AUTOMATIC TARGET RECOGNITION OF QUADCOPTER TYPE DRONES FROM MODERATELY-WIDEBAND ELECTROMAGNETIC DATA USING CONVOLUTIONAL NEURAL NETWORKS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

RUTKAY GÜNERİ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

DECEMBER 2022

Approval of the thesis:

AUTOMATIC TARGET RECOGNITION OF QUADCOPTER TYPE DRONES FROM MODERATELY-WIDEBAND ELECTROMAGNETIC DATA USING CONVOLUTIONAL NEURAL NETWORKS

submitted by **RUTKAY GÜNERİ** in partial fulfillment of the requirements for the degree of **Master of Science** i**n Electrical and Electronic Engineering, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. İlkay Ulusoy
Head of Department, **Electrical and Electronics Engineering** _____

Prof. Dr. Gönül Turhan Sayan
Supervisor, **Electrical and Electronics Engineering, METU** _____

Examining Committee Members:

Prof. Dr. Gülbin Dural
Electrical and Electronics Engineering, METU _____

Prof. Dr. Gönül Turhan Sayan
Electrical and Electronics Engineering, METU _____

Prof. Dr. Mustafa Kuzuoğlu
Electrical and Electronics Engineering, METU _____

Prof. Dr. Gözde Bozdağı Akar
Electrical and Electronics Engineering, METU _____

Prof. Dr. Asım Egemen Yılmaz
Electrical and Electronics Engineering, Ankara University _____

Date: 15.12.2022

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name Last name: Rutkay Güneri

Signature:

ABSTRACT

AUTOMATIC TARGET RECOGNITION OF QUADCOPTER TYPE DRONES FROM MODERATELY-WIDEBAND ELECTROMAGNETIC DATA USING CONVOLUTIONAL NEURAL NETWORKS

Güneri, Rutkay
Master of Science, Electrical and Electronic Engineering
Supervisor: Prof. Dr. Gönül Turhan Sayan

December 2022, 106 pages

In this thesis, the classifier design approach based on "Learning by a *Convolutional Neural Network (CNN)*" will be applied to two different target library/data sets; an ultra-wideband simulation data (from 37 MHz to 19.1 GHz) obtained for a target library of four dielectric spheres, and a moderately-wide band measurement data (from 3.1 to 4.8 GHz) obtained for a target library of four quadcopter type unmanned aerial vehicles (UAVs). While the bandwidth of simulation data for spherical targets is about nine octaves, the bandwidth of measurement data collected for quadcopters is even less than one octave.

As the first task, a CNN-based electromagnetic target classifier will be designed for the spherical targets using that spectrally rich simulated database. Then, its performance will be compared to the performance of another classifier that has been already reported in automatic target recognition (ATR) literature as designed by the "*Wigner Distribution-Principle Component* (*WD-PCA) based Feature Extraction"* technique using the same target library and the same database.

After verifying the effectiveness of the CNN-based classifier design aproach through this comparative investigation, a second CNN-based classifier will be designed for the quadcopter type UAVs using their challenging scattered database of very modest spectral bandwidth. Design details and performances of each classifier will be presented through the thesis discussing the advantages and disadvantages of the CNN-based classifier design approach.

Keywords: Automatic Target Recognition, Electromagnetic Target Classification, Convolutional Neural Network (CNN), CNN Learning for One-Dimensional Data, Quadcopter type UAV Classification.

ÖZ

DÖRT PERVANELİ ROBOT HELİKOPTER TİPİ İNSANSIZ HAVA ARAÇLARININ EVRİŞİMLİ SİNİR AĞLARI KULLANILARAK ORTA DERECEDE BANT GENİŞLİĞİNE SAHİP ELEKTROMANYETİK SİNYALLERDEN OTOMATİK YÖNTEMLERLE TANINMASI

Güneri, Rutkay
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği
Tez Yöneticisi: Prof. Dr. Gönül Turhan Sayan

Aralık 2022, 106 sayfa

Bu tezde, "*Evrişimli Sinir Ağı (CNN) ile Öğrenme*" yöntemine dayalı sınıflandırıcı tasarım yaklaşımı, iki farklı hedef kütüphanesi/veri seti için uygulanacaktır. İlk veri seti, dört dielektrik küreden oluşan bir hedef kütüphanesi için elde edilen ultra-geniş bantlı (37 MHz'den 19.1 GHz'e kadar) simülasyon verilerini içerirken, ikinci veri seti dört pervaneli robot helikopter tipi insansız hava araçlarından (İHA'lar) oluşan bir hedef kütüphanesi için orta derecede bant genişliğine sahip (3.1 ila 4.8 GHz arasında) ölçülmüş verilerden oluşmaktadır. Küresel hedefler için hesaplanan simülasyon verilerinin bant genişliği yaklaşık dokuz oktav iken, quadkopterler için toplanan ölçüm verilerinin bant genişliği bir oktavdan azdır.

İlk iş olarak, dielektrik kürelere ait ultra-geniş bantlı simülasyon veritabanı kullanılarak küresel hedeflerin tanınmasına yönelik CNN-tabanlı bir elektromanyetik hedef sınıflandırıcı tasarlanacaktır. Daha sonra, bu hedef sınıflandırıcının performansı, aynı hedef seti ve aynı veritabanı kullanılarak tasarlanmış, otomatik hedef tanıma (ATR) literatüründe yer alan ve "*Wigner Dağılımı-Temel Bileşenler Analizi (WD-PCA) tabanlı Öznitelik Çıkarma*" tekniği

olarak adlandırlan yönteme dayanan diğer bir hedef sınıflandırıcının performansı ile karşılaştırılacaktır.

Sözü edilen karşılaştırmalı inceleme sonucunda etkinliğinin doğrulanmasını takiben, CNN-tabanlı hedef tanıma yöntemi dört pervaneli robot helikopter tipi hedefler için de hedef sınıflandırıcı tasarımı için kullanılacaktır. Bu ikinci tasarımda düşük bant genişliğine sahip dezavantajlı bir veri tabanının kullanılacak olması problemin zorluk derecesini arttırmaktadır. Tasarlanan iki sınıflandırıcının da tasarım detayları ve performans sonuçları tez içerisinde verilecek, CNN-tabanlı sınıflandırıcı tasarım yönteminin avantaj ve dezavantajları tartışılacaktır.

Anahtar Kelimeler: Otomatik Hedef Tanıma, Elektromanyetik Hedef Sınıflandırma, Evrişimli Sinir Ağı (CNN), Tek-Boyutlu Veriler için CNN Öğrenimi, Dört Pervaneli Robot Helikopter Tipi İHA Sınıflandırması.

To my family

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my supervisor Prof. Dr. Gönül Turhan-Sayan for her invaluable guidance, advice, criticism, encouragements, and insight throughout the research.

I also would like to thank Prof. Dr. Kemal Leblebicioğlu and Raha Shabani for providing quadcopters, and to thank Orhan Bardak for his support during measurements.

I deeply appreciate Prof. Dr. Gözde Bozdağı Akar for the valuable feedback she gave during the writing of the thesis.

I also would like to thank my colleagues Utku Kaydok and Buğra Sofu for their advice and contribution to this study. In addition, I would like to thank my colleagues Utku Kaydok and Onur Baki Ertin for providing quadcopters.

Lastly, I would like to express my gratitude to my parents Ayşe and Orhan Güneri and my dear brother Gökay Güneri for their support, encouragement, and confidence throughout my life.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS

1D	1 Dimensional
ATR	Automatic Target Recognition
CNN	Convolutional Neural Network
FCL	Fully Connected Layer
RELU	Rectified Linear Unit
PCA	Principal Component Analysis
SEM	Singularity Expansion Method
SNR	Signal-to-Noise Ratio
TFR	Time-Frequency Representation
UAV	Unmanned Aerial Vehicle
UWB	Ultra-wideband
WD	Wigner Distribution

CHAPTER 1

INTRODUCTION

1.1 Motivation and Problem Definition

Automatic Target Recognition (ATR) is a wide research area where both civilian and military applications exist. A broad variety of techniques could be used to distinguish objects from each other based on the types of data collected by sensors which may operate at different frequency regimes and in different data formats in a given ATR scenario. For example, electro-optical sensors such as FLIR (Forward Looking Infrared) cameras or radars operating at microwave frequencies can be used in detection and recognition of targets with quite different techniques used for signal processing and interpretation. It is a critical task to quickly and correctly recognize a detected target in certain applications such as the classification of a plane as a fighter aircraft or a passenger airplane in battlefield or around an airport. Such applications call for the use of real-time ATR techniques. In order to recognize or classify a target successfully, manifestation of the target properties (such as its size, shape, material composition, speed, temperature, etc.) in the collected data needs to be well understood.

Design of electromagnetic target classifiers must utilize multi-aspect and/or multi-polarization data because of the strong dependence of scattered signals on aspect angle and polarization. As described by the Singularity Expansion Method (SEM), complex natural resonance (CNR) frequencies are aspect/polarization independent descriptors of an object, hence they have the power to characterize a target uniquely as the bandwidth of electromagnetic data approaches to infinity [1-4]. Therefore, use of ultra-wideband scattered signals is strongly preferred in electromagnetic target

classifier design to provide excellent time resolution and highly improved accuracy in the extraction of physics-based target features which are indirectly based on complex natural resonance (CNR) frequencies [5-8]. The quality of extracted target features is expected to decline for moderately-wide data bandwidths leading to decreasing classifier performance [9]. In such a case, if a sufficiently large amount of scattered data is available, it may be not only meaningful but also necessary to investigate the usefulness of an alternative target classification technique that is based on deep learning algorithms.

In this thesis, use of two different classifier design approaches, the "*Wigner Distribution-Principal Component Analysis (WD-PCA) Method*" that is a well-established ATR method based on physics-based feature extraction [10-11] and the "*Learning by Convolutional Neural Network (CNN) Method*" [12-13] will be investigated in a comparative manner. The CNN-based design approach will be applied to design electromagnetic target classifiers from scratch for two different target library/data sets; an ultra-wideband analytically computed simulation data, from 37 MHz to about 19GHz, obtained for a target library of four similar dielectric spheres, and a moderately-wide band measurement data, from 3.1 to 4.8 GHz, obtained for a target library of four quadcopter type drones. While the bandwidth of simulation data for spherical targets is about nine octaves, the bandwidth of measurement data collected for quadcopters is relatively poor, less than one octave.

As the first task, a CNN-based electromagnetic target classifier will be designed for the spherical targets using that spectrally rich simulated database. Then, its performance will be compared to the performance of another classifier that has been already reported in automatic target recognition (ATR) literature [11] as designed by the WD-PCA based technique using the same target library and the same database.

After verifying the effectiveness of the CNN-based classifier design aproach through this comparative investigation, a second CNN-based classifier will be designed for the quadcopter type UAVs using their challenging scattered database of very modest spectral bandwidth. Design details and performances of each classifier will be

presented through the thesis discussing the advantages and disadvantages of the CNN-based classifier design approach.

Use of Artificial Neural Networks (ANNs) in ATR problems have been demonstrated in literature since late 80's and 90's [14-17]. In recent years with the advancements in artificial intelligence, different deep learning methods are introduced [18] and applied to a wide variety of fields including radar signal processing [19] where the use of deep learning based algorithms are ranging from radar waveform recognition to automatic target recognition. Applications of convolutional neural networks (CNNs) to one-dimensional data [20-22] and especially to two-dimensional (i.e. image type) data [12], [23-25] are becoming more common for the purpose of automatic target recognition. When exposed to very large training datasets, CNNs can learn target features hidden in the input signals by the help of the convolutional layers. Elimination of the need for specially designed feature extraction steps in target classifier design looks to be the major advantage of the CNN learning approach.

