University of Padova

Department of Department of Mathematics "Tullio Levi-Civita"

Master Thesis in Master Thesis in Data Science

Open-set person identification based on mm-Wave Radar Point-clouds using Siamese Neural Networks

Supervisor
Prof. Michele Rossi
University of Padova

Co-supervisor
Jacopo Pegoraro
University of Padova

Master Candidate
Khoa Bach Tran

Academic Year

2020-2021

Dedication.

Abstract

Millimeter-wave (mm-Wave) radar has been widely used in numerous applications in recent years, including drive-assistance system or short-range sensing due to its numerous advantages over other sensing technologies. The mm-Wave radar can measure the micro-Doppler phenomenon caused by moving objects in a scene, including people. The micro-Doppler effect induced by hunan gait has been proved to be a weak biometric identifier, due to the unique way of walking of each individual. In this work, we propose an open-set person identification based on the obtained mm-Wave radar point-clouds which intend to distinguish a new, unknown person from a known set of people. There are three main tasks studied: (1) extending a deep learning classification model to better distinguish unknown subjects in an open-set scenario; (2) applying Siamese Neural Network (SNN) for open-set identification to detect the new person in the recognized group of people; (3) evaluating the proposed method on our own measured data from a mm-Wave device on 20 subjects. We obtain useful experimental results to guide future work in this area.

Contents

Listing of figures

Listing of tables

Listing of acronyms

SNN .... Siamese Neural Network

NN .... Neural Network

(μD) .... micro-Doppler

mm-Wave ...... Milimeter-Wave

MIMO .... Multi-Input Multi-Output

DDRN .... Deep Discriminative Representation Network

CM .... Cosine Margin

CIP .... Class-Inclusion Probability

EVT .... Extreme Value Theory

RF .... Radio Frequency

CW .... Continuous Wave

PRI .... Pulse Repetition Interval

FMCW .... Frequency-Modulated Continuous Wave

AZ .... Azimuth

EL .... Elevation

TDM .... Time-Division Multiplexing

t-SNE .... T-distributed Stochastic Neighbor Embedding

KF .... Kalman Filtering

ED .... Euclidean Distance

CL .... Contrastive Loss

TL .... Triplet Loss

CS .... Cosine Similarity

PCA .... Principal Component Analysis

AoA .... Angle-of Arrival

RX .... Received Signal

IF .... intermediate Frequency

DFT .... Discrete Fourier Transform

RD .... range-Doppler

CA-CFAR ..... Cell-Averaging Constant False Alarm Rate

MTI .... Moving Target Indication

DBSCAN ...... Density-Based Spatial Clustering of Applications with Noise

CM-KF .... Converted-Measurements Kalman Filter

TC .... Temporal Convolution

PC .... Point Cloud

TCPCN .... Temporal Convolution Point-Cloud Network

MLP .... Multi-Layer Perceptron

CNN .... Convolutional Neural Network

1 INTRODUCTION

1.1 General Review

The application of milimeter-Wave (mm-Wave) radars to physical environment sensing is a rapidly expanding research field and featured in many works [6] [7] [8]. These radars can infer the position of surrounding barriers and individuals with high precision by producing an interrogation signal and measuring variations in the received reflected waves. These systems are an effective approach to monitor interior environments and infer important information about people's actions without gathering a visual image of the place, which may create privacy concerns. Because of its extreme sensitivity to frequency shifts caused by the Doppler effect, mm-Waves can be utilized to infer human movement patterns (gait) [9] [10], extracting characteristics from the subjects' micro-Doppler signatures (μD). Due to the significant randomness of mm-Wave propagation, deep learning approaches are frequently used to extract representative features from these (μD) signatures. However, most of the works are only dealing with single-target scenarios, i.e., with a single person walking in a indoor setup environment [11] [12] [13], and have recently developed to multi-target identification [14].

In this thesis, we try to solve the open-set human (gait) identification problem [15] by using point-clouds data instead of μD signatures. We focus on identifying one or several unknown persons from a group of known subjects. We use sparse point-clouds collected from the radar equipment that receives signal from the experimental targets, rather than receiving and processing raw data from the backscattered Milimeter-Wave (mm-Wave) signal. These point clouds, which are obtained at the radar processing unit using detection algorithms, contain information about the three-dimensional spatial coordinates of the reflecting points, their movement velocity, and the reflected power. We use a Neural Network (NN) that is inspired by PointNet [16], which allows significant features to be extracted from sparse point-cloud data. We then use different Loss functions and structures with the NN architecture mentioned above as the core feature extractor, to try to distinguish the unknown targets from a group of training known targets.

The thesis is organized in the following manner. An introduction to mm-Wave radar, Edge Computing device and Point cloud are offered in Chapter 2. From the Siamese Neural Network (SNN), Contrastive Loss, Triplet Loss to other techniques, the primary algorithms used to analyze the data are comprehensively examined in Chapter 3. The mathematical description and derivation of the SNN and the fundamentals of the loss functions used in the work take up a considerable portion of this chapter. Chapter 4 delves into the details of the suggested solution, taking you step by step through the many processing processes and design options. In Chapter 5, the algorithms are evaluated, with their performance and efficacy presented using various measures. The study's findings are presented in Chapter 6, along with a brief discussion of probable future advancements of this work.

1.2 State of the Art

Person identification using backscattered mm-Wave radar signals has sparked great and growing interest in recent years. There are many works [11] [12] [13] [17] [18] [19] [20] featuring micro-Doppler (μD) signatures that employ this kind of raw data received directly from the radars, to serve the classification problem using deep learning neural networks. This approach is both reliable and precise but it also has several flaws such as: firstly, it is a complex matter to extract the individual (μD) signatures from multiple concurrent targets. The majority of the solutions mentioned above are only applicable to a specific single subject. Secondly, in order to acquire highly accurate signatures, this approach requires large bandwidth to transfer the raw radar signal to the processing device. That will prevent the application from being implemented on low-cost devices.

Another approach is using point-clouds data generated by a low-cost mm-Wave Multi-Input Multi-Output (MIMO) radar system, instead of using μD signatures. Because the individual traits that identify each person are more difficult to extract and more vulnerable to external disruptions, the identification task is made more difficult by the sparsity of radar point-cloud data. In this work [1], the authors have created a system that can accurately monitor several subjects based on their point clouds data. They have used a quick and unique domain-specific deep learning classifier is used in conjunction with extended object tracking based on Kalman filtering. In order to improve the identification robustness and eliminate erroneous identity connections and trajectory swaps, they have integrated the tracking and identification modules closely. They have conducted the experiment with more than 2 subjects that share the same physical space, and performed the multiple subjects tracking. Their designed system is based on commercial edge-computing devices, and low-cost mm-wave radars that operate on real-time basis. Their proposed system can be summarized as the Figure 1.1. They have archived positive results with accuracy of 98.56% and 97.96% of 2 and 3 subjects respectively when they followed linear trajectories. And the accuracy scores were 98.56% and 97.88% when the subjects moved freely. However, they have conducted the research with only 3 subjects.

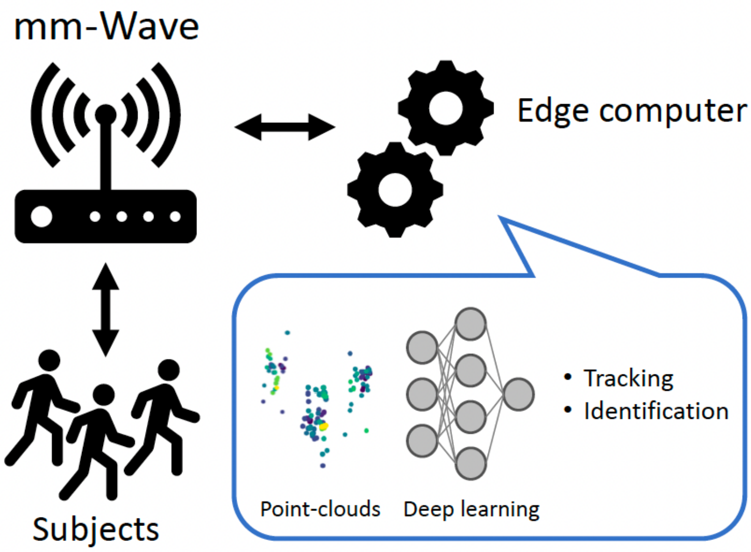

Figure 1.1: Proposed system from [1]

In another work that is based on gait radar micro-Doppler signatures [2], the authors took another approach. They extended the problem from just human identification to developing a recognition system that should be able to correctly recognize known identities as well as reject unfamiliar identities, i.e. the open-set identification [15]. They have used a probabilistic discriminant model based on a deep discriminative representation network (DDRN) [2] to develop the approach. They have trained the DDRN with the cosine margin (CM) loss to map gait samples into an embedding space where learnt features from the same identity are significantly closer together and those from other identities are much farther away. The class-inclusion probability (CIP) model associated with each known identity can then be constructed in the learned embedding space using statistical Extreme Value Theory (EVT) to bound each other's support region, which can then be used to estimate the probe sample's class-belongingness probabilities. Finally, a threshold is set to assess if these probabilities belong to one of the known identities or an unknown class. In their experiment, they have used a random division of 20 identities into 10 knowns and 10 unknowns. The result was promising as shown in the Figure 1.2.

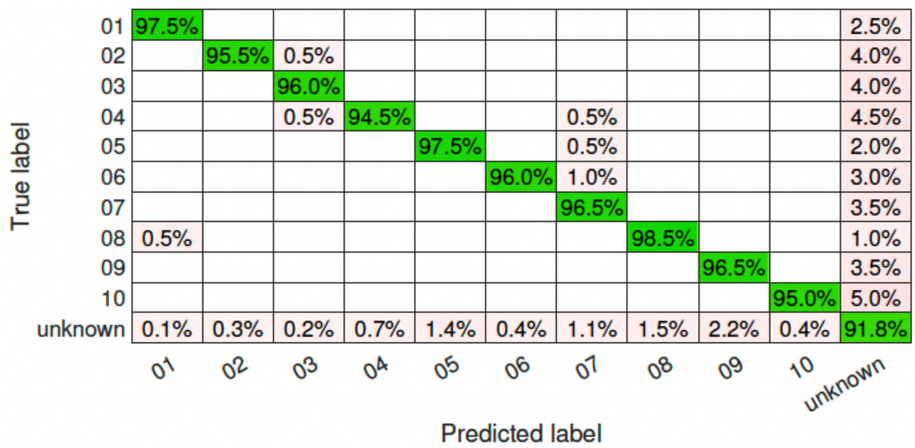

Figure 1.2: Confusion matrix on the test set with the maximum openness. Figure from [2]

In this work of thesis, we hope to combine the advantages of [1] to solve the open-set identification problem that mentioned in [2]. We will use more experimental subjects (up to 21 subjects), extracting their individual point-clouds data in a multiple-subjects setting and using unsupervised deep learning approach with the aim to detect the unknown subjects who have not appeared in the train set.

2

OVERVIEW OF THE TECHNOLOGY

This chapter is divided in two sections, to introduce briefly about the technology used to collect the data in this work.

2.1 Overview of the radar technology

2.1.1 Introduction to Radar

Radar is essentially a detection system that uses radio frequency (RF) waves to determine the range, angle, or velocity of objects in the environment. Instead of passively receiving the signal, radar is actually an active sensing system that emits the RF signals and then receives reflected signals from the objects. It sends out RF electromagnetic waves into the environment, usually in a certain direction, and then receives the reflected signal from objects in the radiation region [3]. There are many types of radar that operate upon different frequency bands of the modulated RF signals. Each frequency has different applications in real life. The major components of a generic radar system can be depicted in the Fig 2.1 below. In summary, a radar device includes:

- Transmitter: The waveform to be transmitted is generated here.
- Antenna: The waveform is propagated into the transmission medium.
- Switch: At any one time, the switch ensures that the antenna is only used by the transmitter or the receiver.

- Receiver: Depending on the type of radar device, the receiver amplifies the received signal and conducts mixing and demodulation.
- Signal Processor: The receiver's output is fed into a signal processor, which analyzes the data and performs target detection.

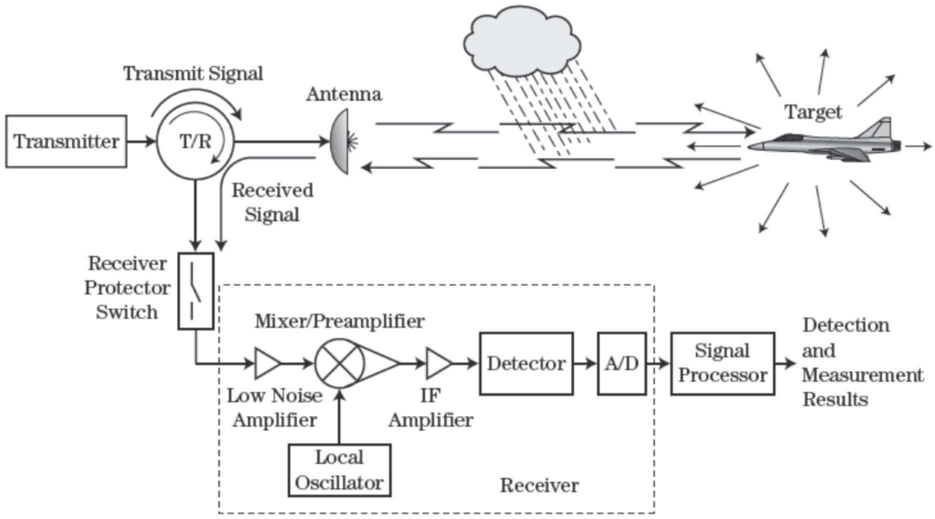

Figure 2.1: Radar components. Figure from [3]

2.1.2 Basics of Radar configuration and Radar Waveforms

Radar devices can be classified as many types based on different perspectives.

