shown in Section V.

V. Experimental results

A. Measurements setup

To evaluate the proposed tracking methods we conducted several measurement campaigns in two different rooms. The first room is an empty 20 m $\times$ 4.3 m corridor. A total of 20 minutes of RDA data were collected using the radar device from each of 4 different subjects walking freely in the environment. The unsupervised models were exclusively trained using the data from this setup. The second room is a 8 m $\times$ 4 m research laboratory equipped with a motion tracking system with 6 infra-red cameras, see Fig. 3a. Here, we collected the training data for the supervised models, along with some test sequences used to evaluate the tracking precision of all architectures: a total of 4 minutes training data and 2 minutes test data were collected, of which 1 minute with 2 targets walking simultaneously in the room. Subjects are allowed to move inside a 2 m $\times$ 4 m rectangle which is the working area of the motion tracking system. Ground truth data are collected by the motion tracking system, synchronized with the radar, through the detection of markers placed on the subjects by the six infra-red cameras. The measurements are acquired with a rate of 60 frames per seconds and include the location information of the subjects, in Cartesian coordinates, at any given time. To compute the orientation angle with respect to the motion tracking reference axes, and to correct the bias in the angular position values, markers were placed on the stationary radar device as well.

B. Evaluation metrics

The proposed tracking procedures are evaluated on the full test sequences of length I through the following metrics:

(i) root mean square error (RMSE), defined as $\text{RMSE} = \sqrt{\sum_{i=1}^{I} ||\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i||^2 / I}$, *(ii)* mean absolute error (MAE), defined as $\text{MAE} = \sum_{i=1}^{I} ||\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i|| / I$, *(iii)* localization error outage (LEO), corresponding to the probability that the error exceeds some threshold δ (expressed in meters), defined as $\text{LEO}(\delta) = P\left(||\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i|| > \delta\right)$.

C. Training and tracking results

The proposed DAE models were trained using the Adam optimizer with Nesterov momentum, while for the S2S models we used RMSprop [16]. Training was carried out until convergence of the loss function on a validation set, which required around 50 epochs for all models. In the S2S models, regularization was applied using *dropout* with retain probability of 0.8 on the layers that output the hidden state and the cell state. Data augmentation was applied to extend the training sets to four times their original size: new samples were obtained by adding Gaussian noise to the collected ones. The best results (see Section V-C) were obtained setting the windows size of the input sequences to $K = 15$ for both uDAE and sDAE, and to $K = 8$ and $K = 5$ for uS2S and sS2S, respectively, sliding the window one step from left to right at each new frame. For the S2S models we used $\alpha = 2$ for uS2S and $\alpha = 0.5$ for sS2S. In Fig. 4, we show the effect of varying parameter α on the resulting error on the test sequences, motivating our settings. The performance of uS2S is improved by tuning α and obtains better results with a higher value with respect to sS2S. Indeed, the enforcement of smoothness through the loss function is beneficial in this case, as no true data is available as a reference. sS2S works better with slight or no smoothness enforcement and is overall less affected by the parameter α.

The EKF and UKF methods are used as benchmarks and are implemented using a constant velocity model, which is customary in the person tracking application literature [1], [5], [17]. More advanced methods, such as switching KFs [18], have not yet been applied to the addressed problem, so we did not include them in the evaluation. Note that the numerical values for the process noise and the measurement covariance matrices have been empirically estimated on our dataset. We found that a proper estimation of these parameters is key towards obtaining good performance with EKF and UKF.

For a visual comparison between the approaches, in Fig. 3b and Fig. 3c, we show the same test trajectory estimated with

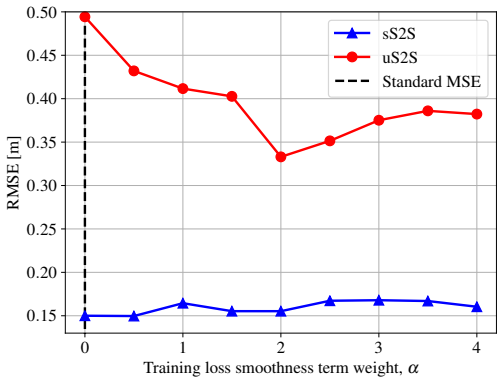


Fig. 4: RMSE obtained varying the α parameter in Eq. (4).

UKF and with our NN models. Our approaches outperform UKF, especially when the subject changes direction.

The numerical results of the different methods obtained in the single and multi-target cases (two subjects) are reported in Tab. 1. For a single target, our approach based on NN architectures outperforms Bayesian tracking. The gap is particularly evident in the supervised case (sDAE, sS2S) where it reaches an RMSE of 15 cm, a MAE of 12 cm and zero empirical outage probability in a range of $\delta = 75$ cm (for a single target). With unsupervised training, our method reaches an RMSE of 30 cm with the uS2S model. This demonstrates the high generalization capabilities of the proposed architectures, given that training is carried out on data taken from a different measurement room, and without using the reference ground truth positions.