1.2 Related Work and Literature Review

The use of ultra-wideband electromagnetic signals to identify objects was initiated by the Kennaugh's work in 1958 [1]. For the first time, Kennaugh et al. applied the impulse response concept to electromagnetic scattering problems, and it was observed that damped sinusoidal oscillations occur in the transient responses of targets. This behavior was associated with complex poles (i.e. complex natural resonance (CNR) frequencies) of the targets' system functions and formulated by Baum as Singularity Expansion Method (SEM) in mid-70's [3]. The CNR frequency values of a target are affected by the physical and electrical properties of a given target such as its size, geometry, material composition and electrical parameters (permittivity, permeability and conductivity). The CNR frequencies are perfectly independent of aspect angle and polarization. However, the residues associated with these system poles are strong functions of aspect/polarization. Consequently, the

scattered response of a target is highly dependent on these measurement conditions. For instance, for a given target, time response at two different aspect angles may look very different from each other. Yet, target responses belonging to two different targets may look very similar at a given aspect angle. Due to these facts, feature extraction from raw electromagnetic scattered data is a crucial step in various target recognition methods. Further details of the SEM formalism and the linear, time-invariant representation of the aspect dependent system function of a target are given in Appendix A.

Early ATR research based on the SEM theory aimed to extract target poles (i.e. the CNR frequencies) directly from the electromagnetic scattered signals [2]. However, the success of this approach was shown to degrade dramatically in the presence of noise. Due to even small errors in estimated pole values, the accuracy of overall classification performance deteriorates quickly as the signal-to-noise ratio (SNR) of data gets lower.

As opposed to using the CNR frequencies directly for ATR purposes, Kennaugh proposed another target identification method based on the indirect utilization of SEM methodology in 1981 [4]. Designing a special time limited excitation signal, called Kill pulse (K-pulse), was the main idea in this method. K-pulse should be designed in a way to annihilate target poles. So, the late-time response of the target becomes time-limited only when the target is excited by its own K-pulse [5-6].

Later, quadratic time-frequency representation (TFR) techniques [26] such as Wigner-Ville distribution and Page distribution, have been used to obtain energy based target features which are indirectly related to the information of complex natural frequencies [10-11], [16]. In this case, the late-time portion of the scattered electromagnetic signals are used (after the forced response is completely over) for target feature extraction in a way to utilize the effects of target poles only. In [16], the late-time features are extracted by Wigner distribution (WD) and a self-organizing map type neural network (SOM) is used for classification. Another WD-based method is proposed by Turhan-Sayan in [10]. The classifier decides on the

correct target choosing the maximum of all correlation coefficients computed between the late-time feature vector of the test signal and reference feature vectors, which are obtained at a small number of training aspect angles for each target in the classifier library. This method is further improved in [11] by using the principal component analysis (PCA) to merge the reference feature vectors computed for a given target at all training aspect angles to obtain a single target-specific feature vector [40]. In addition to WD-based classifiers, Seçmen and Turhan-Sayan proposed in [8] that MUSIC algorithm could be used to extract CNR-based features from the late-time response. Similar to [11], the feature matrices obtained by MUSIC algorithm in only a few reference angles are combined, and a single feature called Fused MUSIC Spectrum Matrix (FMSM) is obtained for each target. The MUSIC algorithm-based method is also applied for multiple target recognition in [27].

With the developments in artificial intelligence, the use of different types of neural networks in electromagnetic target recognition applications has increased. The application of conventional neural networks (supervised or unsupervised) still needed the use of feature extraction for NN training for better ATR results [16-17]. However, the need for hand-crafted feature extraction prior to neural network training is completely eliminated by the convolutional neural networks (CNNs). For example, Selver et al in [21] used a 1D convolutional neural network to classify small scale model aircrafts. The work was based on the 1D CNN foundation stated by Kiranyaz et al in [20] for ECG signal classification. In [21], the classifier designed for small scale model aircraft was composed of three stages. In the first stage, the neural networks are trained with analytically generated and simulated scattering signals of various spheres and dielectric rods. By using transfer learning, the neural networks trained in the first stage are fine-tuned with the small number of measured signals of model aircraft in the second stage. At last, the results of each neural network are ensembled to obtain the final output in the third stage.

The CNN based three stage classifier mentioned above is improved by Toprak et al in [22]. In this work, two different methods are proposed. The first method utilizes magnetic spheres when training the first stage CNN in addition to Selver et al [21].

So, new diverse features are obtained with the addition of the magnetic properties. Secondly, a new architecture is proposed which includes long short-term memory (LSTM) network along with CNN. By the CNN-LSTM architecture, the small number of scattered signals of small scale model aircraft are used directly to train the proposed network.

1.3 Novel Contributions of the Thesis

The novel contribution of this thesis is the design of a CNN-based electromagnetic target classifier for a library of four similar quadcopter type commercial drones using totally experimental, one dimensional (1-D) aspect/polarization dependent scattered field data with a highly restricted bandwidth of 3.1 to 4.8 GHz. Previous research papers reporting CNN-based classifiers for 1-D time-domain signals in ATR design either use very low frequency ECG signals [20] which do not suffer from the physically dictated complications of aspect/polarization dependencies, or they use electromagnetic scattered data like we use but utilize ultra-wideband simulation data as well as ultra-wideband measurement data not for full-size targets but for small-scale targets over a very large bandwidth of 1-12 GHz in their design work [21-22].

1.4 The Outline of the Thesis

In Chapter 1, problem definition and motivation of the study are presented together with literature survey. The novel contributions of the thesis are also mentioned briefly in this Introduction chapter.

Basic theoretical background of artificial neural networks is outlined in Chapter 2.

Design of a CNN-based electromagnetic classifier for lossless dielectric spheres using ultra-wideband simulation data is presented in full detail in Chapter 3. The performance of this classifier is compared to the performance of another classifier

reported in literature (designed by the WD-PCA based feature extraction method) to verify the capacity and usefulness of the CNN-based target classifiers.

In Chapter 4, the CNN-based classification approach is used to classify quadcopter type UAVs. Target parameters, properties of measured database and measurement setup are presented in this chapter together with full details of the classifier design process.

Finally, the conclusions and plans for future work are given in Chapter 5.

The thesis has also six Appendices.

Appendix A provides details on the Singularity Expansion Method (SEM) and summarizes the representation of system response for an electromagnetic target.

Appendix B presents waveform plots for scattered signals in the dielectric sphere database.

Performance metrics of the CNN-based classifiers are defined in Appendix C.

Appendix D presents the confusion matrices obtained in the testing stage of the CNN-based classifier with noisy signals belonging to the dielectric sphere database.

Details of experimental setup used for the measurement of scattered fields of the quadcopter type UAVs are presented in detail in Appendix E.

Finally, the Phyton codes developed for the CNN-based dielectric sphere classifier and the CNN-based quadcopter type UAV classifier are provided in Appendix F and Appendix G, respectively.

CHAPTER 2

BACKGROUND FOR NEURAL NETWORKS

In this chapter, the basic background of artificial neural networks (ANNs) and convolution neural networks (CNNs) are explained briefly for the sake of completeness.

2.1 Artificial Neural Networks

Neural networks are one of the most popular fields in machine learning. The building blocks of an artificial neural network are neurons which are inspired by biological neural systems [35]. A neuron is simply a computational unit that produces an output according to its input values as shown in Figure 2.1. At first, input values are multiplied by the corresponding weights of the neuron. The weighted inputs are summed, and a bias is added to the summation. Then the result is pass through an activation function to produce the output of the neuron. There are commonly used activation functions in neural networks such as sigmoid, tanh, and rectified linear unit (ReLU), which are shown in Figure 2.2.

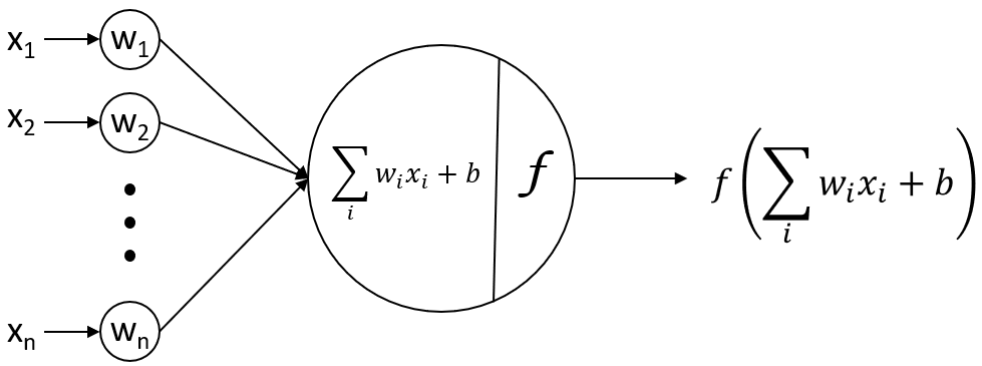
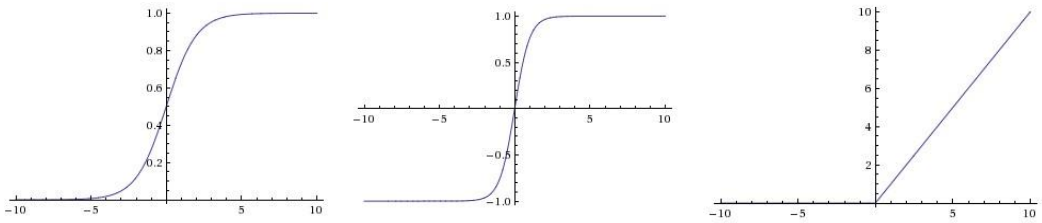


Figure 2.1. Mathematical model of a neuron



(a) (b) (c)

Figure 2.2. Commonly used activation functions (a) sigmoid (b) tanh (c) rectified linear unit (ReLU) [35]

Neurons are organized in layers to create a neural network. The architecture of a neural network is composed of an input layer, several hidden layers, and an output layer in general. As seen in Figure 2.3, a neuron is connected to all neurons in the previous and the preceding layer; therefore, this type of layer is named fully connected layer. In addition, there is no connection between neurons in the same layer. An artificial neural network is named according to the number of layers. Input layer, however, is not counted for the total number of layers in a neural network. For example, the network illustrated in Figure 2.3 is a 2-layer neural network composed of a single hidden layer and an output layer. Meanwhile, the number of hidden layers and neurons in each layer is a design consideration.

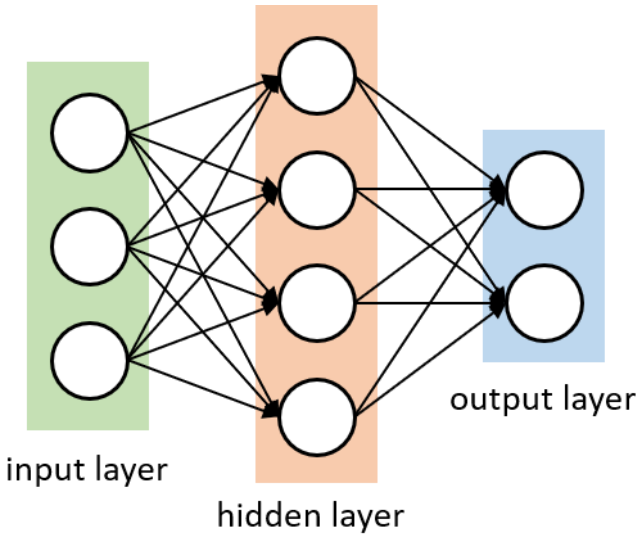


Figure 2.3. A 2-layer Neural Network (one hidden layer of 4 neurons (or units) and one output layer with 2 neurons), and three inputs

The training process of a neural network could be explained basically as optimizing the weights of the network to minimize the loss function. As seen in Figure 2.4, two processes contribute to the loss function which are data loss and regularization loss. The data loss arises from the difference between the output of the network and the true labels. On the contrary, only the weights determine the regularization loss. Various optimization algorithms are preferred to minimize the loss function of a neural network. For example, Gradient Descent and Adam are the popular choices which are iterative optimization techniques. While training a neural network, the weights are randomly initialized at the beginning, and the training dataset is passed through the network. The output is then compared with the true labels to calculate the data loss. In every iteration, the parameters are updated regarding to the loss value.