Radar Configuration

When considering antenna configuration, radar devices are divided into two types. It is noted that the distance between the transmitter and receiver antennas, not whether they are the same antenna or not, is what separates the two groups.

- Monostatic: the receiver and transmitter antennas are the same or very close to each other.
- Bistatic/Multi-static: The transmitter and receiver antennas are far apart enough in space to be considered independent devices. Multi-static radars acquire target reflections from numerous angles by relocating multiple receiver antennas in the environment.

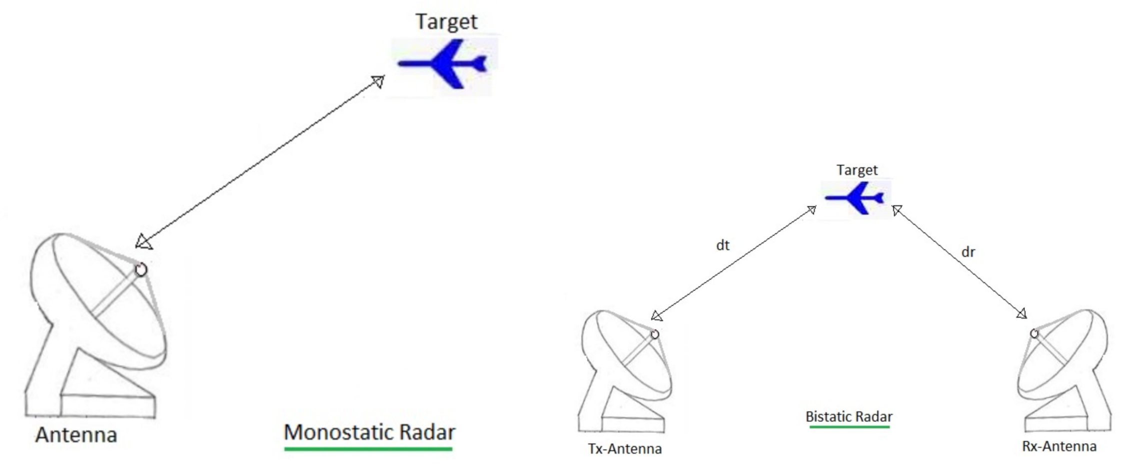

Figure 2.2: Monostatic and Bistatic Radar. Figure from [4]

Radar Waveforms

When considering the waveform type, we have the pulsed and the continuous wave radar.

- Continuous Wave (CW) radar: the device is constantly transmitting when it is in use. Because the transmitter and receiver must both function at the same time, this type of radar is extremely susceptible to transmitter/receiver interference. As a result, bistatic configurations are frequently chosen in CW radars. Its application are also usually confined to close-range.

- Pulsed radar: The waveform has a temporal limit of T, which is referred to as the pulse width. Pulse repetition interval (PRI) or T_{REP} is the time interval between two consecutive pulses. The receiver uses the idle time between the end of one transmission and the next to identify the reflected signal. The returning signal must be sampled with a sampling period of at least T to complete the detection after transmitting a pulse; otherwise, the returned waveform may be missed.

The radar used in this work is a Multi-Input Multi-Output (MIMO) frequency-modulated continuous wave (FMCW) milimetter-Wave radar.

2.1.3 Introduction to mm-Wave radar

The mm-Wave radar typically operates at 60-64 GHz or 76-81 GHz, with a wavelength of a few millimeters. Because of its operating RF frequency band and its high sensitivity to the frequency shifts caused by the Doppler effect, mm-Wave radar is primarily recommended for indoor applications and its use is rapidly expanding. Normally, the features extracted from the subject's micro-Doppler signature or μD, which contains time-frequency information about the induced Doppler shift, is more used. In this work, however, instead of using the raw data obtained from reflected mm-Wave signal, we exploit sparse point-clouds which is acquired from the radar device. We will summarize the fundamentals of the sensing technology used in this work below [1].

- The Multi-Input Multi-Output (MIMO) frequency-modulated continuous wave (FMCW) mm-Wave radars is used to collect data in this work. It operates by sending out chirp signal sequences, linearly sweeping a bandwidth $B = f_1 - f_o$, and examining their copies reflected back from the surroundings. f_o and f_1 are the frequencies of the transmitted chirp signal. In T seconds, the transmitted chirp signal's frequency is increased from a base value f_o to a maximum f_1. T is defined as the chirp duration. The transmitted signal is expressed as:
  $s(t) = \exp\left[j2\pi\left(f_o + \frac{\zeta}{2}t\right)t\right], \quad 0 \leq t \leq T$, where $\zeta = B/T$
- The chirps are transmitted in sequences of L chirps every T_{rep} seconds. One radar frame is equal a full chirp sequence and is repeated with a period Δt seconds. A mixer at the receiver combines the received signal and sent signals to produce the intermediate frequency signal, which is a sinusoid whose instantaneous frequency is the difference between the transmitted sequence and the received signals. Each chirp is sampled with period T_f (fast time sampling), yielding M points, while period T_{rep} (slow time sampling) yields L samples, one per chirp from nearby chirps.
- The usage of MIMO radar devices enables for the extra calculation of the angle-of-arrival (AoA) of the reflections by computing the phase shifts between the reception antenna elements due to their varied positions. This is known as spatial sampling, because it allows the targets to be located in the space. The radar equipment employed in this study has a virtual receiver array of $N_{\text{TX}}N_{\text{RX}}$ = 12 antennas, with N_{TX} = 3 transmitter and N_{RX} = 4 reception antennas. The transmitting elements are grouped in two spatial dimensions, azimuth (AZ) and elevation (EL), and are utilized to transmit chirp sequences using a time-division multiplexing (TDM) technique.
- The suggested system performs person tracking and identification in a sequential manner, estimating the position and identity of persons as they move freely in an interior setting. In this setting, a low-cost Texas Instruments IWR1843BOOST mm-wave, frequency-modulated continuous-wave, MIMO radar are used to collect the signals for the targets. Real-time processing function of the system is handled by a commercial edge-computing device (NVIDIA Jetson series), which is connected to the radar system.

2.2 Point Cloud

Point cloud is an important type of geometric data structure. By using a chain of detection, clustering, and tracking algorithms, a radar point cloud can provide information such as object location, velocity, and trajectory. The point cloud data generated in our experiment carries the information of movement velocity, the reflected power and the three-dimensional spatial coordinates of the reflecting points. These point-clouds are attained by using detection algorithms at the radar processing unit, so that the final data acquired is much smaller compared with the full raw data from the radar in order to to keep the computation complexity low.

3
ALGORITHMS

From a theoretical standpoint, this chapter discusses the algorithms and methods used in this thesis. Section 3.1 presents the basic knowledge of Siamese Neural Network, in section 3.2 the Contrastive Loss is described. Section 3.3 focuses on Triplet Loss. The supporting algorithm T-distributed Stochastic Neighbor Embedding (t-SNE) to represent the output features is discussed in section 3.4. The Kalman Filtering technique which is used for tracking the movement trajectory of each subject in space is introduced in section 3.6.

3.1 Siamese Neural Network

Deep models are crucial in making predictions for a wide range of applications. However, many machine learning techniques lack explainability, i.e., they are *black box* models. Siamese Neural Networks (SNN) are no different where there should be no suitable algorithms to explain it. In short, the SNN is made up of two or more identical neural subnets that share the same set of weights. Two or more inputs are encoded and the output features are compared when training a SNN which can be accomplished in a variety of ways. The similarity measures of the output features will determine the similarity or dissimilarity between a pair of feature vectors [21]. It employs a non-linear data embedding technique with the goal of grouping similar examples and separating dissimilar examples. SNN has many real world applications such as image recognition and verification, novelty and anomaly detection, one-shot or few-shot learning. Bromley and LeCun first used Siamese nets to solve signature verification as an image matching problem in the early 1990s. [21]

Let $\{(\mathbf{x}_i, y_i), i = 1, \ldots, n\}$ be a dataset consisting of *n* feature vectors $\mathbf{x}_i \in \mathbf{R}^m$ of size *m* with labels $\mathbf{y}_i \in \{\mathbf{1}, \mathbf{2}, ..., \mathbf{C}\}$. We construct a new training dataset:

$$S = \{(\mathbf{x}_i, \mathbf{x}_j, z_{ij}), (i, j) \in K\} \tag{3.1}$$

S consists of pairs of examples $\mathbf{x}_i$ and $\mathbf{x}_j$ with binary labels $\mathbf{z}_{ij} \in \{0, 1\}$. If both feature vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ are in the same class, then $\mathbf{z}_{ij}$ is assigned to be 1 and vice versa. Therefore, the training set S can be divided into two subsets: a positive set with label $\mathbf{z}_{ij}$ = 1 and a negative set with label $\mathbf{z}_{ij}$ = 0.

Suppose the outputs of the inputs $\mathbf{x}_i$ and $\mathbf{x}_j$ into the SNN are $\mathbf{h}_i \in \mathbf{R}^D$ and $\mathbf{h}_j \in \mathbf{R}^D$ respectively. The SNN generate a map function *f* such that $h_i = f(x_i)$ which tries to make the Euclidean Distance (ED) between the two outputs $d(h_i, h_j)$ as small (large) as possible. The smaller (large) the distance, the more similar (dissimilar) the two feature vectors are.

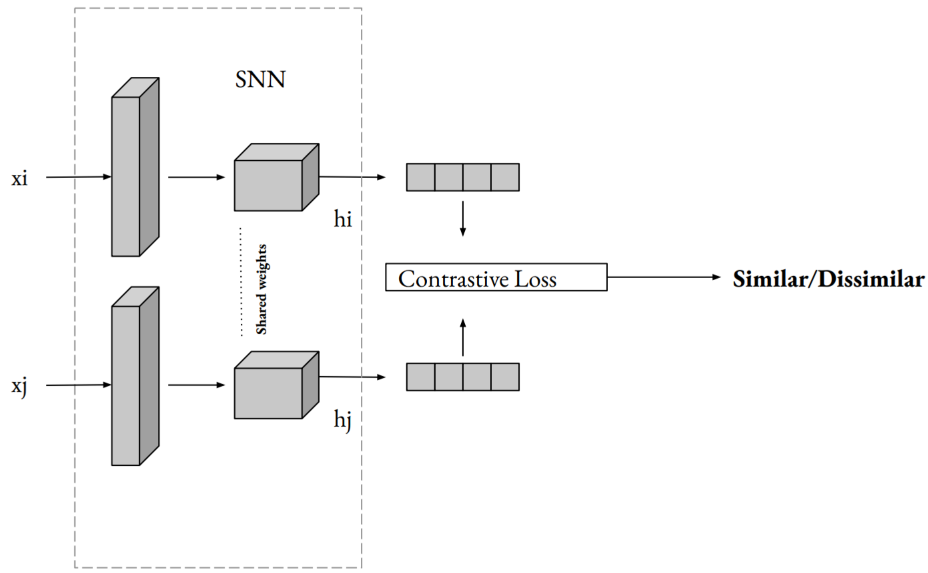

Figure 3.1: A basic architecture of the SNN

There are many loss functions to train the SNN. In this thesis, we will present Contrastive Loss (CL) and Triplet Loss (TL) in the following sections. There are many similarity measures or similarity functions that quantifies the similarity between two objects. We will discuss the two most-used ones which are Euclidean Distance and Cosine Similarity as below.

3.1.1 Euclidean Distance

Cosine similarity and Euclidean distance are both methods for determining the proximity of vectors in a vector space.

In $\mathbb{R}^n$, the ED between two vectors $x = (x_1, x_2, ..., x_n)$ and $y = (y_1, y_2, ..., y_n)$ is defined as $||x-y||_2$, which is equivalent to the L2-norm $||.||_2$ or the different $x-y$ between two vectors. It is computed as:

$$||x-y||_2 = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} = \sqrt{(x_1-y_1)^2 + (x_2-y_2)^2 + \ldots + (x_n-y_n)^2} \quad (3.2)$$

A Euclidean vector space such as $\mathbb{R}^n$ is a vector space in which Euclidean distances can be measured.

3.1.2 Cosine Similarity

Euclidean distances are represented by measuring distances with a ruler from a bird-eye view, whereas angular distances are represented by measuring differences in rotations. Cosine similarity is a measure of similarity that quantifies the cosine of the angle between two non-zero vectors in an inner product space. We can calculate Cosine similarity between two vectors by dividing their dot product by the product of their magnitudes. We suppose to have two vectors $x = (x_1, x_2, ..., x_n)$ and $y = (y_1, y_2, ..., y_n)$ as the above, then the Cosine similarity $\cos(\theta)$ of x and y is defined as:

$$\cos(\theta) = \frac{x \cdot y}{||x|| \cdot ||y||} \quad (3.3)$$

Cosine similarity is a judgment of orientation and not magnitude. The value of two vectors is within the range $[-1, 1]$. Two vectors with the same orientation have the cosine similarity value of 1. If they are at 90° relative to each other, the similarity value is 0. And the similarity of two diametrically opposed vectors is -1, regardless of their magnitude.

3.1.3 When to use Euclidean Distance or Cosine Similarity

The choice of the metric to employ is determined by the work at hand. Each work should be applied with suitable metric to have a better result.

- Several tasks such as preliminary data analysis benefit from both metrics. Because they allow for extraction of various insights into the data's structure. Other tasks such as text classification function better with Euclidean distance. Several other tasks such as retrieving the most comparable texts to a given document, work better with cosine similarity.

- In real life experiment, as is often the case in machine learning, we often understand all of the techniques and the heuristics that go along with them by learning through trial and error.