In the multi-target case, the unsupervised methods perform similarly to the single target one. Supervised methods instead show lower precision, most likely due to having been trained with a small amount of data. This causes difficulties on very noisy measurements such as in the multi-target case, where two targets concurrently move within a small space. Having more labeled data would solve this issue.

VI. Conclusions

In this work, we have presented two neural network architectures for indoor person tracking from a mm-wave radar signal, namely a denoising autoencoder and a sequence to sequence autoencoder. The proposed processing pipeline features a preprocessing phase with a threshold-based denoising step, a density-based clustering step and the final tracking procedure via the neural network models. The system has been tested on real measurements collected in a realistic indoor space, including furniture, obstacles and other people. With the best algorithm, we obtained an average tracking error of 0.12 m with a single target and 0.21 m with two targets. Our approach can be effectively applied in a smart home scenario, e.g., to detect anomalies in elderly people movements, and it is non-intrusive, as people are not require to wear any device. Future research directions include the joint estimation of the state and the covariance matrix of the monitored process, providing a richer statistical description of the motion.

	Single target – multi target		
Method	RMSE [m]	MAE [m]	LEO(0.75) [%]
EKF	0.41 – 0.44	0.28 – 0.32	5.02 – 6.93
UKF	0.37 – 0.38	0.27 – 0.30	4.71 – 6.42
uDAE	0.35 – 0.36	0.25 – 0.25	5.40 – 5.92
uS2S	0.30 – 0.33	0.20 – 0.22	4.51 – 3.29
sDAE	0.22 – 0.29	0.19 – 0.26	0 – 0.3
sS2S	0.15 – 0.24	0.12 – 0.21	0 – 0.4

TABLE 1: Comparison between the proposed models, EKF and UKF methods. $X - Y$ in this table refers to the performance metric for the single target (X) and the multi-target (Y) cases (two subjects).

References

[1] P. Zhao, C. X. Lu, J. Wang, C. Chen, W. Wang, N. Trigoni, and A. Markham, "mID: Tracking and Identifying People with Millimeter Wave Radar," in *15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, (Santorini Island, Greece), May 2019.

[2] 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 Transactions on Geoscience and Remote Sensing*, vol. 56, no. 7, pp. 3941–3952, 2018.

[3] 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 Transactions on Aerospace and Electronic Systems*, vol. 54, pp. 1709–1723, Feb 2018.

[4] S. A. Shah and F. Fioranelli, "RF sensing technologies for assisted daily living in healthcare: A comprehensive review," *IEEE Aerospace and Electronic Systems Magazine*, vol. 34, no. 11, pp. 26–44, 2019.

[5] N. Knudde, B. Vandersmissen, K. Parashar, I. Couckuyt, A. Jalalvand, A. Bourdoux, W. De Neve, and T. Dhaene, "Indoor tracking of multiple persons with a 77 GHz MIMO FMCW radar," in *European Radar Conference (EURAD)*, (Nuremberg, Germany), Oct 2017.

[6] V. Winkler, "Range Doppler detection for automotive FMCW radars," in *European Radar Conference (EuRAD)*, (Munich, Germany), Oct 2007.

[7] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Journal of machine learning research*, vol. 11, pp. 3371–3408, Dec 2010.

[8] S. Amiriparian, M. Freitag, N. Cummins, and B. Schuller, "Sequence to sequence autoencoders for unsupervised representation learning from audio," in *Detection and Classification of Acoustic Scenes and Events Workshop (DCASE)*, (Munich, Germany), Nov 2017.

[9] S. M. Patole, M. Torlak, D. Wang, and M. Ali, "Automotive radars: A review of signal processing techniques," *IEEE Signal Processing Magazine*, vol. 34, pp. 22–35, Mar 2017.

[10] D. Dardari, P. Closas, and P. M. Djurić, "Indoor tracking: Theory, methods, and technologies," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 4, pp. 1263–1278, 2015.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Journal of Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[12] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[13] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *2nd International Conference on Knowledge Discovery and Data Mining*, (Portland, Oregon, USA), Aug 1996.

[14] Kuhn, Harold W, "The Hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[15] J. Pegoraro, F. Meneghello, and M. Rossi, "Multi-person continuous tracking and identification from mm-wave micro-doppler signatures," *arXiv*, no. 2003.03571, 2020.

[16] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[17] T. Wagner, R. Feger, and A. Stelzer, "Radar signal processing for jointly estimating tracks and micro-Doppler signatures," *IEEE Access*, vol. 5, pp. 1220–1238, Feb 2017.

[18] K. P. Murphy, "Switching Kalman Filters," *Technical report, DEC/CompaqCambridgeRe-searchLabs*, 1998.