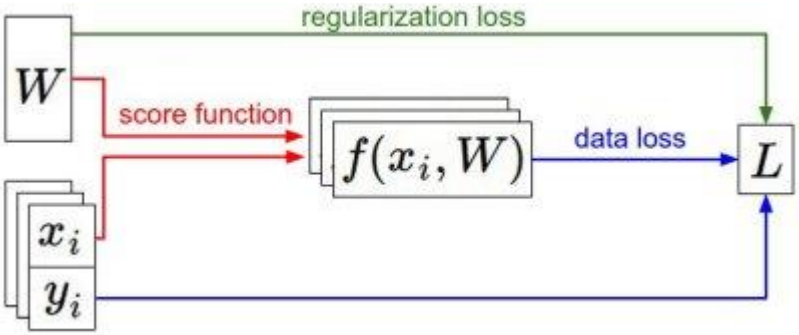


Figure 2.4. Summary of forward propagation to compute loss function [35]

2.2 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a special class of artificial neural networks (ANNs) used commonly in the resent years. Although, these networks are broadly preferred in image and video processing problems, numerous applications of CNNs are reported in the field of radar signal processing [35]. The main difference of CNNs from the regular artificial neural networks is the convolutional layers. Different from a fully connected layer, neurons are connected to a small portion of the previous layer while the convolution operation is performed. Since the parameters of the convolutional layers are trainable, a CNN can learn features of the concerned signal. By this way the feature extraction effort in signal processing research is eliminated.

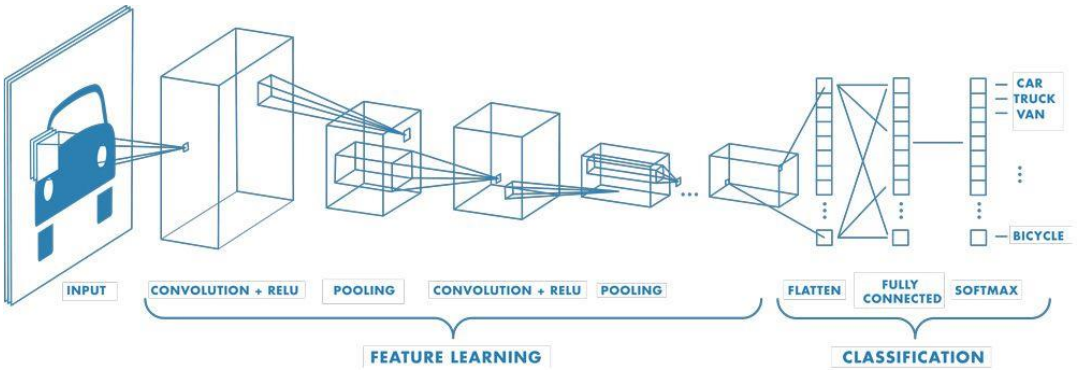


Figure 2.5. A convolutional neural network illustration [36]

A convolutional neural network is composed of several different layers which are convolutional layer, pooling layer, and fully connected layer (FCL) as shown in Figure 2.5. Convolutional layer is the layer where convolutional operation between the output of the previous layer and each neuron in the layer is computed. Similar to fully connected layers, an activation function is placed at the output of convolutional layer neurons, where rectified linear unit (ReLU) is selected in general. After the convolutional layers, down-sampling operation is performed in the pooling layers so that the spatial size is reduced. At the end, CNNs include fully connected layers at the output. The classification of the input is handled by the fully connected layers. The chosen classifier is varying according to number of target classes. For example, sigmoid function is commonly used for binary classification. Meanwhile, softmax is a very popular choice in classification of multiple targets.

CHAPTER 3

ELECTROMAGNETIC TARGET CLASSIFIER DESIGN FOR DIELECTRIC SPHERES BY CONVOLUTIONAL NEURAL NETWORKS

In this chapter, we will develop the design procedure for a convolutional neural network (CNN) based target classifier for the target library of four lossless dielectric spheres using an ultra-wideband analytically simulated database. Then, the performance of the resulting CNN-based classifier will be compared to the performance of a WD-PCA based classifier designed for the same spherical target set using exactly the same database to validate the effectiveness of the CNN-based classifier. As the design details and the performance results for the WD-PCA classifier are already reported in literature [11], WD-PCA classifier design for the spherical targets will not be repeated in this work.

3.1 Properties of the Target Library

The target library is composed of four lossless dielectric spheres with the same size (with diameter of 10 cm) but slightly different permittivity values as presented in Table 3.1. These are exactly the same targets used in [11]. Although the geometry of the spherical targets looks perfectly symmetrical and simple, classification of dielectric spheres is a challenging problem as a dielectric sphere has a highly dense target pole pattern due to interior resonating modes [6-7]. Using spherical targets in classifier design [8,10-11,21-22, 27] comes handy as the ultra-wideband far field scattered signals can be conveniently computed by using the Mie series formulation [28].

Table 3.1 Properties of the dielectric spheres

	ε_r	μ_r	*Radius (cm)*
Target 1	3	1	10
Target 2	4	1	10
Target 3	5	1	10
Target 4	6	1	10

3.2 Properties of the Database Simulated for the Dielectric Spheres

The far-field scattered responses from each spherical target were already computed at 13 different bi-static aspects by Turhan-Sayan [10] in frequency domain using the excitation/observation scenario presented in Figure 3.1. Then, the resulting frequency domain data were windowed to minimize Gibbs phenomena and transformed to time domain by using the inverse fast Fourier transformation (IFFT). Signals in the time-domain database were also normalized to have unit energy prior to being used in the WD-PCA based feature extraction and classifier design approach. One of the important aspects in WD-PCA classifier design is to minimize the volume of data needed for design. Accordingly, the scattered field database was created at only 13 aspects; only 5 of these aspects were used in classifier design, and the remaining 8 aspects were used for testing the classifier. In the case of deep learning approach however, the neural networks such as the CNN require very large volumes of datasets for training. So, we needed to augment the available time domain database. This task is done in this thesis work by creating the noisy versions of the database at various SNR levels just by synthetically adding random Gaussian noise to original noise-free data as discussed in the subsection 3.2.2. The resulting augmented dataset is then divided into training, validation and test subsets as described in section 3.2.3.

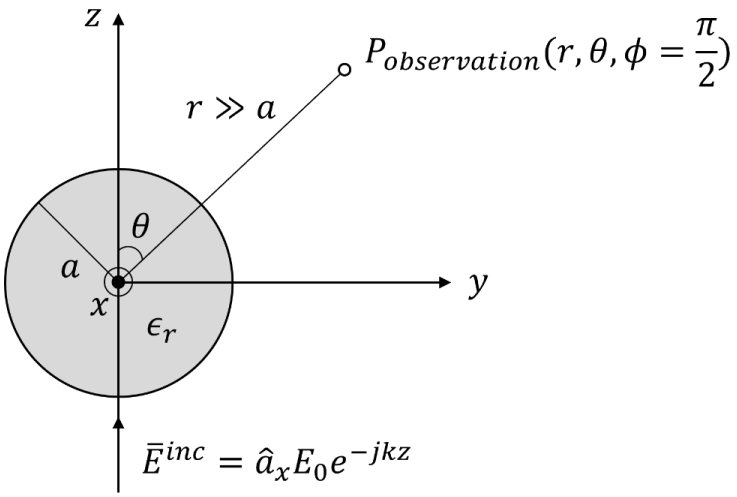


Figure 3.1. Problem geometry used to generate electromagnetic signals scattered from the spherical dielectric targets

3.2.1 Simulation of Scattered Database for Dielectric Spheres

The scattered electromagnetic signals of dielectric spheres are generated for the problem geometry given in Figure 3.1. The excitation plane wave is linearly polarized in x direction and propagates in z direction. As seen in Figure 3.1, the scattered fields are calculated at the $\phi = \frac{\pi}{2}$ plane for different θ angles. Due to the perfect spherical symmetry, the scattered responses had to be computed at bi-static aspect angles. Otherwise, we would obtain the same monostatic response in each angle. In [10], thirteen different θ angles are chosen as 5°, 15°, 30°, 45°, 60°, 75°, 90°, 105°, 120°, 135°, 150°, 165°, and 179° for the WD-PCA based classifier design. We also use the same aspect angles to form our dataset for the CNN based classifier design here.

The frequency domain scattered responses are computed over the ultra-wide bandwidth from 37 MHz to 19.1 GHz at 512 equally spaced frequency points. An example of frequency response is illustrated in Figure 3.2 for the dielectric sphere with permittivity 3 at the aspect angle 165°. In order to obtain the time domain scattering signal, we firstly apply Gaussian windowing to the frequency response so

that the effect of Gibbs' phenomenon is decreased. Afterwards, the Inverse Fast Fourier Transform (IFFT) is employed with 1024 points. A time domain scattered signal with a total observation duration of 26.81 ns is obtained at the end of this process as seen in Figure 3.3. As a result, a total of 52 different scattering signals are obtained for four different dielectric spheres at 13 bi-static aspect angles.

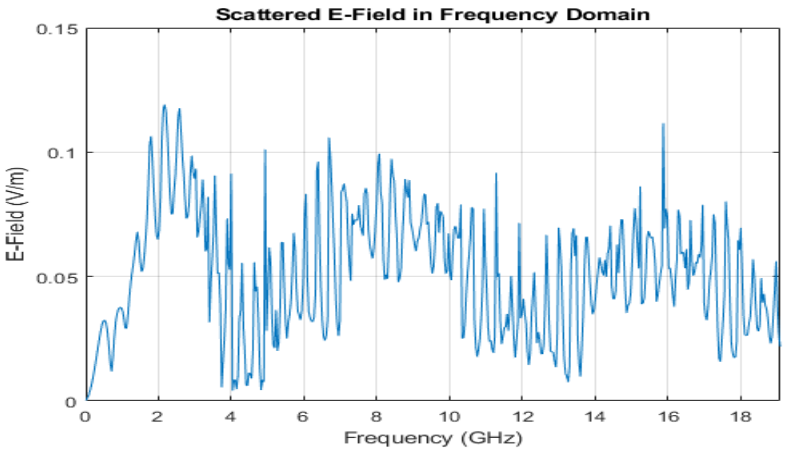


Figure 3.2. The magnitude of the far-field frequency response of the dielectric sphere with $\epsilon_r = 3$ where the bistatic angle is 165°.

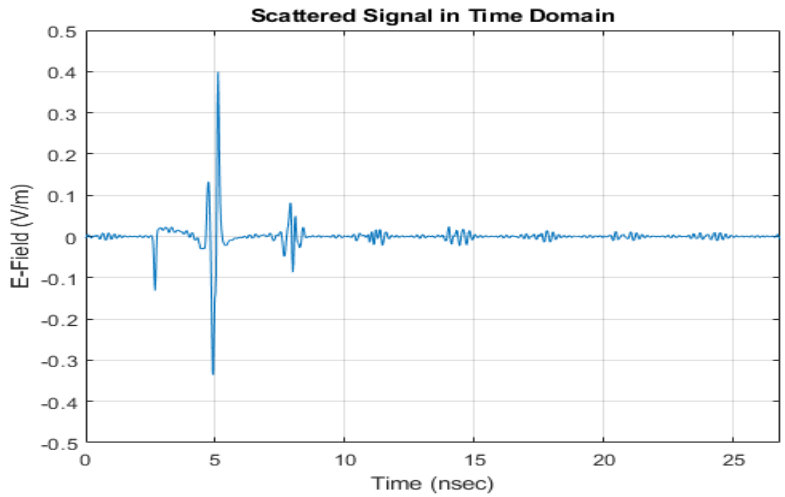


Figure 3.3. The far-field time domain scattered signal of dielectric sphere with $\epsilon_r = 3$ where the bistatic angle is 165°.