The way to speed up this process, though, is by holding in mind the visual images we presented here. If we do so, we'll have an intuitive understanding of the underlying phenomenon and simplify our efforts.

3.2 Contrastive Loss

Recently, a number of articles have exploited contrastive loss to demonstrate state-of-the-art outcomes with unsupervised learning. [22] [23] [24] follow the similiar patterns of using SNN and contrastive loss. When dealing with the datasets that equip with class labels for the inputs, a normal network and Categorical Cross-Entropy loss could be used. But the problem is we do not always have nicely labeled data which is usally the case, then the Contrastive loss can be used here. Contrastive loss compares the distance between a positive example and a similar example of the same class to the distance between negative instances. To put it another way, if positive samples are encoded to comparable representations and negative examples are encoded to distinct representations, the loss is minimal. The Contrastive is briefly explained as follow [25].

Firstly, the training set is divided to positive and negative pairs of training data points. An anchor sample x_a and a positive sample x_p, which is similar to x_a in the metric we want to learn, make up a positive pairing. Similarly, a negative pair is composed of the anchor sample x_a and a negative sample x_n, which is dissimilar to x_a. For positive pairs, the goal is to learn representations with a minimal distance d between them, and for negative pairs, a distance bigger than some margin value m. For positive pairs, contrastive loss forces a distance of 0; for negative pairs, it forces a distance greater than a certain margin. Let r_a, r_p and r_n are the outputs of the network of the corresponding input samples x_a, x_p and x_n and distance function d, we have:

$$L = \begin{cases} d\left(r_a, r_p\right) & \text{if} \quad \text{Positive Pair} \\ \max\left(0, m - d\left(r_a, r_n\right)\right) & \text{if} \quad \text{Negative Pair} \end{cases} \tag{3.4}$$

For positive pairings, the loss will be 0 only if the network creates representations for both items in the pair with no distance between them, and the loss will grow as the distance between them increases. Similarly, when the distance between the representations of the two pair elements is higher than the margin m, the loss for negative pairings is 0. The loss value will be equal to m as most when the distance of r_a and r_n is 0. The loss is positive when that distance is not bigger than m, the network will then have to update the weight to get more distant representation.

Let r_0 and r_1 are the outputs or the pair elements representations, y is the binary label that is equal to 1 for a positive pair and 0 for negative pair. We have the formula for the Constrastive loss:

$$L\left(r_0, r_1, y\right) = y\left\|r_0 - r_1\right\| + (1 - y)\max\left(0, m - \left\|r_0 - r_1\right\|\right) \tag{3.5}$$

3.3 Triplet Loss

The Triplet loss [26] utilizes a setup where triplets of training data samples are used, instead of pairs. The triplets are formed by an anchor sample x_a, a positive sample x_p and a negative sample x_n. The objective of Triplet loss is similar with Constrastive loss discussed above. The network will be optimized so that the distance between the anchor sample and the negative sample network ouputs $d\left(r_a, r_n\right)$ is bigger than an margin m and also greater than the distance between the distance between the anchor and positive representations $d\left(r_a, r_p\right)$.

With the same notions above, the formula of the Triplet loss can be written as [26] [27]:

$$L\left(r_a, r_p, r_n\right) = \max\left(0, m + d\left(r_a, r_p\right) - d\left(r_a, r_n\right)\right) \tag{3.6}$$

There are three situations of this Triplet loss:

- Easy Triplets: $d\left(r_a, r_p\right) > d\left(r_a, r_p\right) + m$. This situation orcurs when the negative sample is already sufficiently away from the anchor sample in comparison to the positive sample. The loss value is 0 and the network parameters are not updated.
- Hard Triplets: $d\left(r_a, r_p\right) < d\left(r_a, r_p\right)$. The negative sample is closer to the anchor sample than the positive sample. The loss value is positive and greater than the margin m

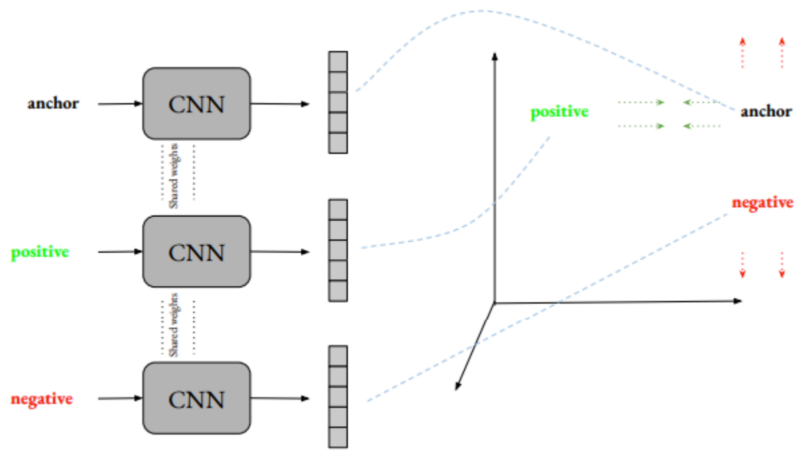

Figure 3.2: An example of Triplet loss

- Semi Hard Triplets: $d(r_a, r_p) < d(r_a, r_n) < d(r_a, r_p) + m$. The negative sample is more away from the anchor than the positive sample but the distance is not greater than the margin m. So that the loss value is positive and smaller than m.

The figure below illustrates three situations of Triplet loss.
When training on triplets, a model can use either online of offline triplet mining.

- Offline triplet mining: The triplets are manually generated then fit to the network.
- Online triplet mining: In this approach, the training set is divided into batched. The triplets are generated using all samples in the batch and the loss is calculated on it. In this case, the triplets can be randomized so that we can increase the chance to find the triplets with high losses which in turn will reduce the training time.

3.4 T-distributed Stochastic Neighbor Embedding (t-SNE)

Since we have a number of features in the outputs in the embedding, so in order to represent the feature in a 2-dimensional or 3-dimensional space, we use two common techniques to reduce the dimensionality of a dataset while maintaining the most important information in the dataset. These two techniques are T-distributed Stochastic Neighbor Embedding (t-SNE) [28] and Principal Component Analysis (PCA), which will be discussed in the following sections.

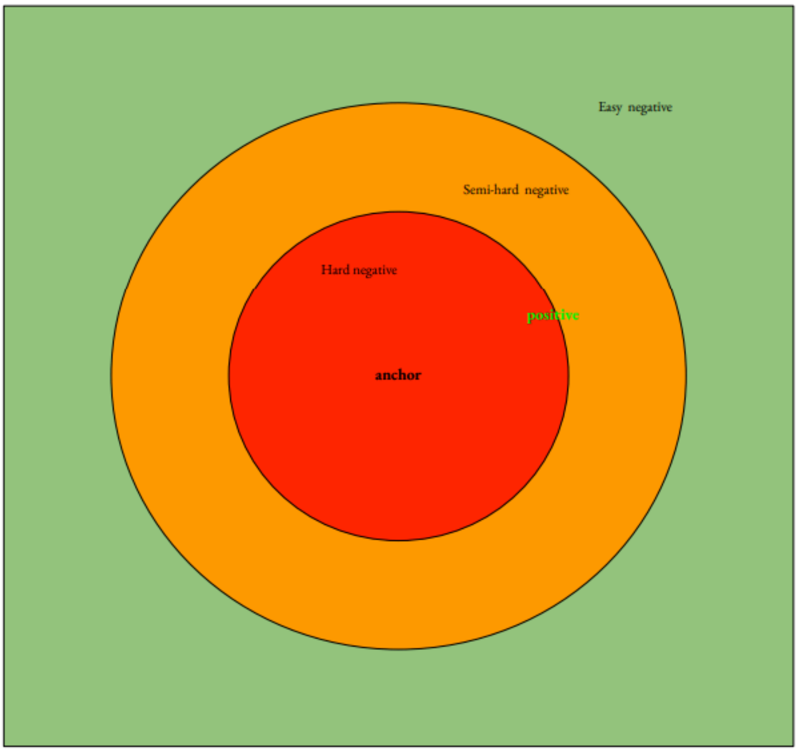

Figure 3.3: The demonstration of three situation of the Triplet loss

The objective of dimensionality reduction of data is to preserve as much of the high-dimensional data's significant information as possible in the low-dimensional version. It also help to easily visualize the features and enhance interpret-ability of the data in the lower dimension. One crucial thing of dimensionality reduction is minimizing information and structure loss as much as possible. t-SNE and PCA are two unsupervised dimensionality reduction techniques which are often used to visualize high-dimensional data in a lower-dimensional environment. The algorithm of t-SNE can be stated in three stages [29]:

- Creating a joint probability distribution that illustrates the data points' similarities.
- Making a dataset of points in the target dimension and computing their joint probability distribution.
- Gradient descent then is used to alter the dataset in low-dimensional space. The joint probability distribution representing it is as close to the one in high-dimensional space as possible.

For the **first stage**, we calculate the Euclidean distances of each point from all other points. After that, we convert these distances into conditional probabilities that explain the similarity between every point pair. Let x_i and x_j are two point. Then the conditional probability $p_{j|i}$ of x_j to be next to x_i can be written as:

$$p_{j|i} = \frac{\exp\left(-\left\|x_i - x_j\right\|^2 / 2\sigma_i^2\right)}{\sum_{k\neq i} \exp\left(-\left\|x_i - x_k\right\|^2 / 2\sigma_i^2\right)} \tag{3.7}$$

We have σ_i is the standard deviation of a Gaussian centered at x_i, which is the conditional probability $p_{j|i}$ of point x_i to have x_j as its neighbor. Because we may need to cope with clusters of differing densities, we divide by the total of all the other points placed at the Gaussian centered at x_i. In reality, the density of each cluster is often different from each other. As a result, if we solely use a Gaussian to calculate the similarities between each pair of points, the results may not reflect the real similarity.

As you can see the density of the orange cluster is lower than the density of the blue cluster. Therefore, if we calculate the similarities of each two points by a Gaussian only, we will see lower similarities between the orange points compared to the blue ones. In our final output we won't mind that some clusters had different densities, we will just want to see them as clusters, and therefore we do this normalization. At the end, we only want to perceive them as clusters, so we conduct this normalization.

In the **second stage**, we generate a collection of points and calculate a joint probability distribution for them in a low-dimensional space. A random dataset of points with the same amount of data points as the original dataset. The new dataset is built with K features, where K is the target dimension which often has value of 2 or 3 for the visualization. Then, in stead of Gaussian distribution, we create the joint probability distribution using the t-distribution. We have the following formula as the joint probability distribution with q is the probability and y is the points.

$$q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_{k \neq l} \left(1 + \|y_k - y_l\|^2\right)^{-1}} \tag{3.8}$$

The choice of the t-distribution over the Gaussian distribution is due to the t-heavy distribution's tails property. This feature causes moderate distances between locations in high-dimensional space to become extreme distances in low-dimensional space, preventing overlapping in the lower dimension.

For the **thrid stage**, we change the data to lower dimensional space in order to visualize it. We have the Kullback-Leiber [30] divergence between two distributions is the measure how different they are from each other. For distribution P and Q, the Kullback-Leiber divergence is defined as:

$$D_{\mathrm{KL}}(P\|Q) = \sum_{x \in \mathcal{X}} P(x) \log\left(\frac{P(x)}{Q(x)}\right) \tag{3.9}$$

The smaller the Kullback-Leiber divergence is, the more similar two distributions are to each other. When two distribution are identical, Kullback-Leiber divergence is equal zero. Thus, we have the Kullback-Leiber divergence as our target function with the cost function that the gradient descent attempts to minimize is:

$$C = KL(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} \tag{3.10}$$

KL is the Kullback-Leiber divergence of the distribution P and Q. P is the distribution from the higher dimension space and Q is the distribution from the low-dimensional space.

3.5 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a dimenionality reduction method aiming to reduce the diemnsionality of large datasets. It transforms a large collection of varialbles into a smaller one while retaining the majority of the information of the original dataset. Naturally, reducing the number of variables in a data set reduces accuracy; nevertheless, the answer to dimensionality reduction is to exchange some accuracy for simplicity. Smaller datasets are simpler for data visualization and exploration as well as more computing cost effective for machine learning and deep learning algorithms. The main purpose of PCA, in summary, is to reduce the number of features of the dataset, while retaining as much information as possible. The method of PCA can be followed by several steps.

- Step 1: The first step is standardization. The purpose is to normalize the range of continuous variables so that they all contribute equally to the analysis. If the ranges of initial variables differ significantly, the variables with wider ranges will prevail than those with smaller ranges resulting in biased results. Each value of each variable can be calculated by removing the mean and dividing by the standard deviation.

$$z = \frac{\text{value } - \text{ mean}}{\text{standard deviation}} \tag{3.11}$$

- Step 2: This step is to understand how correlated each variable with each other. If the variables are highly connected, they might carry duplicated information. We compute covariance matrix in order to find these connections. The covariance matrix is a $p \times p$ symmetric matrix containing the covariances associated with all possible pairs of the initial variables as entries. p is the number of dimensions. The covariance matrix for a 3-dimensional data set with three variables x, y, and z is a 3×3 matrix as the following example.

$$\begin{bmatrix} \mathrm{Cov}(x,x) & \mathrm{Cov}(x,y) & \mathrm{Cov}(x,z) \\ \mathrm{Cov}(y,x) & \mathrm{Cov}(y,y) & \mathrm{Cov}(y,z) \\ \mathrm{Cov}(z,x) & \mathrm{Cov}(z,y) & \mathrm{Cov}(z,z) \end{bmatrix} \tag{3.12}$$

  The covariance matrix entries are symmetric with respect to the main diagonal, meaning the upper and lower triangular parts are equal.