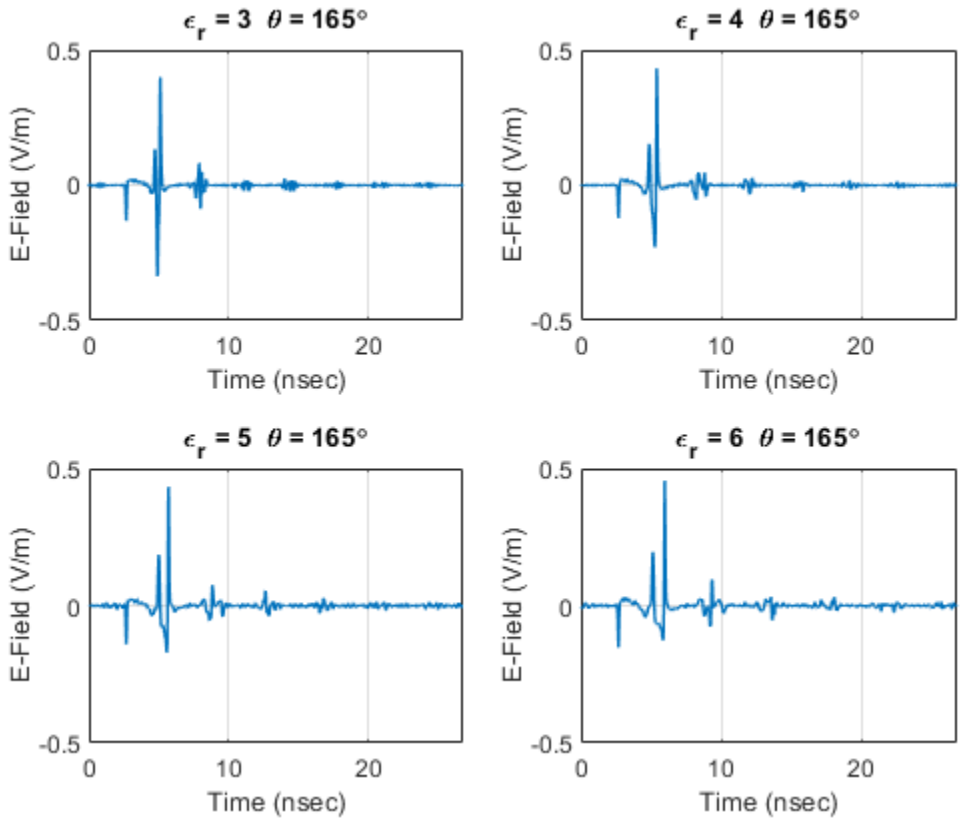


Figure 3.4. The simulated far-field time domain scattered signals of four different dielectric spheres with $\epsilon_r = 3, \epsilon_r = 4, \epsilon_r = 5$ *and* $\epsilon_r = 6$ where the bistatic angle is 165°.

Time-domain scattered signals of four dielectric spheres at the same aspect angle of 165° are illustrated in Figure 3.4 to examine the small variations in the signals. As the targets are very similar to each other, their transient responses are also very similar, which complicates the classification problem.

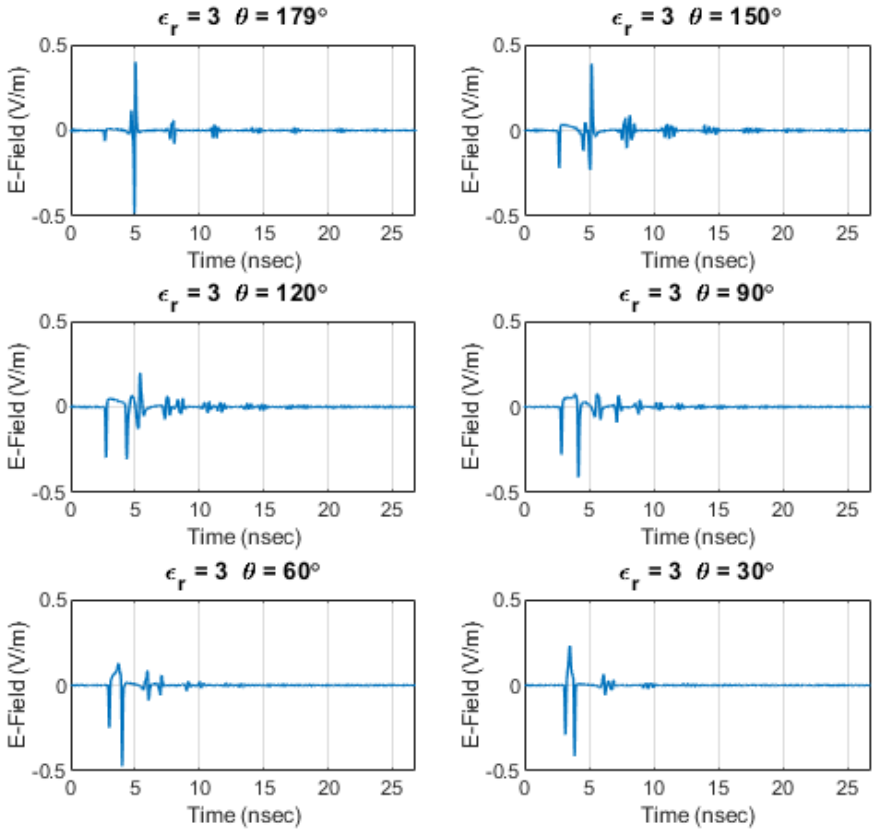


Figure 3.5. Simulated far-field time domain scattered signals of the dielectric sphere with $\epsilon_r = 3$ at the bistatic angles 179° (almost back-scattered aspect), 150°, 120°, 90°, 60°, 30°.

In addition to the arbitrarily chosen aspect angle of 165°, the scattered signals at aspect angles of 179°, 150°, 120°, 90°, 60° and 30° are also presented for the spherical target with $\epsilon_r = 3$ in Figure 3.5. Similar database signals for the other spherical targets are plotted in Appendix B. The extensive changes in early-time responses could be observed in these figures as the aspect angles vary. Therefore, it is a difficult task to discriminate targets only using the early-time response. On the other hand, late-time response contains the effects complex natural resonance frequencies, which do not depend on the aspect angle. So, designing a classifier with minimal sensitivity on aspect variation becomes possible by utilizing the late-time responses.

3.2.2 Data Augmentation

Thirteen different bistatic aspect angles are chosen to compose our dataset for dielectric spheres. Turhan-Sayan [10-11] used only five of them as reference for the WD-PCA based classifier design and the remaining eight aspect angles are used to test the performance of the classifier. However, a large dataset is required to develop a CNN based classifier. Hence, we need to augment the analytically generated dataset. For image classification purposes, a wide range of methods could be utilized to augment the original dataset [38]. For example, flipping, cropping, rotating and noise injection are some common image augmentation techniques. On the other hand, we could not take advantage of the vast majority of these techniques for 1D electromagnetic signals. This is because the important features of the scattered signals could be disrupted. For instance, temporal information of the signal could not be maintained when flipping is applied.

Han et al. apply three different augmentation methods for FMCW radar signals in [13], which are frequency shifting, noise addition and time stretching. Moreover, Selver et al. state in [21], adding Gaussian noise is an effective technique to augment ultra-wideband scattering signal database. When we examine these methods, frequency shifting and time stretching are seen not suitable for our problem since the target specific complex natural resonance frequencies could not be maintained. So, we decide to add Gaussian noise at three SNR levels of 15 dB, 17 dB and 20 dB to the generated noise-free signals for augmentation. In Figure 3.6, an example of a scattered signal with 17 dB SNR is illustrated for the dielectric sphere with $\epsilon_r = 3$ at the aspect angle 165°. In addition, the samples of noisy scattered signals at each SNR levels are shown in Figure 3.7. It can be observed from these figures that noiseless signal and the signal with 20 dB are very similar to each other because the noise level of the latter signal is quite low. Therefore, we do not add the noiseless signals in the dataset.

The analytically generated dataset is augmented with 30 different trials of randomly and independently generated Gaussian noise at three different SNR levels of 15 dB,

17 dB and 20 dB for each aspect angle. By this way, a total of 1170 scattered signals are obtained for each sphere at thirteen bi-static aspect angles, and the dataset for all dielectric spheres contains 4680 ultra-wideband scattered signals as stated in Table 3.2.

Table 3.2 Details of Dielectric Sphere Dataset

Bistatic Angles	*SNR Levels*	*For Single Target*	*For All Targets*
5°, 15°, 30°, 45°, 60°, 75°,	15 dB		
90°, 105°, 120°, 135°, 150°,	17 dB	1170	4680
165°, 179°	20 dB		

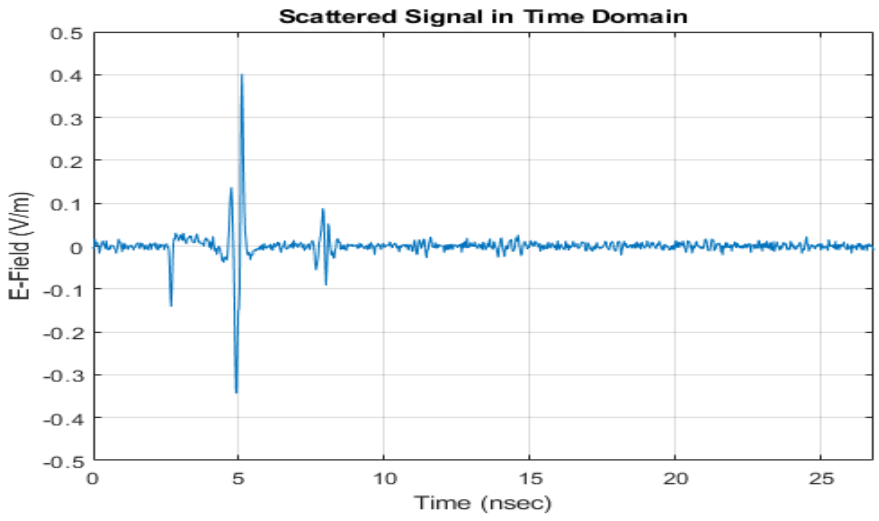


Figure 3.6. An example of a noise-added scattered signal for the dielectric sphere with $\epsilon_r = 3$ where the bistatic angle is 165° and SNR level 17 dB.

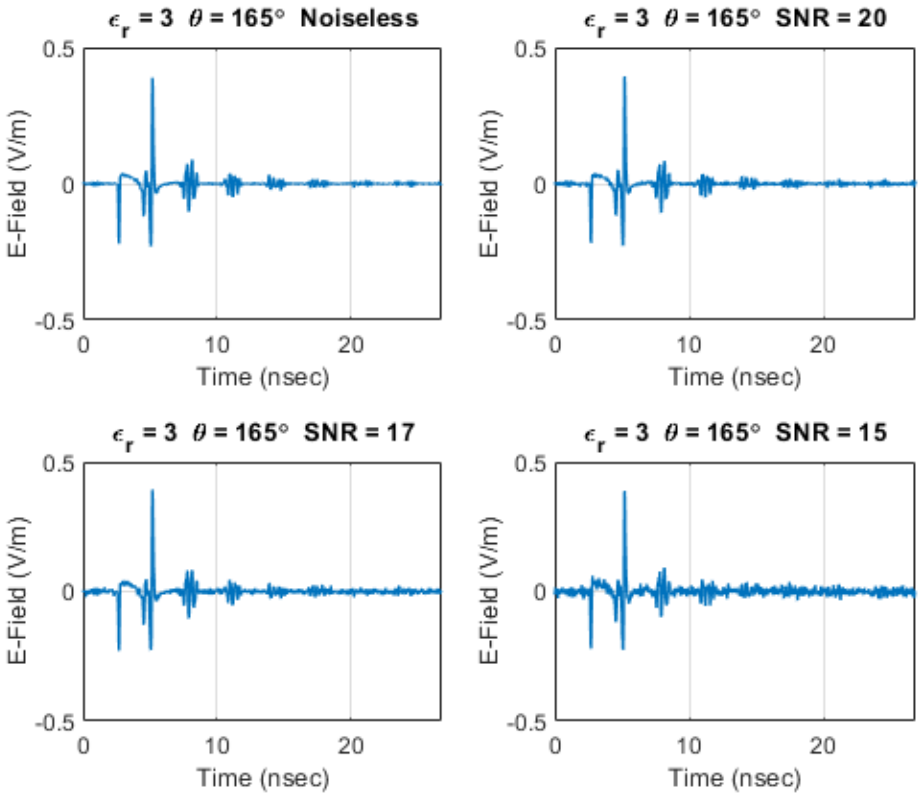


Figure 3.7. Noisy scattered signals with three different SNR levels of 20 dB, 17 dB and 15 dB for the sphere with $\epsilon_r = 3$ at the bistatic angle of 165°.