- Step 3: In this step, we compute the eigenvectors and eigenvalues of the covariance matrix calculated above in order to identify the principal components. Principal components are fresh variables which are created by combining or mixing the initial variables in a linear way. The new created variables are uncorrelated. The majority of the information of the initial variables is compressed into the first components. Thus, the n-dimensional data will provide you with corresponding n primary components. But the maximum possible information will be put into the first component and so on. By discarding the components with low information and considering the remaining components as your new variables, we can reduce dimensionality without losing much information. However, the principal components are more difficult to interpret because they are just the linear combinations of the initial variables and have no real meaning. The first principal component contains the largest possible variance in the data set. The second principal component will take the maximum remaining information and so on. The number of principal components is equal to the number of variables of the data. In order to attain principal components, we actually have to obtain eigenvectors and its eigenvalues. The covariance matrix's eigenvectors are the axes' directions with the greatest variance with the most information. We call them principal components. Eigenvalues are the coefficient attached to eigenvectors which give the amount of information in each principal component. Therefore, by ranking eigenvectors by their eigenvalues, we get the order of principal component from the most to the less significance.

- Step 4: In this step, we optimize the result for the last step. We decide whether to keep all principal component of discard those with insignificant eigenvalues. The remaining ones are formed as a matrix of vectors that called Feature Vector. To put it simple, the feature vector is a matrix with the eigenvectors of the components that we decide to keep as columns. Thus, from **n** dimensions as the original, we decide to keep only **p** eigenvectors or **p** dimensions. At the end, it is up to us to decide whether we want to keep all the dimensions or dispose of the ones with lesser information in order to reduce dimension.

- Step 5: In this step, we use the feature vector, which is constructed by the eigenvectors and eigenvalues above, to readjust the original axes of the data to the ones defined by the principal components. The following equation simply explain how we do it. The transpose of the feature vector multiplies with the transpose of original dataset to form the final one.

$$\text{FinalDataSet} = \text{FeatureVector}^{T} * \text{StandardizedOriginalDataSet}^{T} \tag{3.13}$$

Variable	Definition	Structure	Category
$\mathbf{x}$	state variable	$n \times 1$ column vector	Output
P	state covariance matrix	$n \times n$ matrix	Output
z	measurement	$m \times 1$ column vector	Input
A	state transition matrix	$n \times n$ matrix	System Model
H	state-to-measurement matrix	$m \times n$ matrix	System Model
R	measurement covariance matrix	$m \times m$ matrix	Input
Q	process noise covariance matrix	$n \times n$ matrix	System Model

Table 3.1: Kalman Filter algorithm reference terms

3.6 Kalman Filtering (KF) technique

The signal received by the radar is processed in order to detect the object. The points generated by the subjects are separated from clutter and noise using density-based clustering method. After that, the Kalman Filtering (KF) techniques are applied to each cluster centroid in order to follow the the movement trajectory of each subject in space. We will briefly introduce the fundamentals of KF in this section.

Kalman filtering is an algorithm that estimates a joint probability distribution over the variables for each timeframe using a series of measurements observed over time, which has long been considered the best solution for many tracking and data prediction tasks. It has been widely reported in the investigation of visual motion. Filtering is used to retrieve only the information needed from a signal while discarding everything else. A cost or loss function can be used to determine how well a filter fulfills this duty. In fact, we can describe the filter's purpose as the minimization of this loss function. The filter is built as a mean squared error minimiser, but an alternative derivation that shows how the filter links to maximum likelihood statistics is also presented. It is also a recursive filter that uses a sequence of noisy observations to estimate the internal state of a linear dynamic system. The Kalman Filter's true strength is not in smoothing measurements. It is the ability to estimate system parameters that cannot be accurately measured or observed. The algorithm of Kalman filter can be summarized as the following diagram [5]

The variables utilized in the algorithm are listed in the table 3.1. There is a structural type and category for each variable listed.

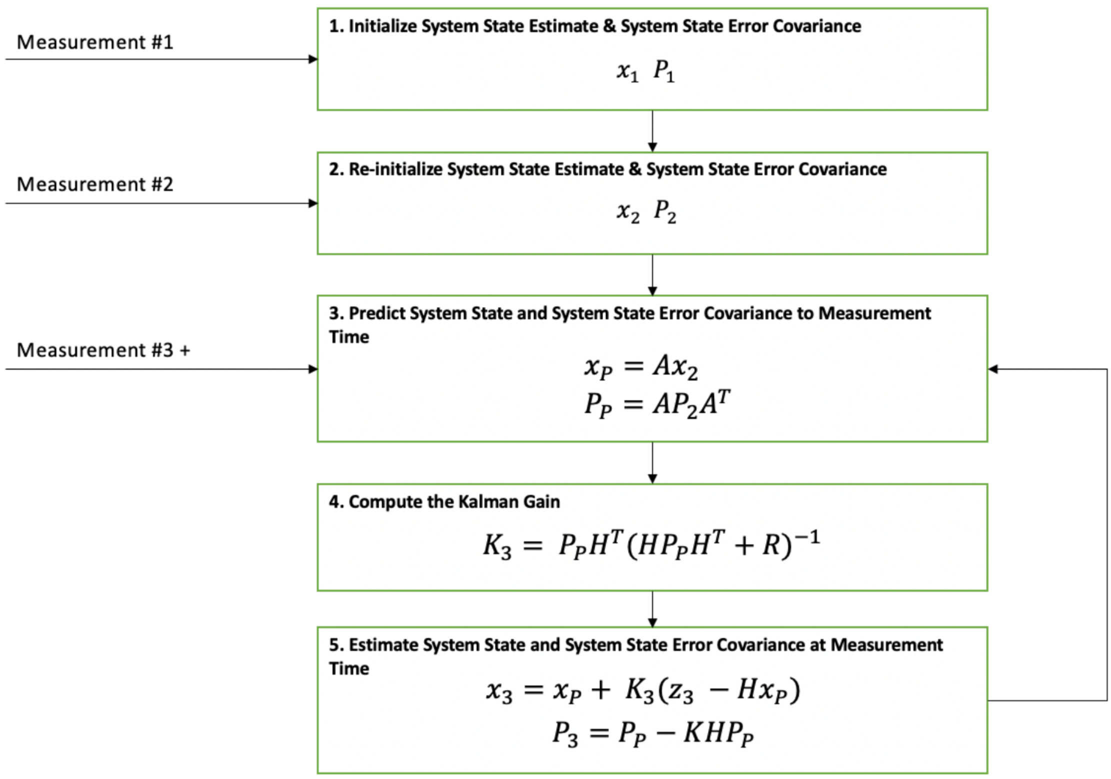

Figure 3.4: Kalman Filter algorithm diagram. Figure from [5]

4

PROPOSED SOLUTION

This chapter briefly introduce the radar signal and data processing as well as describes the proposed classification process using SNN.

4.1 Radar Signal and Data Processing

The radar signal and data processing method used in this thesis is similar to what has been used in this paper [1]. We will introduce concisely step by step as follows.

4.1.1 mm-Wave Radar Signal Processing

The frequency-modulated continuous wave (FMCW) estimates the distance and the radial velocity of the target with respect to the device by emitting sinusoidal waves with frequency that is linearly increased over time. Then the frequency shift of the reflected signal is measured at the receiver. By calculating the phase shifts caused by the different positions of the receiver antenna elements, the multiple-input multiple-output (MIMO) radar devices used in the experiment also measure the angle-of arrival (AoA) of the reflections. This is known as spatial sampling that helps to localizing the targets in the physical space.

- The first step of radar signal processing is to extract the range and Doppler information. A mixer at the receiver combines the received signal (RX) with the transmitted signal to produce the intermediate frequency (IF) signal. The IF signal can be expressed as a function with respect to the sampling indices along the fast and slow time. Samples of the IF signal can be organized into a matrix that contains all of the information provided by a single antenna over a given time period. Applying a bi-dimensional discrete Fourier transform (DFT) along the fast and slow time dimensions, followed by taking the square magnitude of each obtained complex value, yields the range and velocity of each reflector, resulting the radar range-Doppler map (RD). The main reflecting points are identified using the RD maps and the cell-averaging constant false alarm rate (CA-CFAR) algorithm. A processing step is also required to remove the reflections from static objects, which is carried out by using a moving target indication (MTI) high pass filter. This filter will remove reflections with Doppler frequency values near zero. The final result of the process is a sparse RD map with a determined number of detected reflecting points. Several desired quantities can be identified for each detected point.

- The second step is to estimate Azimuth and Elevation angles. The spatial diversity of the receiver array which presents a different phase shift can be used to estimate the azimuth and elevation angles of targets. The final output is a vector that contains the information of each single detected reflecting point. Each vector has five components including the Cartesian coordinates, the velocity and the reflected power of the detected reflecting point.

4.1.2 Data Processing

Normally, the common approach to people movement tracking from radar point-clouds includes 2 parts.

- Detection: Separating the points generated by the subjects from clutter and noise using density-based clustering method.
- Tracking: To follow the movement trajectory of each subject in space, Kalman Filtering (KF) techniques are applied to each cluster centroid.

Because the experiment is performed in a room that has many other static objects, the received signals contains not only the target but also other matters. Thus, we have to group the points detected by CA-CFAR into several clusters, a density-based clustering algorithm is used, DBSCAN in this case. Each cluster will represent a different subject in the environment. The density-based clustering algorithm has been critical in determining nonlinear shape structures based on density. The input samples are grouped by their local density. By doing that, we can classify which cluster is the target. We use Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [31], which is most widely used density based algorithm and has been used in many works. DBSCAN is used in this case because it is a unsupervised learning algorithm that we do not need to know the number of clusters in advance. The method also has a low processing complexity due to its noise rejection quality, which, along with its density-based clustering technique, allows for reliable and automatic separation of reflections from different individuals. It is beneficial in real life situation where the received signals of the subjects may be spoiled by environmental noise, signal blockage or imperfect clutter cancellation. The algorithm is also low in computational complexity which will benefit limited hardware resource. DBSCAN sequentially samples all data points and grows the cluster until the specified density connection criteria are no longer met. DBSCAN takes two input parameters ε and m_{pts}, which are a radius surrounding each point and the minimal number of additional points that must be inside that radius to meet a density condition respectively. DBSCAN takes the coordinates of points in the horizontal plane $(x - y)$ as input and produces a list of discovered clusters and a set of points classed as noise. The subject's position is determined by the centroid of each cluster, which is then sent into a KF tracking algorithm.

In this tracking phase, we employ a converted-measurements Kalman filter (CM-KF). This filter will not only estimates the position of the targets in Cartesian coordinates, but also estimates the subject's extension in the horizontal plane $(x - y)$. It allows the object to be considered as an extended object rather than an ideal point-shaped reflector. The KF is briefly introduced in the section 3.6 above.

4.1.3 Architecture

This section introduces the neural network architecture used to extracting the features (temporal convolution point-cloud network). This architecture was well-presented in this paper [1]. We will briefly introduce it in this section.

The reason we use this neural network architecture in this work is because this network can extract useful features from a temporal sequence of point clouds generated by the detection and tracking steps. The network is included with two two different blocks, termed point-cloud block (PC) and temporal convolution block (TC). To extract a feature vector, a PC block is applied to each every time step. To understand the temporal patterns in the series of feature vectors, causal dilated convolutions are used.

- Point-cloud Block: combined with a number of K of identical feature extraction blocks which are applied to the standardized input point-clouds, $Z_i, i = 1, \ldots, K$. Each of

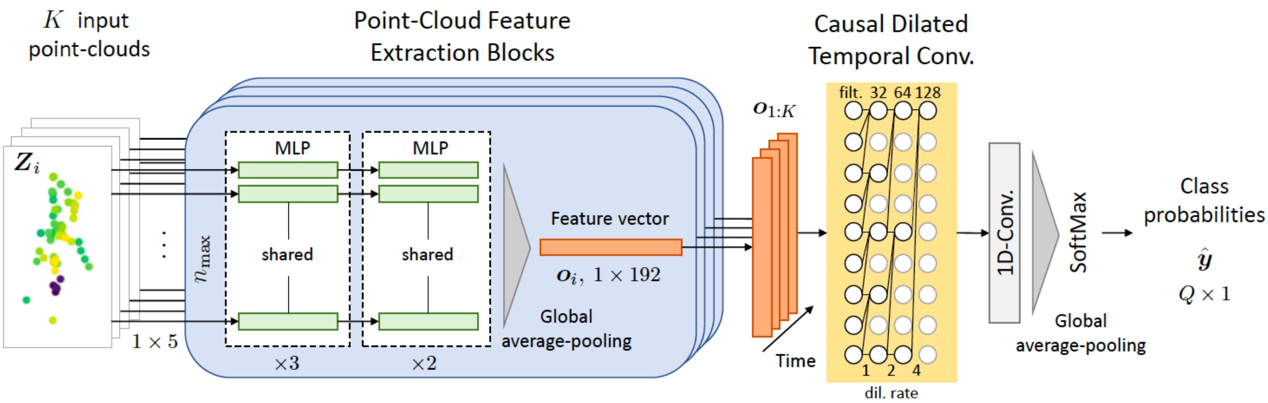

Figure 4.1: Architecture of the temporal convolution point-cloud network (TCPCN) for subject identification. Figure from [1]

the block is the cascade of a multi-layer perceptron (MLP) [32] followed by a global average pooling operation denoted as the function $f_{\boldsymbol{W}}(\cdot)$, where W is a set of weights to be learned. The structure of the function $f_{\boldsymbol{W}}(\cdot)$ is inspired by the popular PointNet [16]. Both PointNet and the PC block use only functions that are invariant to the ordering of the input points by sharing the MLP weights and utilizing appropriate pooling algorithms. Because point clouds that differ only in how the points are ordered would provide the same output, this ensures resilience and generality.