3.2.3 Separation of Training, Validation and Test Sets

The development process of a neural network simply requires the dataset should be divided into sufficiently large training and test sets. However, our dataset of dielectric spheres is of very limited size. In this thesis, for CNN-based classifier design, we utilize nested cross validation to obtain an unbiased model and observe the generalization performance [29]. Therefore, the generated dataset is separated into training, validation, and test sets to implement nested cross validation. As seen in Figure 3.8, nested cross validation is composed of an outer loop and an inner loop. The test samples are set apart in the outer folds to evaluate the model performance. In the inner folds, the model is fitted with the training set, and the validation set is used for investigation of the learning procedure and parameter selection.

In our original dataset, the distinct information is contained in the 13 aspect angles. Therefore, we split the data into 13 groups to perform nested cross validation. In each outer fold, one group is separated as the test set. For the inner folds, the remaining samples belong to 12 aspect angles are grouped into six where every group contains two different aspect angles. From these six groups, one of them is established as the validation set of that fold. With this separation, the number of samples in each subset is summarized in Table 3.3. As a result, 6-folds are carried out in every inner loop along with 13 outer folds. In total, we are able to fit our model with 78 different dataset separations to examine the effects of hyper-parameters and observe the generalization performance of the network.

Table 3.3 Number of Samples in Each Set for Nested Cross Validation

Training Set	Validation Set	Test Set	Total
3600	720	360	4680

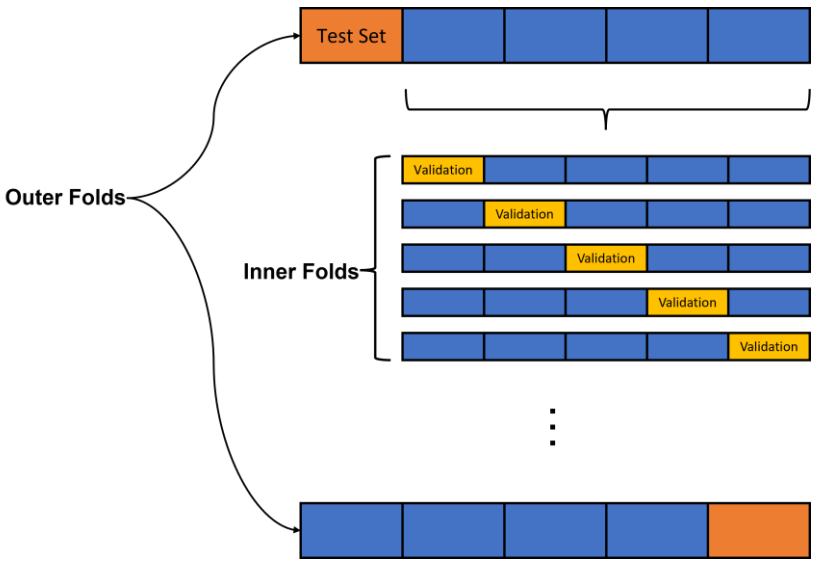


Figure 3.8. Data Separation for Nested Cross Validation

3.3 Preprocessing of the Input Signal for CNN

To improve the training process of the neural network, the input signal should be preprocessed with a couple of simple operations. The first preprocessing operation is normalization where the distribution of the input signal is converted into zero mean and unit variance. Secondly, the dataset is shuffled because the scattered signals are generated in a row. For a successful training process, the data should be given in a mixed order to the neural network.

3.4 CNN Architecture of the Dielectric Sphere Classifier

The number of the training samples directly affects the size of a neural network model because of the data-driven nature of neural networks. As the size of the dataset grows, we could train deeper networks. Since our dataset of dielectric spheres has a limited number of samples, we choose to work with a shallow model. The network to classify dielectric spheres is chosen to be composed of mainly three convolution layers as illustrated in Figure 3.9. The activation function of these three convolution layers is selected as rectified linear unit (ReLU) function. Zero padding is also applied so the output of a convolutional layer has the same shape as the input.

Filters with trainable coefficients forms a convolution layer, and a convolution layer basically performs convolution operation between the input signal and each trainable filter. By the help of these filters, the target specific features could be extracted in the network. So, we do not need to extract hand-crafted features but the features of the targets are learned from the training data samples.

Weight regularization methods are included in the network besides the convolution layers. With the help of regularization, the network can avoid overfitting, and the statistics of the given dataset can be learned successfully. The regularization methods are only executed during training stage and not included after training is finished. As a common choice in the field of neural networks [23], two techniques of regularization which are L2 regularization and dropout (with the rate of 0.5) are used

together in the network. At the early steps of the development process, it is clearly observed that overfitting occurs immediately if any regularization method is not used. In fact, the limited number of scattered signals could be considered as the main reason of that situation. So, we also apply batch normalization after each convolutional layer to avoid overfitting. Although batch normalization has a regularization effect, the primary advantage of this method is to accelerate the training procedure and produce a more robust neural network [30]. We take advantage of batch normalization and achieve satisfying results with small number of epochs like 20.

In CNNs, pooling layers are placed between convolution layers to reduce the number of parameters and the computation cost. Pooling layers can achieve this effect by down-sampling the output of the previous convolutional layer. In our architecture, we choose to use max pooling with the pooling size 2. This means that the size of the signal is reduced to its half in every pooling layer by taking the maximum value of two consecutive time samples of the signal.

After three convolutional layers, a fully connected layer with 64 hidden units takes place where the classification task is performed. Fully connected layers are indeed similar to regular artificial neural networks. Each neuron in the hidden layer is connected to previous and following layers. Lastly, a softmax layer follows the fully connected layer as the output layer. Since we have four different dielectric spheres as targets, the softmax layer contains four units. At the output of softmax layer, we obtain the classification probabilities for each target. The output values computed by softmax function are located between zero and one, and the sum of all values are equal to one. An ultra-wideband scattered signal is then classified as belonging to one of the four library targets according to the highest probability value.

The CNN architecture described so far is implemented in Python with TensorFlow framework, and the code written for this purpose is given in Appendix F. The hardware used for training the CNN model for dielectric classification is NVIDIA Quadro K2200 GPU and Intel Xeon E5-1603 CPU.

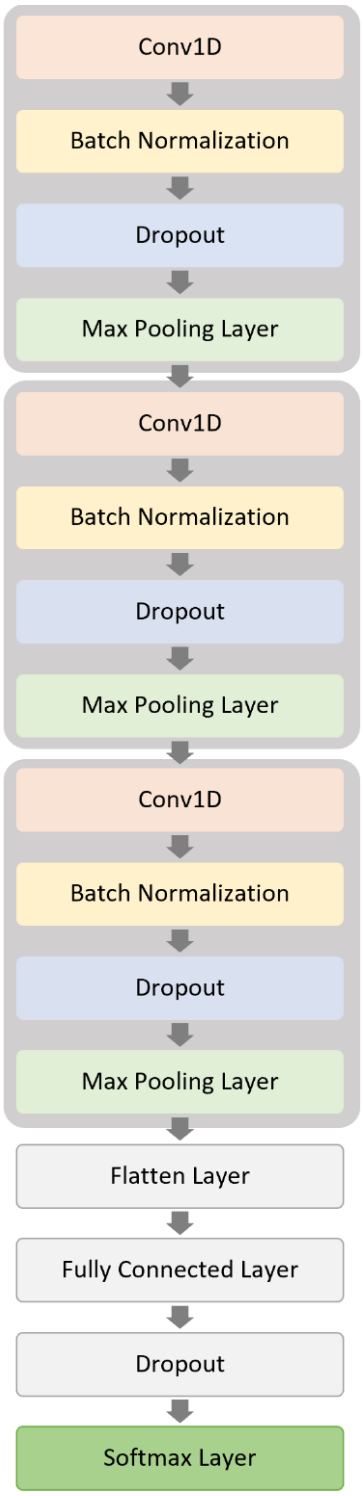


Figure 3.9. CNN architecture used for dielectric sphere classification

3.5 Hyper-parameter Optimization for Dielectric Sphere Classifier

The general architecture of the CNN developed for dielectric sphere classification is described in the previous section. To obtain an optimum deep learning model, we search different hyper-parameters which are the kernel size of convolutional layers, learning rate and the mini-batch size. The different hyper-parameters are tested by applying grid search. The grid search experiments are listed in Table 3.4.

In the convolutional layers, we use 16, 32 and 64 filters respectively. To simplify the parameter selection procedure, we prefer to use the same number of kernels in each three convolutional layers. The values of 3, 5 and 7 are examined to determine the optimum kernel size.

During the training stage of the deep learning model, the optimizer is chosen as Adam algorithm which is a computationally efficient form of stochastic gradient decent. The common choice of the learning rate used with Adam algorithm is 0.001. Hence, we start with this value in our experiments. Since it is observed in the first trials that overfitting could occur due to high learning rates, we examine the lower values instead. So, 0.001, 0.0005 and 0.0003 values are picked in our grid search for learning rate.

Training of a neural network occurs in an iterative manner. All samples in the training set could be used several times to update weights of the network. An epoch represents the iterations when all the training samples have been seen once by the network. However, an epoch could be divided into mini batches of training set. The choice of the mini-batch size directly affects training procedure because the loss function is computed in every iteration with the selected number of samples in the batch. For the sake of more efficient computations, mini-batch size is selected as the powers of 2. Therefore, we pick the values 32, 64 and 128 during our grid search.

Table 3.4 List of Hyper-parameter Optimization Experiments for Sphere Classification

Experiment No	Kernel Size	Learning Rate	Mini-Batch Size	Accuracy
Experiment 1	(3, 3, 3)	0.001	32	0.8077
Experiment 2	(3, 3, 3)	0.001	64	0.7962
Experiment 3	(3, 3, 3)	0.001	128	0.7695
Experiment 4	(3, 3, 3)	0.0005	32	0.8049
Experiment 5	(3, 3, 3)	0.0005	64	0.8027
Experiment 6	(3, 3, 3)	0.0005	128	0.7991
Experiment 7	(3, 3, 3)	0.0003	32	0.7954
Experiment 8	(3, 3, 3)	0.0003	64	0.8014
Experiment 9	(3, 3, 3)	0.0003	128	0.7996
Experiment 10	(5, 5, 5)	0.001	32	0.8143
Experiment 11	(5, 5, 5)	0.001	64	0.7957
Experiment 12	(5, 5, 5)	0.001	128	0.7728
Experiment 13	(5, 5, 5)	0.0005	32	0.8121
Experiment 14	(5, 5, 5)	0.0005	64	0.7872
Experiment 15	(5, 5, 5)	0.0005	128	0.8140
Experiment 16	(5, 5, 5)	0.0003	32	0.8006
Experiment 17	(5, 5, 5)	0.0003	64	0.8021
Experiment 18	(5, 5, 5)	0.0003	128	0.8147
Experiment 19	(7, 7, 7)	0.001	32	0.8161
Experiment 20	(7, 7, 7)	0.001	64	0.8132
Experiment 21	(7, 7, 7)	0.001	128	0.8034
Experiment 22	(7, 7, 7)	0.0005	32	0.7948
Experiment 23	(7, 7, 7)	0.0005	64	0.8146
Experiment 24	(7, 7, 7)	0.0005	128	0.7884
Experiment 25	(7, 7, 7)	0.0003	32	0.8106
Experiment 26	(7, 7, 7)	0.0003	64	0.7806
Experiment 27	(7, 7, 7)	0.0003	128	0.7965

The hyper-parameter experiments are conducted with nested cross validation as described in 3.2.3. For evaluation, accuracy, precision, and recall are selected as the performance metrics. The definitions of the metrics are stated in Appendix C. The resulting metrics of cross-validation are averaged and illustrated in Figure 3.10. As seen in the Table 3.4 and Figure 3.10, we obtain the best result from Experiment 19 with 81.6% accuracy. In this configuration, the kernel size is 7, learning rate is 0.001 and mini-batch size is 32. The individual effect of each parameter on performance of the model is discussed in the following sections.

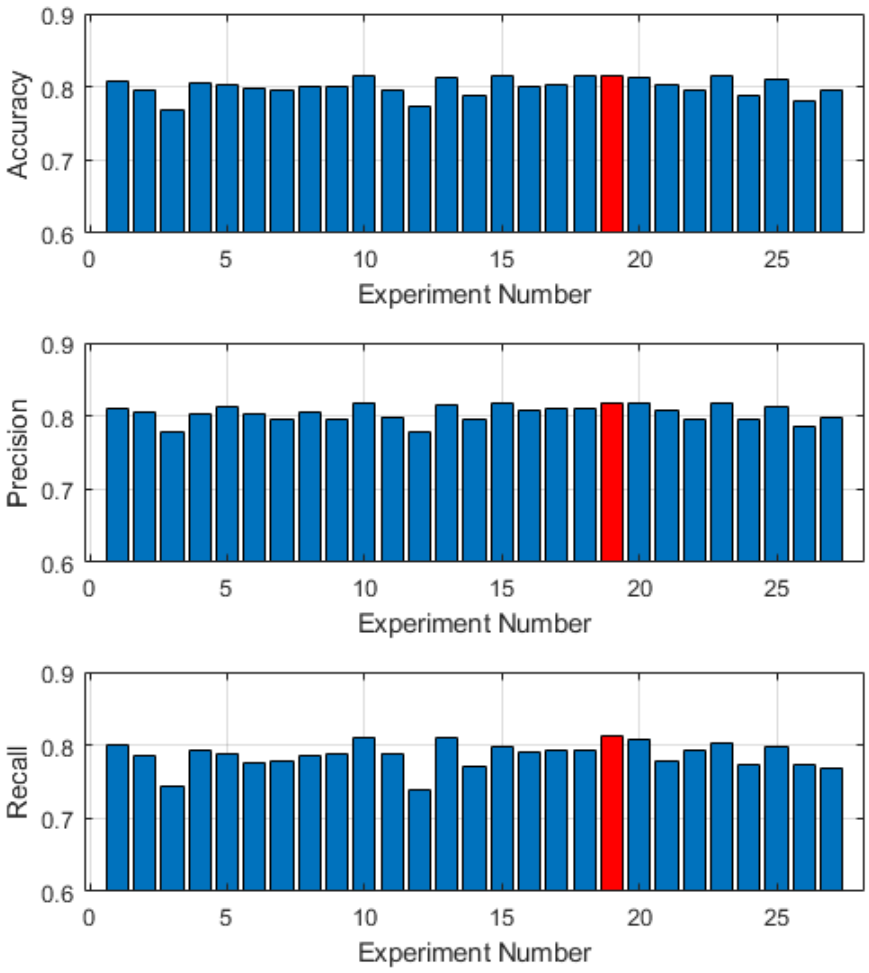


Figure 3.10. The results of hyper-parameter optimization experiments

3.5.1 Impact of the Kernel Size

The individual scores of different hyper-parameters are computed by averaging the results of the grid search while keeping the corresponding hyper-parameter constant. The obtained scores for kernel size are shown in Figure 3.11. It is observed that the best score is achieved with the kernel size 7. The result is consistent with the outcome of Experiment 19. As seen in Figure 3.11, the performance metrics increase while the kernel size grows. It could be said that the features representing the damped sinusoids due to the natural resonance mechanism are learned better with larger kernel size.

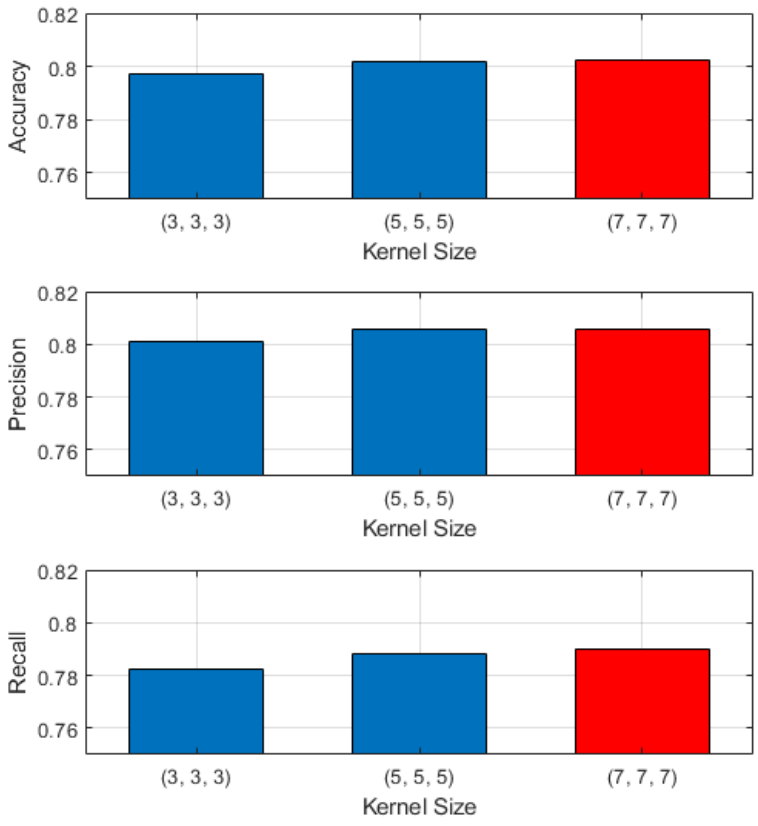


Figure 3.11. Impact of different kernel sizes on the performance of the classifier

3.5.2 Impact of the Learning Rate

When we examine the effect of learning rate on the model performance, it is observed that changes in the learning rate have a slight influence on the network. Although the scores for the three different values are almost the same, the highest score is obtained with the learning rate 0.0005. However, we receive the optimum result with the learning rate 0.001 in the grid search. The difference could be ignored since the learning rate has a small effect on the network performance as seen in Figure 3.12.

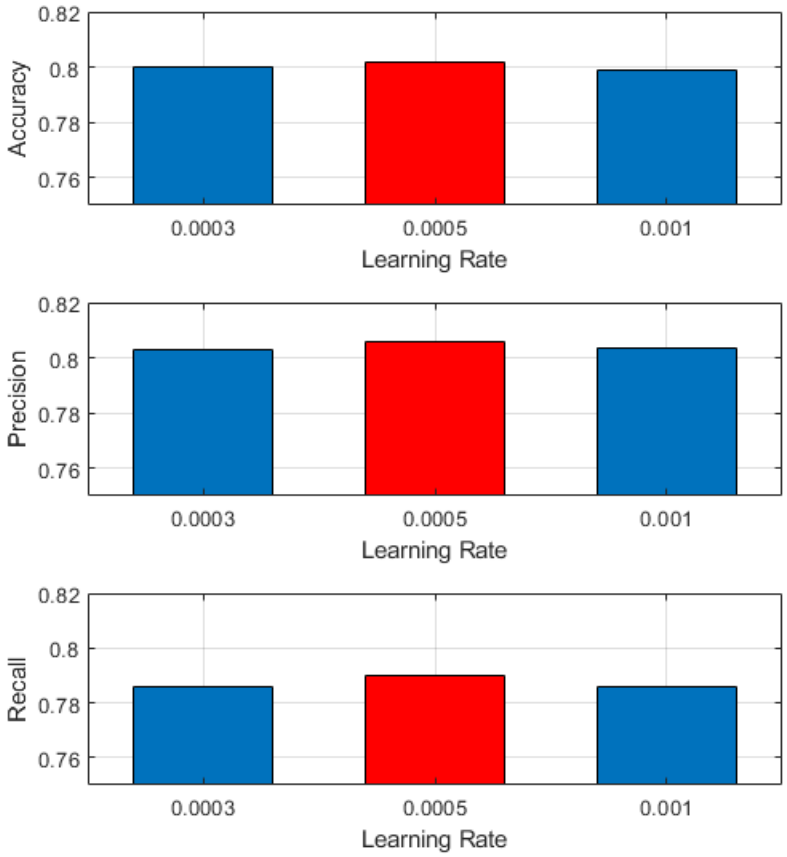


Figure 3.12. Impact of different learning rates on the performance of the classifier

3.5.3 Impact of the Mini-batch Size

Among the three different hyper-parameters, the mini-batch size has the most dramatic effect on the network. As seen in Figure 3.13, the performance metrics of our network are decreasing while the mini-batch size increase from 32 to 128. We obtain the best score with the mini-batch size of 32. This situation is also seen in the result of the overall grid search where Experiment 19 gives the highest score.

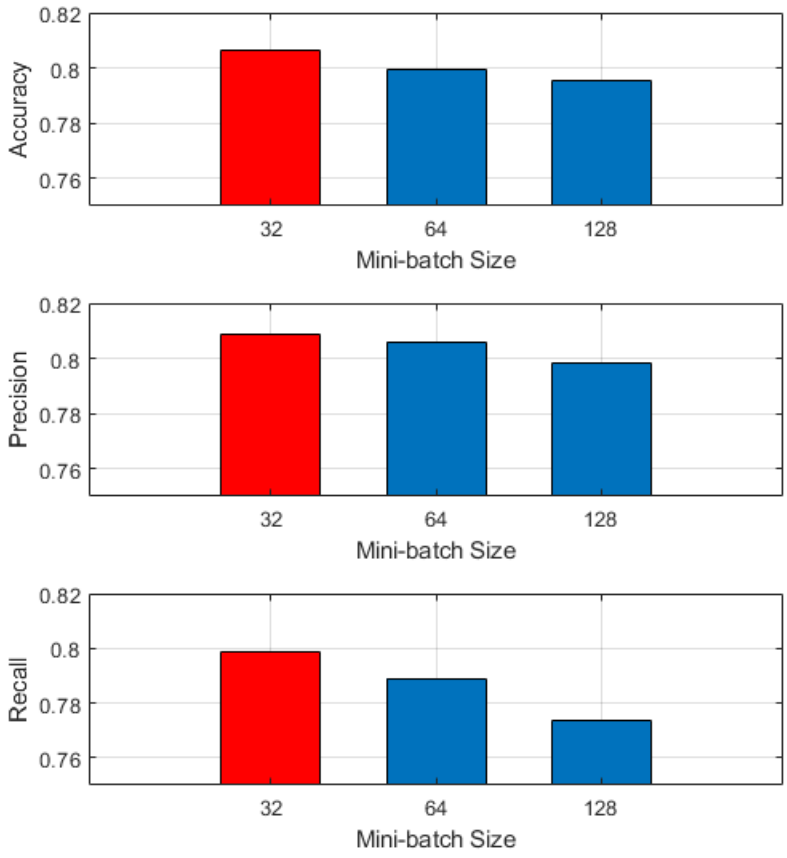


Figure 3.13. Impact of the mini-batch size on the performance of the classifier

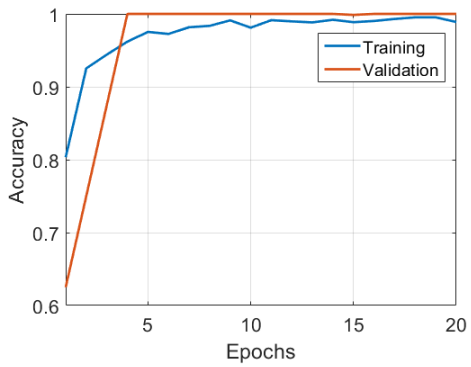
3.6 Training and Evaluation of the Optimized Network

The hyper-parameter optimization is carried out in the previous section for the defined CNN architecture in section 3.4. We conclude that the classifier performs better with the increasing number of kernel size. In addition to kernel size, it is seen that the mini-batch size has a high influence on the classification performance. For dielectric sphere classifier, using a small value of mini-batch size works better. On the other hand, the learning rate is seen not an effective hyper-parameter of the proposed network.