- Temporal Convolution (TC) Block: The sequence of feature vectors $\boldsymbol{O}_{1:K} = \{f_{\boldsymbol{W}}(\boldsymbol{Z}_i)\}_{i=1:K}$, each of dimension 192 is input to the TC block, applying a function $h_{\boldsymbol{U}}(\cdot)$ along the temporal dimension, where U is another set of weights (like W). $h_{\boldsymbol{U}}(\cdot)$ has temporal convolutions, which are a form of convolutional neural network (CNN) layer [32] [33] in which the input is convolved with a uni-dimensional filter with learnt weights to recognize temporal patterns. We use three temporal convolution layers in the proposed TCPC network, each with filters of dimension 3 with dilation rates of 1, 2, and 4, respectively. The applied filters are repeated along the input's feature vector components, yielding 32, 64, and 128 feature maps at each layer.

- The TCPCN's last layer is a temporal convolution layer, which converts the extracted temporal characteristics into Q feature maps, each of which corresponds to one of the output classes. To integrate the information from each feature map into a single vector of dimension Q, it does a normal convolution with a kernel size of 3 and then a global average pooling.

4.2 Prepare Data

Before being transmitted to the NN classifier, the point cloud sequence $Z^t_{k-K+1:k}$, where t is the track, k is the time step and K is the length, produced from each CM-KF track is preprocessed. By subtracting their mean value and dividing by their empirical standard deviation, the points' characteristics are standardized. Furthermore, because the TCPCN is a feed forward neural network that processes fixed size input vectors, the point clouds must have a predetermined number of points before being delivered to it. The maximum number of points for a single time step is picked as $n_{\max} = 100$. The value of $n_{\max}$ was determined by evaluating the distribution of the number of detected points for various human subjects and determining an appropriate value empirically. $n_{\max} = 100$ is sufficient to cover all users in practically every frame in the experiment.

4.3 Train model with SNN and try with several losses

4.3.1 Contrastive Loss

A siamese network's goal is to distinguish between image pairs rather than classify them. Contrastive loss is essentially a measure of how well the siamese network distinguishes between pairs of input samples. The distinction is subtle but significant. We will create pairs of samples and label the pairs to feed into two sisters network. As mentioned in section 3.1, the formula of the Constrastive Loss is:

$$L\left(r_0, r_1, y\right) = y\left\|r_0 - r_1\right\| + (1-y)\max\left(0, m - \left\|r_0 - r_1\right\|\right) \tag{4.1}$$

Or we can rewrite it to a simpler form.

$$L = Y^* D^2 + (1-Y)^* \max(\text{margin} - D, 0)^2 \tag{4.2}$$

- Y is the label. Y is equal 1 if the two samples in the pair in the same class. Y is equal 0 if they are in different classes.
- D represents the Euclidean distance between the sister network embeddings' outputs.
- The max function will take the max value of 0 and the margin minus the distance.

The dataset after loading is then precessed by:

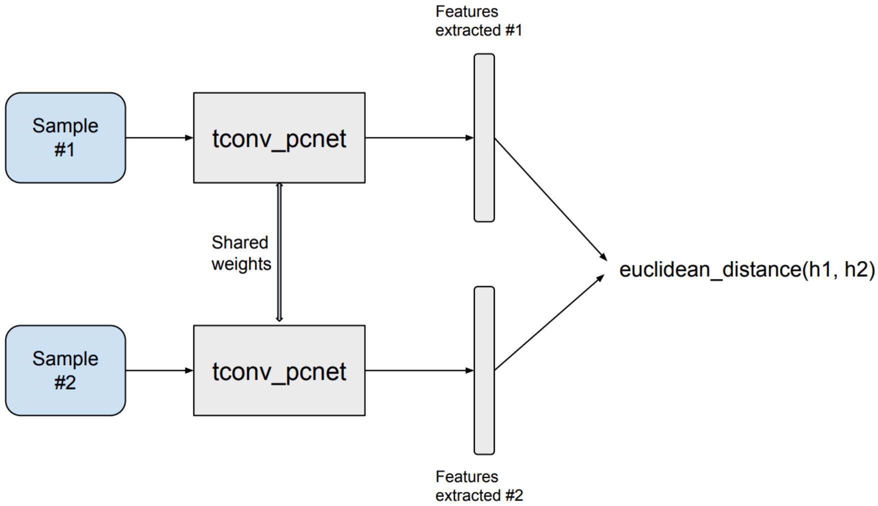

Figure 4.2: Siamese networks with Contrastive loss

- Data augmentation by shuffle the point cloud sequence as well as randomly remove timestep, row or column of the point cloud sequence.
- Constructing the sample pairs. For each point cloud sequence sample, we randomly select one sample in the same class and one sample in a different to create 2 pairs, one positive and one negative.
- To constructing the sister network architecture, we refer the *temporal convolution point-cloud network* (TCPCN) in this paper [1]. The network will act as the feature extractors.
- Each point cloud sequence in the pair will be processed by the feature extractor, yielding a vector that quantifies each sample.
- We then compute the euclidean distance between our two vectors using the vectors generated by the sister networks. This distance serves as the output from the Siamese network. The smaller the distance is, the higher chance two samples belong to the same subject. The larger the distance is, the lower chance two samples in the same subject.

4.3.2 Triplet Loss

The algorithm of the Triplet loss can be visualized in the Figure 4.2. Basically, it minimizes the distance between the anchor and a positive sample, in which both of them belong to the same class. It also maximizes the distance between the anchor and a negative sample from a different class. For each training sample. The feature representations of inputs belonging to various classes are separated using the triplet loss. This is accomplished by selecting triplets of input samples from the training set, two of which are from the same class, resulting in feature vectors $\boldsymbol{v}_a$ and $\boldsymbol{v}_p$, and one from a different class, resulting in feature vector $\boldsymbol{v}_n$. The triplet loss function can be written by:

$\mathcal{L}_{\text{tri}}\left(\boldsymbol{v}_a, \boldsymbol{v}_p, \boldsymbol{v}_n\right) = \max\left\{\|\boldsymbol{v}_a - \boldsymbol{v}_p\|_2^2 - \|\boldsymbol{v}_a - \boldsymbol{v}_n\|_2^2 + \mu, 0\right\}$

μ is a margin hyperparameter, set to 1.

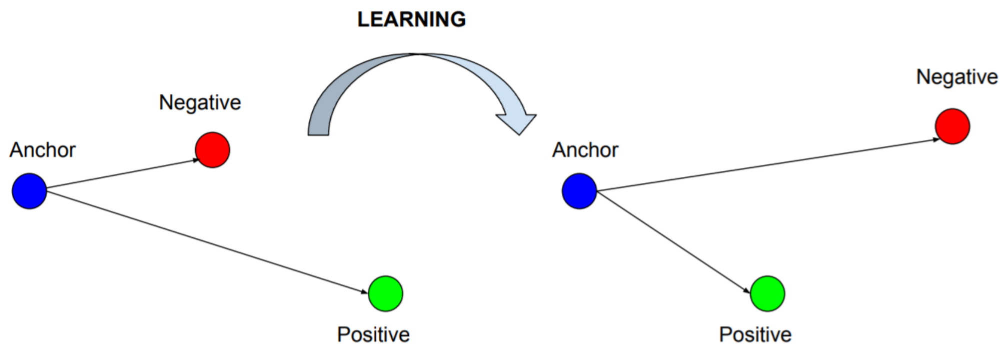

Figure 4.3: Triplet Loss visualization

4.3.3 Contrastive Loss combined with Triplet Loss

In this algorithm, we combine between the Contrastive Loss and Triplet Loss into the structure. For each branch, we use the same TCPCN as the section 4.3.1. The Triplet Loss is applied for each branch as the loss function in order to extract the features. The output of each branch is a vector contained the learned features. The euclidean distance is used to compute the distance between our two vectors.

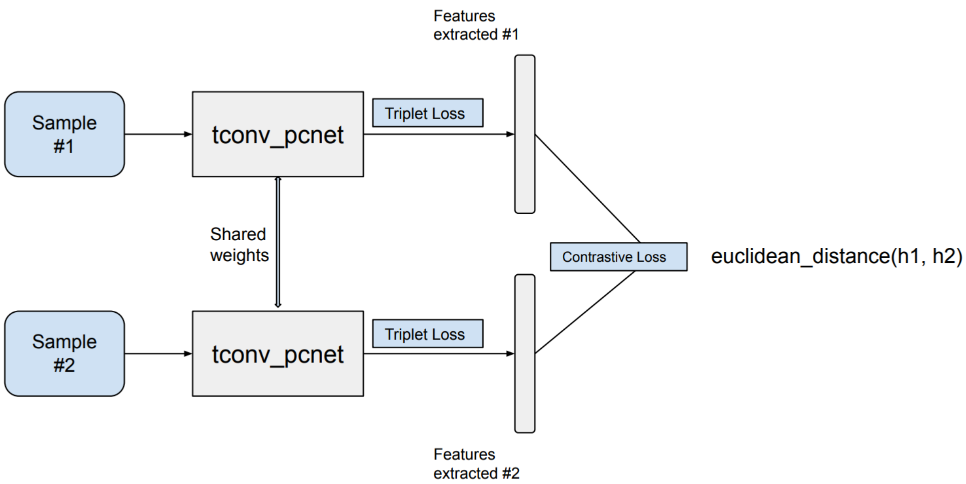

Figure 4.4: Combine Triplet Loss and Contrastive Loss

5
EXPERIMENTAL RESULTS

This chapter provides a detailed summary of the results obtained with the proposed method. The

5.1 Dataset Description

The dataset used in the experiment includes total of 21 targets noted from 1 to 21. We use the subject from 1 to 16 for the train and validation dataset. The left from 17 to 21 is the subjects who have not appeared in the training set, which is used for the test set. A training dataset of mmWave radar point-clouds from 16 targets was employed in the implementation. It was collected in a variety of interior environments in order to improve the NN's generalization capabilities. The Point-cloud data are organized as lists of python dictionaries, each of which corresponds to 1 timestep. Each dictionary contains a point cloud with N points, the following keys are present:

- *elements*: array of dimension $(N, 2)$, contains (x, y) coordinates in m
- z_coord: array of dimension N, contains the z coordinates in m
- *dopplers*: array of dimension N, contains the velocities in m/s
- *powers*: array of dimension N, contains the received powers in dB

5.2 Training

When training, state-of-the-art siamese networks typically use either contrastive loss or triplet loss — these loss functions are better suited for siamese networks and tend to improve accuracy. To train the model, we use different loss functions and combination. The based network is the one that has been mentioned above, which is used to extract features from the point cloud.

5.2.1 Contrastive Loss

To see if the features extracted from different targets can be clustered well by the model, we use t-SNE, which is discussed in the section 3.4, to plot the features of each subject in training, validation and test set.

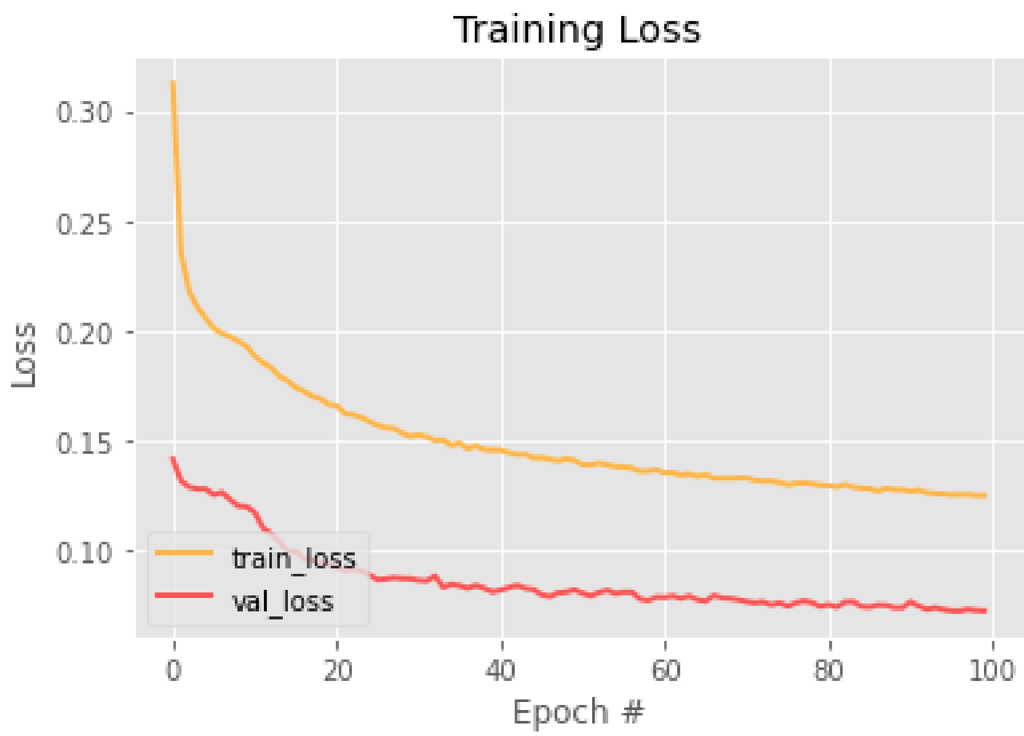

Figure 5.1: The loss plot of SNN with Contrastive loss in 100 epochs

We select 500 random samples from each target from known and unknown class in test set to feed to the trained model. We then plot the histogram distribution of the distance between each random pairs. As we can see an example from the Fig.5.3, the histogram shows the distribution

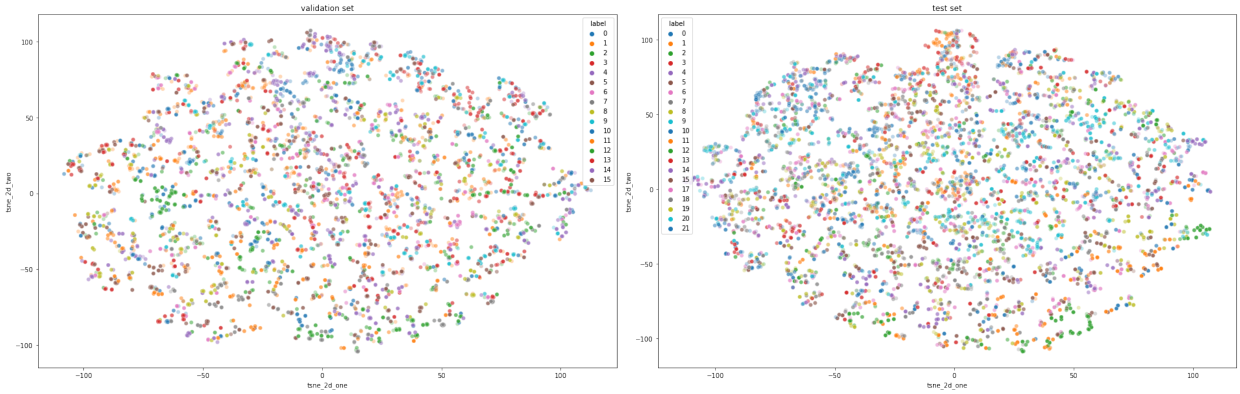

Figure 5.2: t-SNE plotting for validation and test set for model trained with Contrastive loss

of the distance between two samples from the same class and two samples from different class. We randomly select 500 pairs for the calculation. The anchor target is the one that did not appear in the training set. We compare the anchor target with targets from known classes. The example here is the target 21.

5.2.2 Triplet Loss

5.2.3 Contrastive Loss combined with Triplet Loss

5.3 Performance Evaluation

Several comments on the results can be derived from the plots above.

- Contrastive Loss: The validation loss is converging. However, the loss value has stopped decreasing significantly. It somehow shows that the training loss remains flat regardless of training. It is a sign of underfitting, indicating that model was unable to learn the training dataset. The t-SNE plot shows that the targets are not clustered well. We select 500 random samples for each target to compute t-SNE. We can see that the features of each target are randomly distributed. As a result, we can see in the Figure 5.3 a d 5.4, the model cannot show the distance in feature vectors between subject 21 and other known subjects in any case.

- Triplet Loss: We can see that the loss plot in Figure 5.5 shows the overfitting of the model. The model has learned the training dataset too well, including statistical noise and random fluctuations. Overfitting has the drawback of making a model less able to generalize to new data as it gets more specialized to training data, resulting in an increase in generalization error. This can be shown in the Figure 5.6 where ther train set t-SNE is clearly clustered, but the validation set t-SNE is somehow not clustered well depending on the target. In Figure 5.7 and 5.8, we can see that for some specific subjects, the model can distinguish them with the unknown subject 21. The data here is taken from the test set, where the model has not learnt from subject 21. We use cosine similarity to measure the similarity of two feature vectors in this case.

- Combination of Contrastive loss and Triplet loss: When combining the COntrastive Loss and Triplet Loss as the loss function of the model, specializing the loss weight of each loss in final loss combination, we expect a better result that the combination loss will help to reduce the underfitting and overfitting of two models above. However, the result obtained is not as expected. As we can see from the loss plot, the validation loss converges acceptably, with little sign of overfitting. But when we plot the t-SNE features, we can see that the model could not cluster the subjects in both validation and test set. As a result, the histogram plots show the same story with the first model trained with Contrastive loss.

Because the model trained with Triplet loss shows the most positive result among three models, we will further discuss about its result.

Since we only take 500 random samples to represent each target in the similarity measurement between targets, so that it hardly generalize the whole characteristics of the targets. If the sample size is increase, the result would tend to be more generalized. We consider the similarity distribution histograms, if the clearly separation between two histograms is observed, we note it as *classifed* - 1, else: *unclassified* - 0. The Tale 5.2 show the predicted similarity between 5 unknown targets (from 17 to 21) and 16 known targets (from 0 to 15). We can see that as for some subjects, the separation is clear between unknown and known (subject: 10, 11, 12, 14). These 4 subjects have a clear separation with all 5 unknowns subjects. Thus, the model is somehow bias with these 4 subjects that results in overfitting. The Table 5.1 shows the predicted separation result of 5 unknown targets versus 16 known targets. We also calculate the True Positive Rate and False Negative Rate of these predictions. As of these results, we can say that the model has been failed on detect the new unknown subjects among the group of known subjects.

Target	Predict 1	Predict 0	TPR = TP/(TP + FN)	FNR = FN/(TP + FN)
17	9	7	0.5625	0.4375
18	7	9	0.4375	0.5625
19	10	6	0.625	0.375
20	8	8	0.5	0.5
21	10	6	0.625	0.375

Table 5.1: The predicted separation result of unknown targets

Target	17	18	19	20	21
0	0	1	1	0	0
1	0	1	1	0	0
2	0	0	1	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	1	0	1	0	1
6	0	0	0	0	1
7	1	0	1	1	1
8	1	0	1	1	1
9	0	0	0	0	0
10	1	1	1	1	1
11	1	1	1	1	1
12	1	1	1	1	1
13	1	1	0	1	1
14	1	1	1	1	1
15	1	0	0	1	1

Table 5.2: The similarity between unknown targets and known targets.

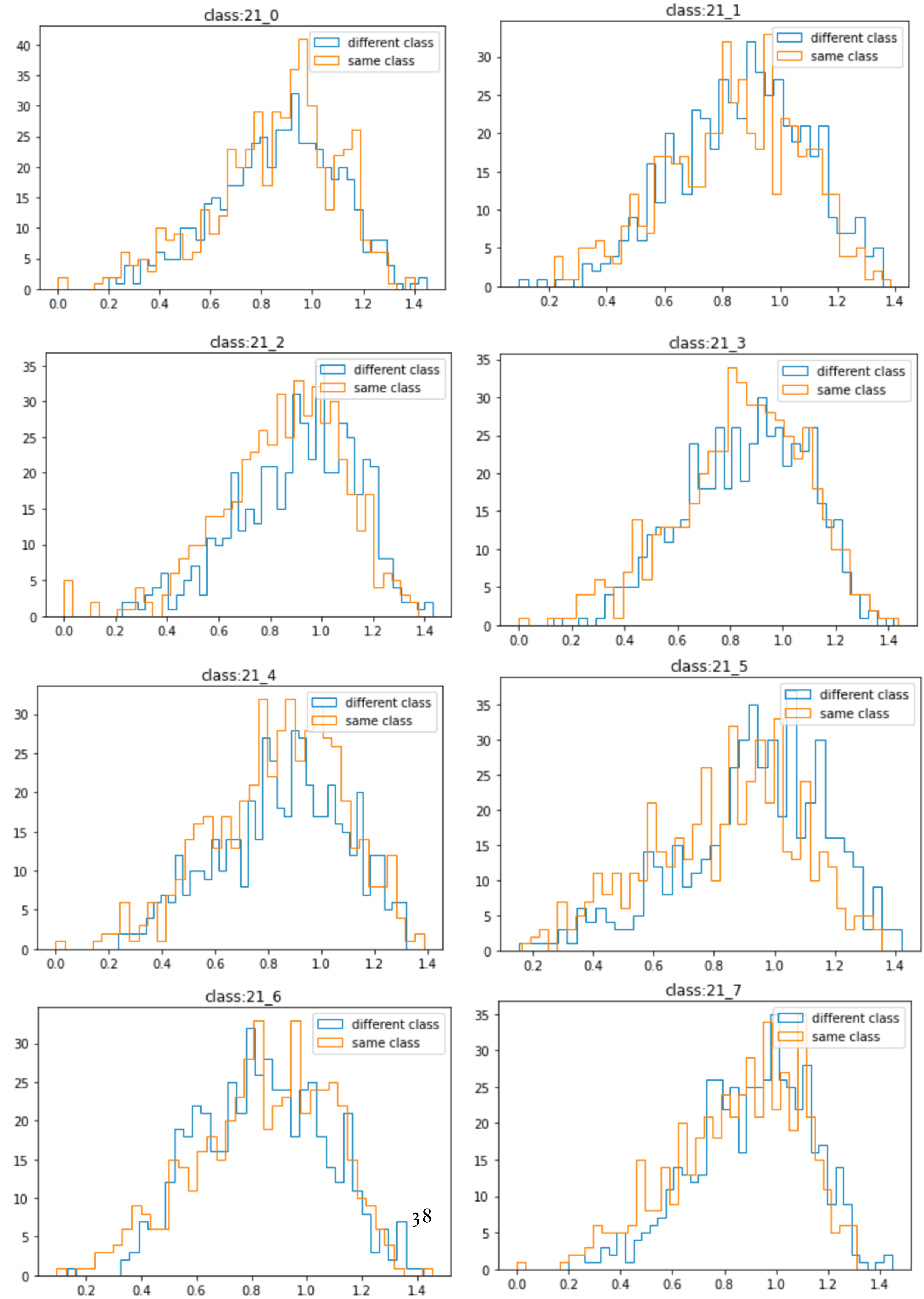

Figure 5.3: Histogram plotting of the distribution of the distance between the sample input of unknown targets 21 and the targets from 0 to 7, using 500 random samples - Contrastive Loss

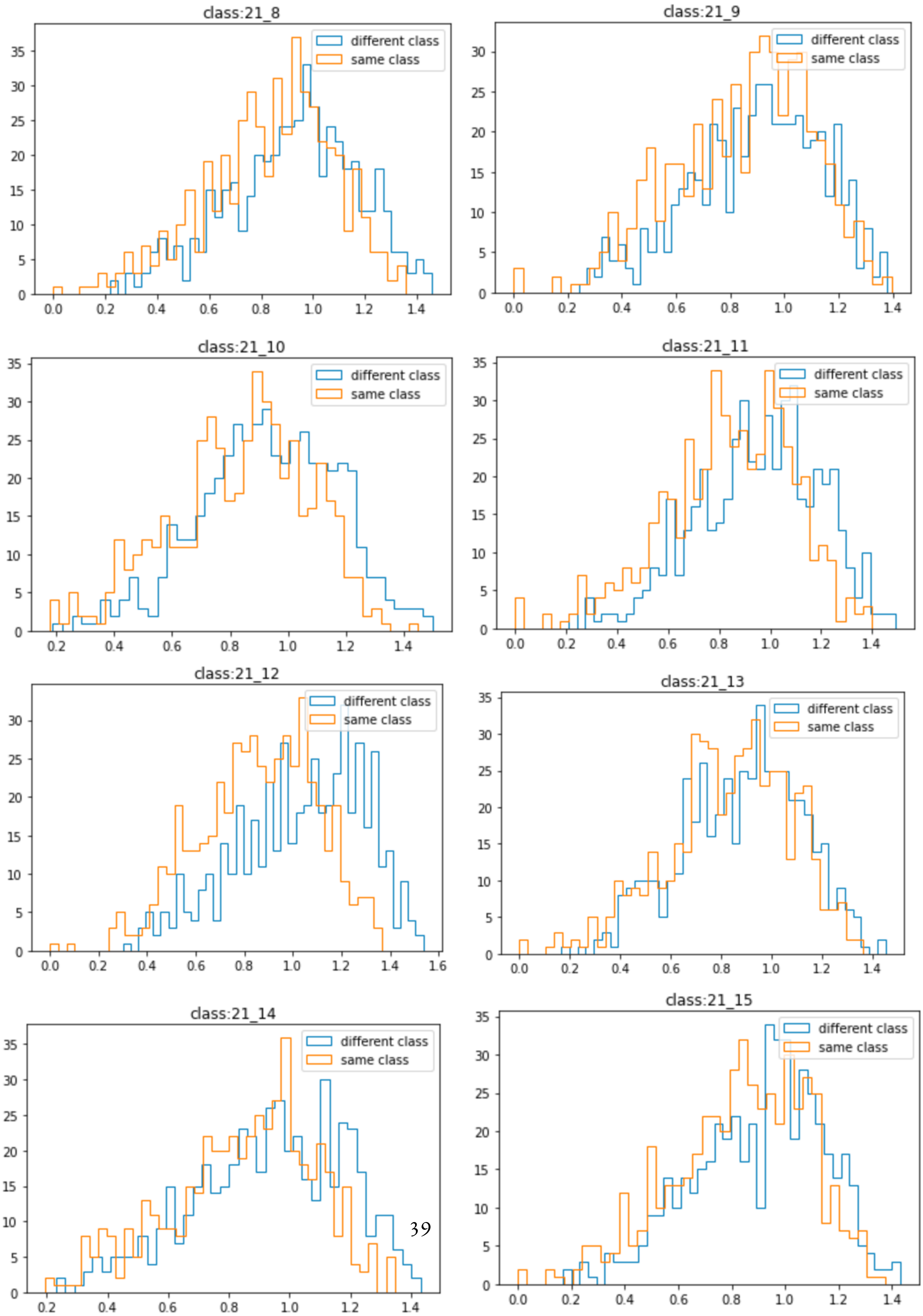

Figure 5.4: Histogram plotting of the distribution of the distance between the sample input of unknown targets 21 and the targets from 8 to 15, using 500 random samples - Contrastive Loss