The optimized deep learning model with the hyper-parameters used in Experiment 19 is summarized in Table 3.5. The total number of parameters of the network is 543204. We train the optimized network with the dataset of the outer fold 3 through 20 epochs. In the training process, the learning curves are obtained as in Figure 3.14. From the loss curve, we could say that the model is trained in a good manner. This is because the validation loss follows the training loss which decays exponentially. Besides, the accuracy curve shows that overfitting is not occurred. If overfitting was occurred, there would be a big gap between the training accuracy and the validation accuracy. Also, the validation loss would not follow the training loss, and diverge from it. As a result of this training process which essentially use the complete dataset for the sphere targets at all 13 aspects due to the use of cross-validation approach, we obtain the confusion matrix illustrated in Figure 3.15 reported for only the bi-static aspect angle of 30 degrees, as an example. The achieved test accuracy for this sample case is 97.8%.

Table 3.5 Model Summary of Optimized Network for Sphere Classification

Layers	**Output Shape**
Input Layer	1024
Conv 1D, ReLU, L2 Regularization	(1024, 16)
Batch Normalization	(1024, 16)
Dropout	(1024, 16)
Max Pooling 1D	(512, 16)
Conv 1D, ReLU, L2 Regularization	(512, 32)
Batch Normalization	(512, 32)
Dropout	(512, 32)
Max Pooling 1D	(256, 32)
Conv 1D, ReLU, L2 Regularization	(256, 64)
Batch Normalization	(256, 64)
Dropout	(256, 64)
Max Pooling 1D	(128, 64)
Flatten	8192
Fully Connected Layer, L2 Regularization	64
Dropout	64
Softmax Layer	4



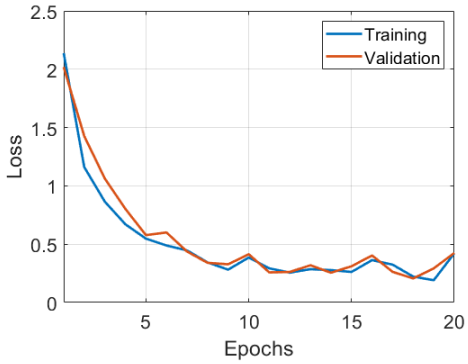


Figure 3.14. The learning curves of the optimized network for dielectric sphere classification

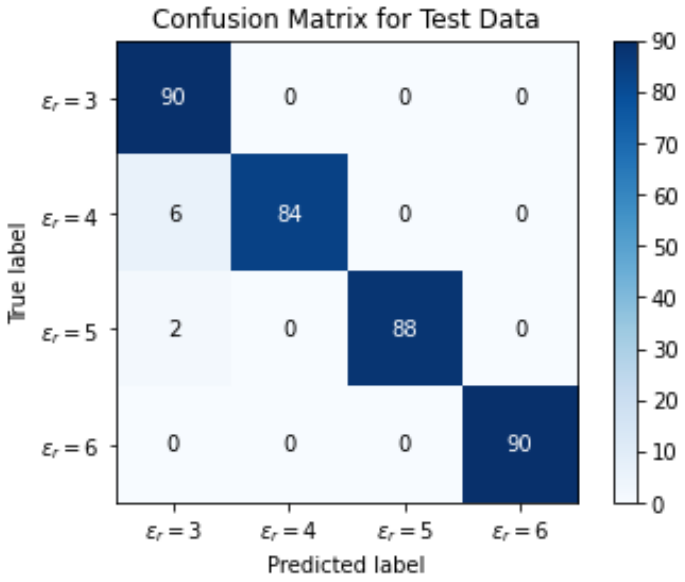


Figure 3.15. Confusion matrix of the optimized network for dielectric sphere classification considering only the test data at 30 degrees bi-static aspect angle.

3.7 Training the Optimized Network with a Smaller Training Set

As described in section 3.2.3, the available dataset for the spherical targets is split into training, validation and test set and the nested cross validation approach is utilized to examine the generalization performance of the CNN network for dielectric sphere classification by exposing the network to all the available data for four targets, 13 aspect angles, three SNR levels. One of our aims in Chapter 3 is to verify the performance of the CNN-based classifier design method against the performance of the WD-PCA based classifier design method for the target library of dielectric spheres over the same database. For this reason, we will separate the sphere dataset into training and test sets for the next CNN-based sphere classifier design in such a way that is compatible with the training/testing data set separation used in [11]. It

should be also mentioned that the same set of hyper-parameters optimized in the previous CNN-based classifier design (i.e. kernel size of 7, mini batch size of 32 and learning rate of 0.001) will be maintained in the classifier design to be reported next.

3.7.1 Separation of Training and Test Sets

In this case, the training dataset is composed of the scattered signals simulated at five different aspect angles while the rest of the dataset at eight more aspect angles belongs to the test dataset. The bi-static aspect angles of the training set are 5°, 45°, 90°, 135°, 179° while the aspect angles of the test dataset are 15°, 30°, 60°, 75°, 105°, 120°, 150°, 165° as selected in the WD-PCA classifier design approach in Turhan-Sayan's work in [11]. As a result, for each dielectric sphere, the training dataset contains 450 samples while the testing dataset contains 720 samples as shown in Table 3.6.

Table 3.6 Training and Test Sets compatible with [11]

Dataset	*Bistatic Angles*	*SNR Levels*	*For Single Target*	*For All Targets*
Train	5°, 45°, 90°, 135°, 179°	15 dB 17 dB 20 dB	450	1800
Test	15°, 30°, 60°, 75°, 105°, 120°, 150°, 165°		720	2880

3.7.2 Training and Evaluation of the Network

In this section, the previously optimized network stated in Table 3.5 is trained by also using the previously optimized hyper-parameters but making use of an aspect-

wise restricted dataset this time as described in section 3.7.1, instead of using the whole available dataset.

As the network is trained with the mentioned dataset, we observe a variation in the performance of the classifier at different training repetitions. The accuracy of the classifier varies between 78.5% and 99.4% when we train the network from scratch 5 times. 89.9% mean accuracy is acquired for the 5 repetitions. Here, in Figure 3.16 and Figure 3.17, we report the results of the classifier with the best performance of 99.4% accuracy out of these five design trials. During training of the network, we obtain the learning curves shown in Figure 3.16. Even though a smaller training set is used, no overfitting is observed. As seen in the figure, the validation loss follows the training loss with a small bias. Moreover, the validation accuracy reaches 90% after epoch 5. As a result, the confusion matrix illustrated in Figure 3.17 is obtained. We achieve 99.4% test accuracy along with 99.6% precision and 98.2% recall for dielectric sphere classification.

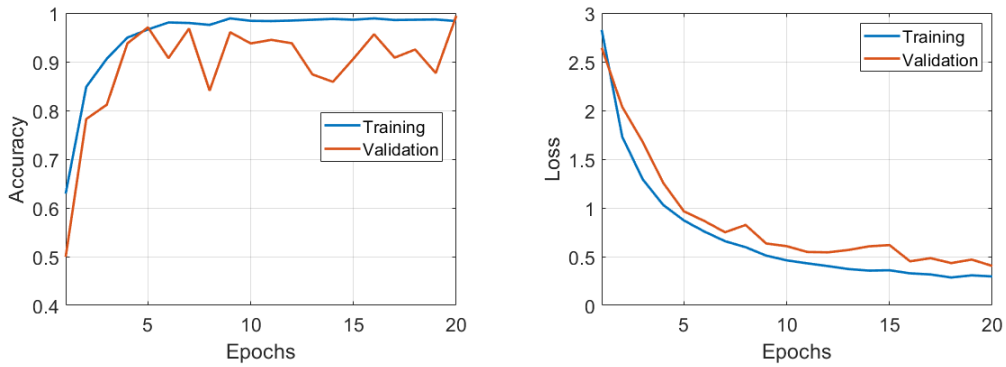


Figure 3.16. The learning curves of the best optimized network (out of five trials) when the dataset described in section 3.7.1 is used.

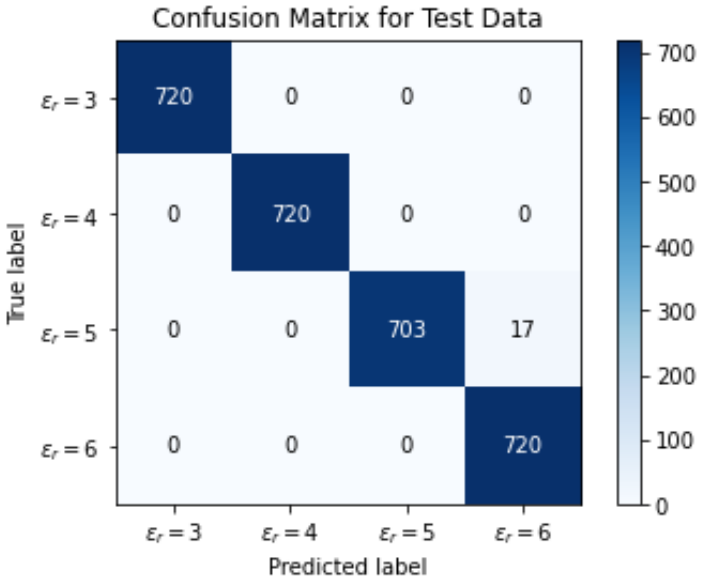


Figure 3.17. Confusion matrix of the best optimized network (out of five trials) when the dataset described in section 3.7.1 is used for training.

3.7.3 Noise Performance of the CNN-based Classifier for Dielectric Spheres

The deep learning model is developed with the training and test sets which both include scattered signals with the SNR levels of 15 dB, 17dB, and 20 dB. To observe the performance of the CNN-based classifier under noise and to make a comparison with the WD-PCA method, a new testing dataset is generated. Bi-static aspect angles are kept the same, but SNR levels differ in the new test dataset. Five different noisy datasets are generated with the SNR levels 0 dB, 5 dB, 10 dB, 15 dB, and 20 dB. For the signals with 20 dB SNR, the accuracy of the classifier is observed to be 100% as seen in

Table 3.7. As SNR level of the signal decreases, the accuracy of the classifier decreases to 98.7%, 97.5% and 93.4% for 15 dB, 10 dB and 5 dB SNR, respectively, as expected. In the case of 0 dB SNR, the classifier produces 81.6% accuracy where

the classifier maintains its high performance under noise. The corresponding confusion matrices are given in Appendix D.

The testing accuracy levels of the WD-PCA based classifier were reported to be 100 %, 96 %, 91 % and 75 % at 20 dB, 15 dB, 10 dB and 5 dB SNR levels, respectively in [11] where the classifier was designed using only noise-free data. In previous research, it has been demonstrated that use of noisy training data having moderate noise levels in classifier design leads to notable improvements in the classification accuracy rates at lower SNR testing levels [33-34]. While both CNN-based and WD-PCA based classifiers reach 100% accuracy at 20 dB SNR level, the former classifier demonstrates better accuracy performance at lower SNR values probably due to this biasing effect as the noisy data at 20 dB, 17 dB and 15 dB are utilized in training the CNN classifier because of the need for dataset augmentation.