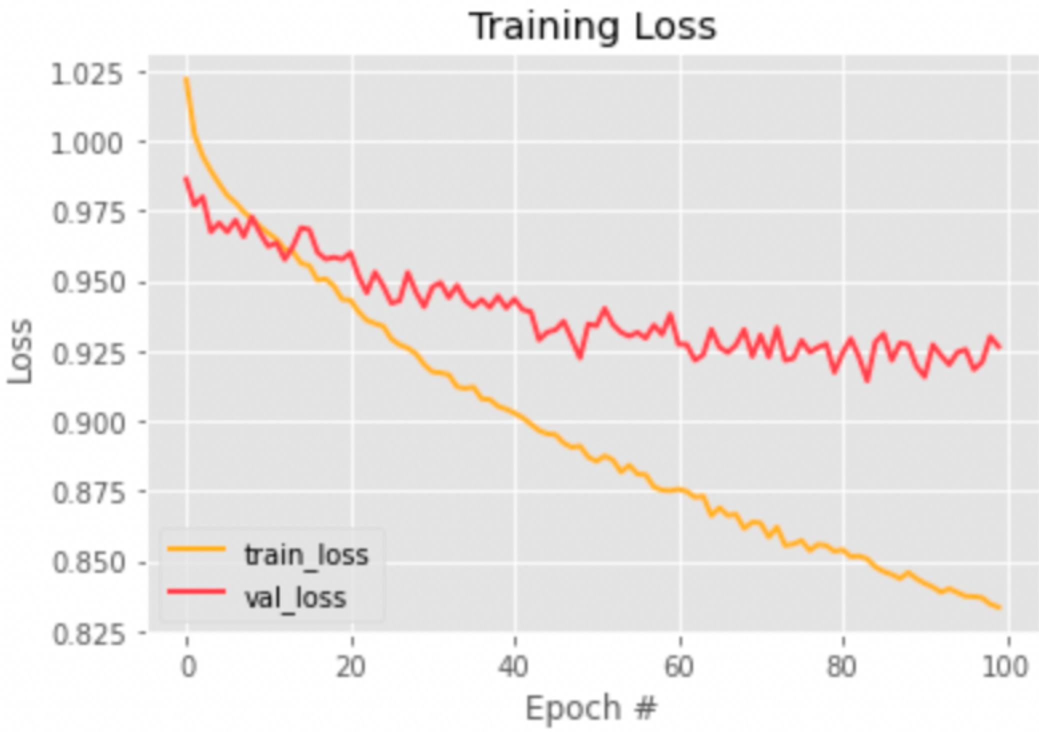

Figure 5.5: The loss plot of Triplet loss in 100 epochs

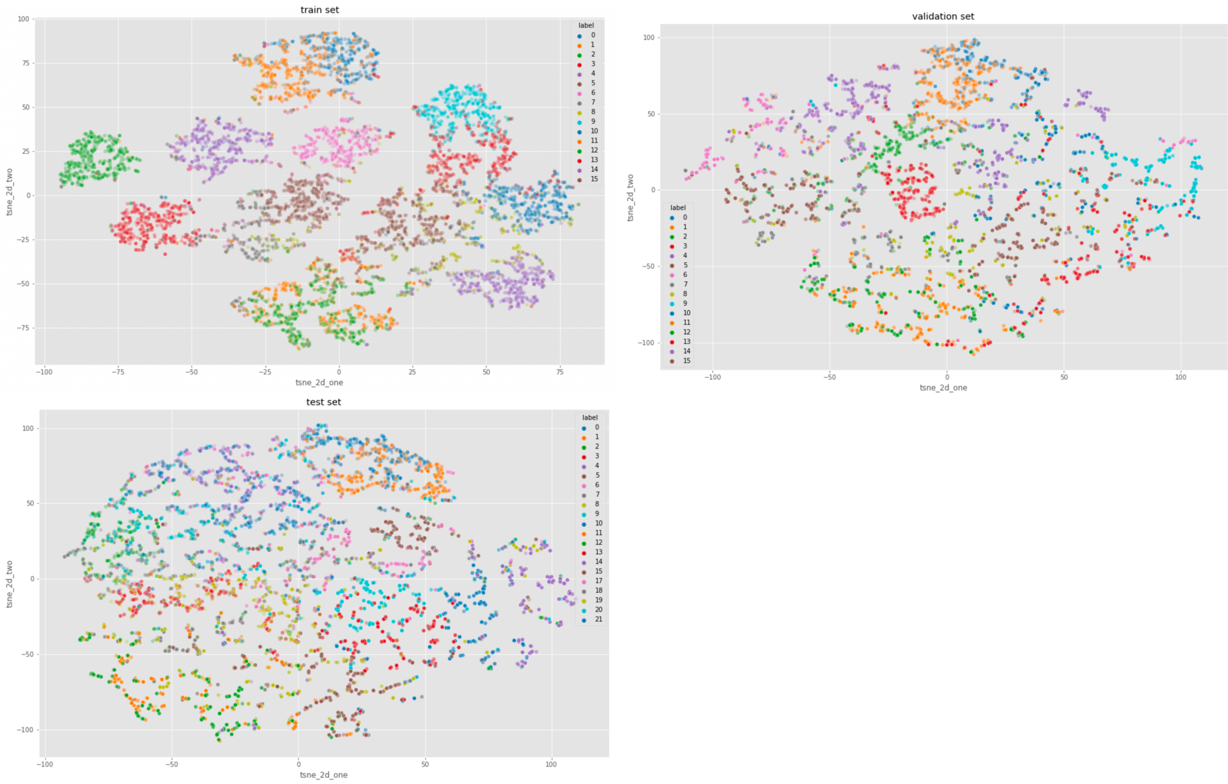

Figure 5.6: t-SNE plotting for train, validation and test set for model trained with Triplet loss

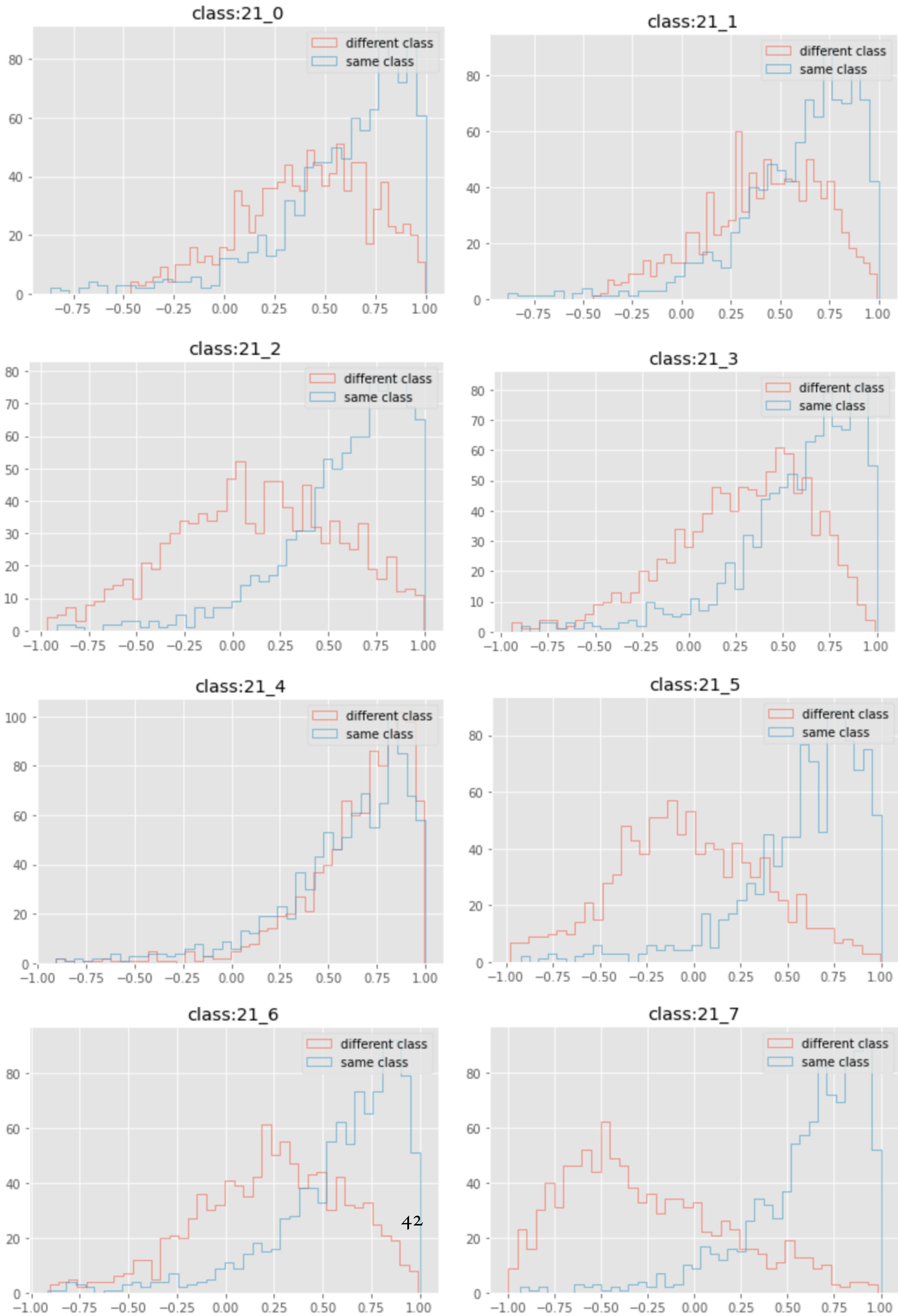

Figure 5.7: Histogram plotting of the distribution of the distance between the sample input of unknown targets 21 and the targets from 0 to 7, using 500 random samples - Triplet Loss

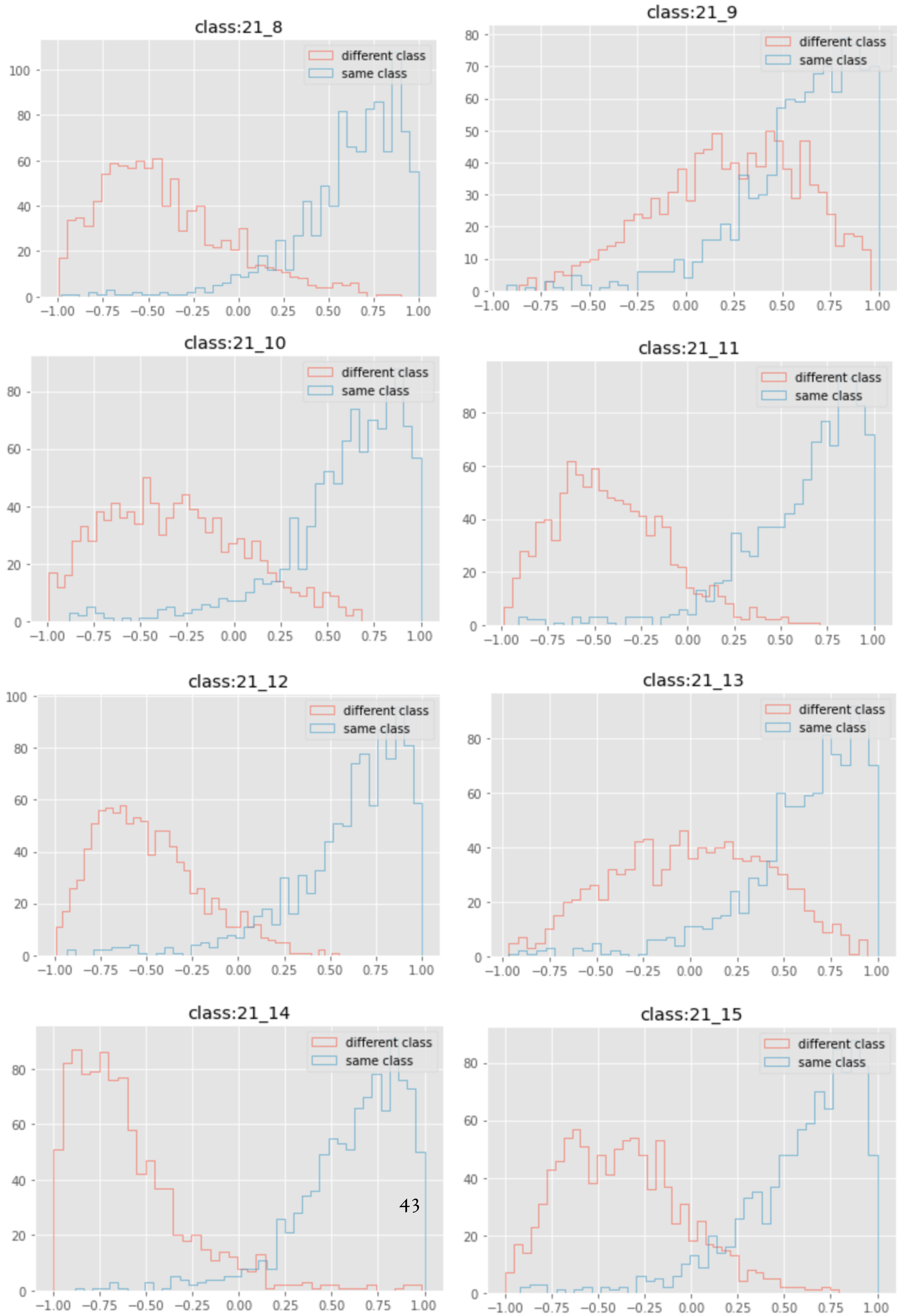

Figure 5.8: Histogram plotting of the distribution of the distance between the sample input of unknown targets 21 and the targets from 8 to 15, using 500 random samples - Triplet Loss

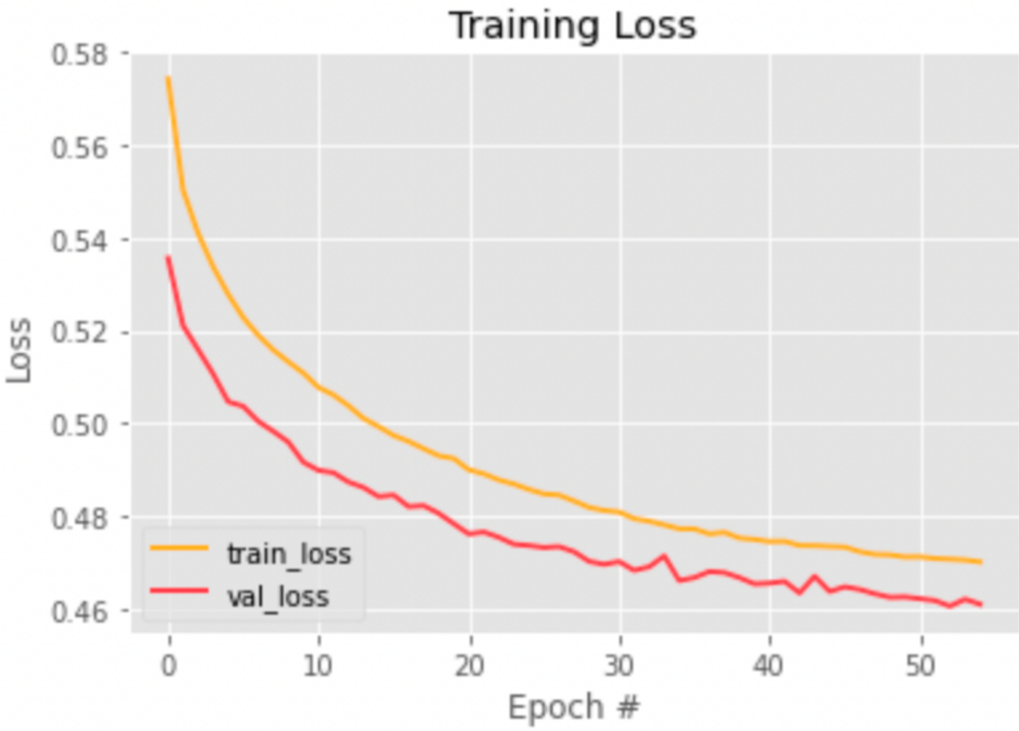

Figure 5.9: The loss plot of combined losses

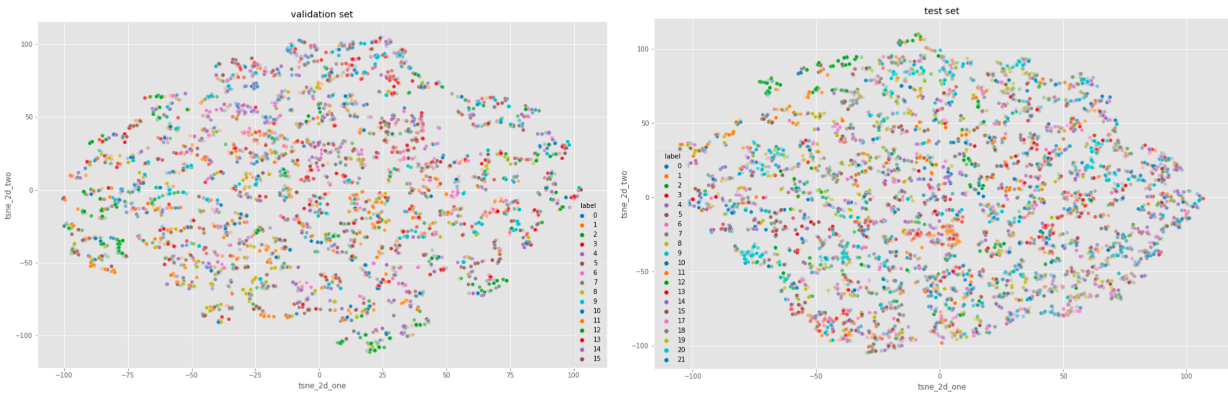

Figure 5.10: t-SNE plotting for validation and test set for model trained with combination of Contrastive loss and Triplet Loss

Figure 5.11: Histogram plotting of the distribution of the distance between the sample input of unknown targets 21 and the targets from 8 to 15, using 500 random samples - Combine Contrastive Loss and Triplet Loss

Figure 5.12: Histogram plotting of the distribution of the distance between the sample input of unknown targets 21 and the targets from 8 to 15, using 500 random samples - Combine Contrastive Loss and Triplet Loss

6 CONCLUSIONS AND FUTURE WORKS

6.1 Conclusions

The goal of this thesis was to identify one or more unknown people from a group of known subjects by exploiting signal data from a mm-Wave radar. We present an open-set person identification framework based on mm-Wave radar point-clouds, with the goal of distinguishing a new, unknown person from a known collection of people. The tasks in this work can be summarized as the following:

- The data used in the work was collected from different rooms to ensure the randomness. The raw data was the signal data of several mm-Wave radars that are from total of 21 targets with 16 known targets for training and 5 targets for testing.
- Instead of gathering and processing the raw data acquired from the backscattered mm-wave signal, we instead employ sparse point-clouds collected immediately from the radar device to keep the computing complexity low. On range-Doppler maps, the CA-CFAR technique is used to detect the primary reflecting sites, which involves applying a dynamic threshold to each RD value based on the power of nearby training data. The points detected by CA-CFAR are divided into numerous clusters using a density-based clustering technique, each of which corresponds to a different object in the environment. We then use a CM-KF, in a multi-target tracking (MTT) architecture, to estimate

the location, velocity, and extension of the subjects by processing the data output by the preceding phase.

- Feature extraction with TCPCN: a deep NN classifier is applied to a temporal series of K successive point-clouds associated with each trajectory. We use TCPCN for this task which is inspired by the standard PointNet architecture for 3D point-cloud classification and segmentation.

- Using different loss functions: We use the TCPCN combined with different loss functions choice such as Contrastive loss, Triplet Loss, and a combination between two of them.

6.2 Future Works

The results shows large room for future improvement. Here is a rundown of some potential developments.

- The triplet loss, on the other hand, focuses mostly on acquiring accurate ordering on the training set. It still has a poor capacity to generalize from the training set to the testing set, resulting in poor performance. So that we can try to train the model with more powerfull loss function such as Quadruplet loss [34], which, when compared to the triplet loss, can result in a model output with a bigger inter-class variation and a smaller intra-class variation.

- Improve on the current training by using cosine similarity as the measure of similarity between two feature vectors instead of using Euclidean distance in Contrastive Loss. Parameters tuning for the network and data augmentation to reduce overfitting when training the network with Triplet loss.

References

[1] J. Pegoraro and M. Rossi, "Real-time people tracking and identification from sparse mm-wave radar point-clouds," *IEEE*, vol. 9, pp. 78 504–78 520, May 2021.

[2] Z. Ni and B. Huang, "Open-set human identification based on gait radar micro-doppler signatures," *IEEE Sensors Journal*, vol. 21, no. 6, pp. 8226–8233, 2021.

[3] W. A. H. M. A. Richards, J. Scheer and W. L. Melvin, "Principles of modern radar," *Citeseer*, 2010.

[4] Monostatic radar vs bistatic radar-difference between monostatic radar and bistatic radar. [Online]. Available: https://www.rfwireless-world.com/Terminology/Monostatic-radar-vs-Bistatic-radar.html

[5] William Franklin. Kalman filter explained simply. [Online]. Available: https://thekalmanfilter.com/kalman-filter-explained-simply/

[6] 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.

[7] 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 *2017 European Radar Conference (EURAD)*, 2017, pp. 61–64.

[8] S. M. Patole, M. Torlak, D. Wang, and M. Ali, "Automotive radars: A review of signal processing techniques," *IEEE Signal Processing Magazine*, vol. 34, no. 2, pp. 22–35, 2017.

[9] A.-K. Seifert, M. G. Amin, and A. M. Zoubir, "Toward unobtrusive in-home gait analysis based on radar micro-doppler signatures," *IEEE Transactions on Biomedical Engineering*, vol. 66, no. 9, pp. 2629–2640, 2019.

[10] F. Jin, R. Zhang, A. Sengupta, S. Cao, S. Hariri, N. K. Agarwal, and S. K. Agarwal, "Multiple patients behavior detection in real-time using mmwave radar and deep cnns," in *2019 IEEE Radar Conference (RadarConf)*, 2019, pp. 1–6.

[11] A. J. I. C. A. B. W. D. N. B. Vandersmissen, N. Knudde and T. Dhaene, "Indoor person identification using a lowpower fmcw radar," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, pp. 3941–3952, Jul 2018.

[12] Y. L. G. Y. Y. H. Y. Yang, C. Hou and W. Xiang, "Person identification using micro-doppler signatures of human motions and uwb radar," *IEEE Microwave and Wireless Components Letters*, vol. 29, pp. 366–368, May 2019.

[13] F. F. Z. Chen, G. Li and H. Griffiths, "Personnel recognition and gait classification based on multistatic micro-doppler signatures using deep convolutional neural networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, pp. 669–673, May 2018.

[14] J. Pegoraro, F. Meneghello, and M. Rossi, "Multiperson continuous tracking and identification from mm-wave micro-doppler signatures," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 4, pp. 2994–3009, 2021.

[15] S. Z. Li and A. Jain, Eds., *Open-Set Identification*. Boston, MA: Springer US, 2009, pp. 1022–1022. [Online]. Available: https://doi.org/10.1007/978-0-387-73003-5_805

[16] K. M. C. R. Qi, H. Su and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (Honolulu, Hawaii, USA)*, Jul 2017.

[17] M. Y. J. Z. P. Cao, W. Xia and J. Zhou, "Radar-id: human identification based on radar micro-doppler signatures using deep convolutional neural networks," *IET Radar, Sonar Navigation*, vol. 12, pp. 729–734, Jul 2018.

[18] K. A. B. K. B. Y. S. Abdulatif, F. Aziz and U. Schneider, "Person identification and body mass index: A deep learning-based study on micro-dopplers," *IEEE Radar Conference (RadarConf), (Boston, Massachusetts USA)*, Apr 2019.

[19] W. D. N. A. Jalalvand, B. Vandersmissen and E. Mannens, "Radar signal processing for human identification by means of reservoir computing networks," *IEEE Radar Conference (RadarConf), (Boston, Massachusetts USA)*, Apr 2019.

[20] B. V. I. C. V. Polfliet, N. Knudde and T. Dhaene, "Structured inference networks using high-dimensional sensors for surveillance purposes," *International Conference on Engineering Applications of Neural Networks (EANN), (Crete, Greece)*, May 2018.

[21] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," *Computer Vision and Pattern Recognition*, vol. 1, p. 539–546, 2005.

[22] M. Shorfuzzaman and M. S. Hossain, "Metacovid: A siamese neural network framework with contrastive loss for n-shot diagnosis of covid-19 patients," *Pattern Recognition*, vol. 113, p. 107700, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320320305033

[23] Z. Wang, C. Peng, Y. Zhang, N. Wang, and L. Luo, "Fully convolutional siamese networks based change detection for optical aerial images with focal contrastive loss," *Neurocomputing*, vol. 457, pp. 155–167, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231221009838

[24] B. Ghojogh, M. Sikaroudi, S. Shafiei, H. Tizhoosh, F. Karray, and M. Crowley, "Fisher discriminant triplet and contrastive losses for training siamese networks," in *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1–7.

[25] Brian Williams. Contrastive loss explained. [Online]. Available: https://towardsdatascience.com/contrastive-loss-explaned-159f2d4a87ec

[26] Wikipedia. Triplet loss. [Online]. Available: https://en.wikipedia.org/wiki/Triplet_loss#:~:text=Triplet%20loss%20is%20a%20loss,matching%20input%20(called%20negative).

[27] Susmith Reddy. The intuition of triplet loss. [Online]. Available: https://medium.com/analytics-vidhya/triplet-loss-b9da35be21b8

[28] Wikipedia. t-distributed stochastic neighbor embedding. [Online]. Available: https://en.wikipedia.org/wiki/T-distributed_stochastic_neighbor_embedding

[29] Renu Khandelwal. T-distributed stochastic neighbor embedding(t-sne). [Online]. Available: https://towardsdatascience.com/t-distributed-stochastic-neighbor-embedding-t-sne-bb60ff109561

[30] Wikipedia. Kullback–leibler divergence. [Online]. Available: https://en.wikipedia.org/wiki/Kullback%E2%80%93Leibler_divergence

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

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

[33] C. M. Bishop, *Pattern recognition and machine learning*, 2007.

[34] J. Z. Weihua Chen, Xiaotang Chen and K. Huang, "Beyond triplet loss: a deep quadruplet network for person re-identification," *arXiv*, 2017.

Acknowledgments

There were numerous hurdles and roadblocks encountered during the course of research. I would like to send a big Thank to Jacopo Pegoraro, my co-supervisor, who has always been kind and patient with me, providing me with constructive criticism and encouraging recommendations. Without his support and guidance, the project and research would not have been completed. Furthermore, Professor Michele Rossi, who provided me with the opportunity to learn and apply deep learning to a real-world and practical problem at his research lab, SIGNET, deserves my sincere appreciation. Nonetheless, I want to express my gratitude to the entire staff of the University of Padua for the knowledge and passion you have shared with me during my Master's studies.

Throughout the duration of this endeavor and my studies, I am grateful for the love and support of my family and friends. I was able to keep going because of the love and support I received.

Finally, I'd want to express my gratitude to my colleagues at the University of Padua. I regret not being able to connect with you when I had the opportunity. Regardless, you have provided me with a pleasant remembrance of my brief trip in Padua.