Table 3.7 Noise Performance of the CNN-based Classifier

SNR Level	Accuracy	Precision	Recall
0 dB	81.6%	86.5%	72.2%
5 dB	93.4%	95.4%	91.5%
10 dB	97.5%	97.8%	97.2%
15 dB	98.7%	99.1%	98.1%
20 dB	100%	100%	100%

3.8 Effect of Number of Aspect Angles in the Training Set

In the previous section, the network with optimized hyper-parameters is trained with a training set containing signals from 5 different aspect angles to compare the CNN-based and WD-PCA based classifiers. On the other hand, we would like to examine the effect of the number of the aspect angles in the training set in this section.

Although it is seen that a high performance is achieved in section 3.7, there is a large variance in the performance of the designed networks at different trials. In this section, a different set of hyper-parameters (the kernel size of 3, the learning rate of 0.0005, and the mini-batch size of 128) is utilized while investigating the effect of using training data over a wider selection of the aspect angle. The network with the mentioned hyper-parameters is trained with three different training sets including 5, 6 and 7 bi-static aspect angles, and the results are presented below.

3.8.1 Training Set Containing 5 Aspect Angles

We start with 5 aspect angles to examine the effect of training set size on the performance of the classifier. The scattered signals from the aspect angles 5°, 45°, 90°, 135° and 179° construct the training set. The remaining signals are used for testing. When we train the network from scratch 5 times, it is observed that the accuracy of the classifier changes between 90.4% and 97.4% with the mean accuracy of the 5 trials being 93.2%. Here, the classifier design procedure looks more stable as we observe a smaller performance variation as compared to the results of Section 3.7.2 where the accuracy rate was found to change between 78.5% and 99.4%. As an example among the 5 design trials of this section, the learning curves of the CNN classifier with the accuracy of 91.5% are illustrated in Figure 3.18. Also, the confusion matrix belonging to this specific classifier design is shown in Figure 3.19. For this classifier, the precision and the recall metrics are also obtained as 93.1% and 84.4%, respectively.

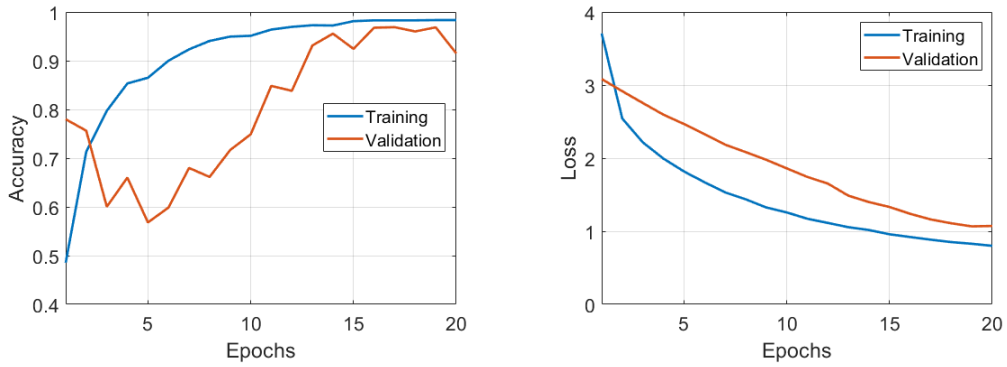


Figure 3.18. The learning curves of the sample CNN classifier when 5 aspect angles (5°, 45°, 90°, 135° and 179°) are used in the training dataset.

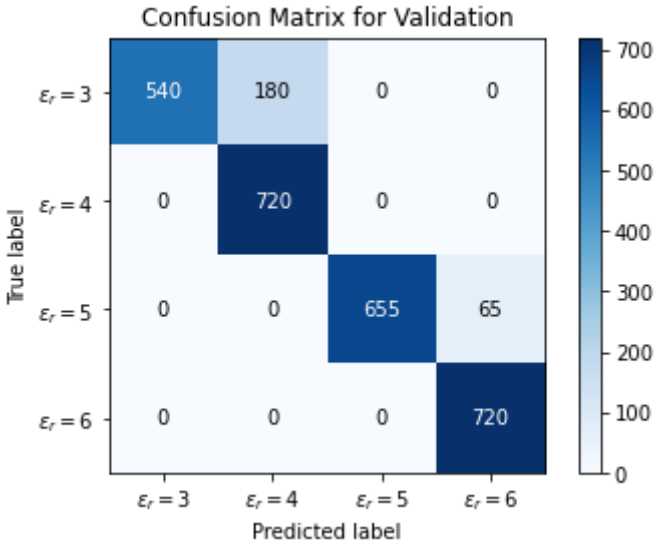


Figure 3.19. Confusion matrix of the sample CNN classifier when 5 aspect angles (5°, 45°, 90°, 135° and 179°) are used in the training dataset.

3.8.2 Training Set Containing 6 Aspect Angles

This time, the training dataset is composed of the scattered signals from the aspect angles 15°, 45°, 75°, 105°, 135°, and 165°. When we trained the CNN classifier at 5 different aspect angles, we achieved 93.2% average accuracy over 5 repetitions of training. Now, however, when the number of aspect angles are increased to 6, the mean accuracy over 5 trials becomes 98.4%. In different repetitions of training, the

accuracy of the classifier varies in a much narrower range between 96.5% and 100%, i.e. the training of the CNN classifier becomes more stable. In other words, we observe a performance improvement with the increased training information regarding the coverage of aspect angles. In Figure 3.20, the learning curves are plotted for the network with 96.5% accuracy; 99.7% precision and 91.2% recall are also achieved with this network. The confusion matrix of the network is shown in Figure 3.21. All signals from Target 1, Target 2, and Target 4 are recognized correctly. Only 87 signals from Target 3 are misclassified as Target 4.

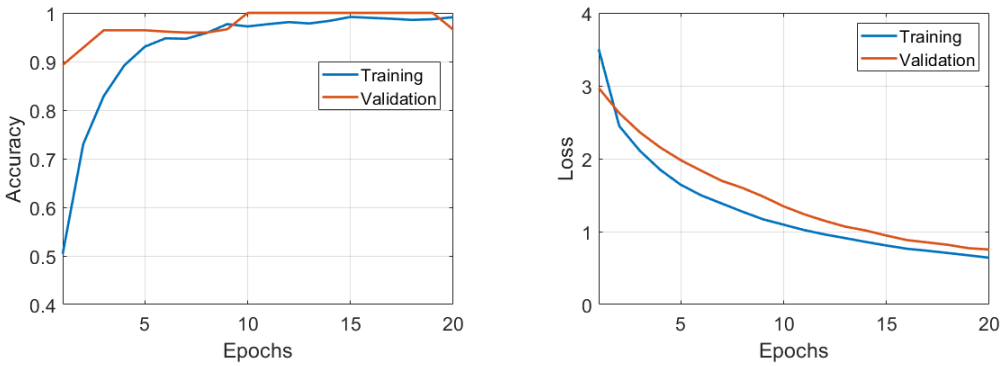


Figure 3.20. Learning curves of the sample CNN classifier when 6 aspect angles (15°, 45°, 75°, 105°, 135°, and 165°) are used in the training dataset.

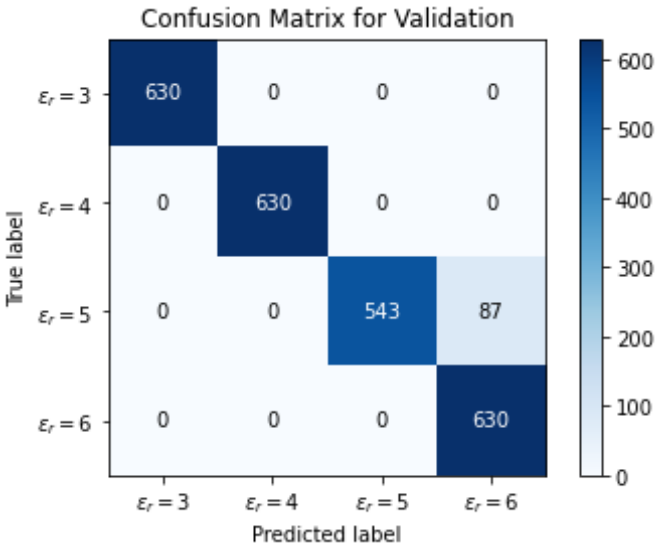


Figure 3.21. Confusion matrix of the sample CNN classifier when 6 aspect angles (15°, 45°, 75°, 105°, 135°, and 165°) are used in the training dataset.

3.8.3 Training Set Containing 7 Aspect Angles

It is observed that with the increasing number of aspect angles, the CNN-based dielectric sphere classifier performs better. Hence, we train the deep learning model with a training set containing 7 aspect angles which are 5°, 30°, 60°, 90°, 120°, 150°, and 179°. Again, the training procedure is repeated 5 times to see if there exists any fluctuation of performance in different trials. With the training set including 7 aspect angles, 100% accuracy is obtained at all trials. An example learning curve is shown in Figure 3.22, and all signals in the test subset are classified correctly as seen in Figure 3.23.

As a conclusion, the performance of the CNN-based dielectric sphere classifier is improved with the increasing number of aspect angles. This behavior could be explained as the network learns better with more information in the training dataset. When the mentioned 7 aspect angles are chosen for the training dataset, the performance metrics all reach to 100% and variance in the performance metrics is not observed in different trails of training procedure.

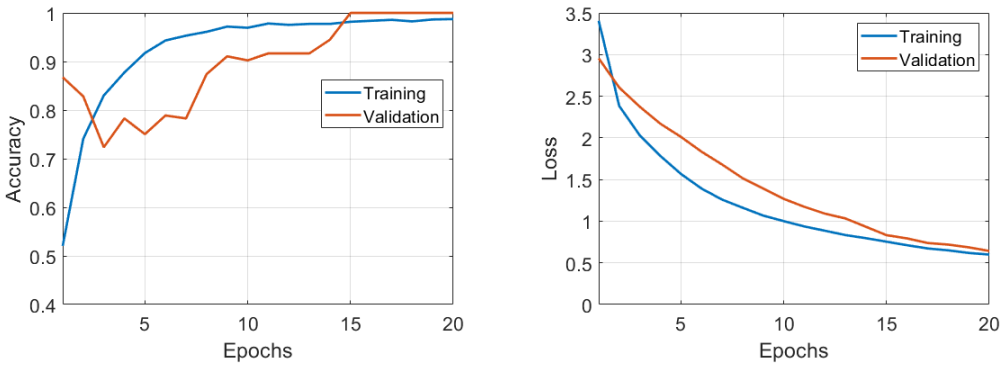


Figure 3.22. Learning curves of the sample CNN classifier when 7 aspect angles (5°, 30°, 60°, 90°, 120°, 150°, and 179°) are used in the training dataset.

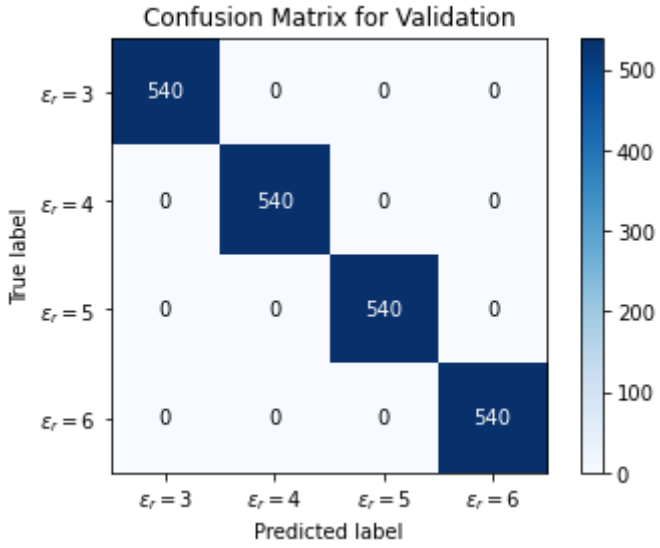


Figure 3.23. Confusion matrix of the sample CNN classifier when 7 aspect angles (5°, 30°, 60°, 90°, 120°, 150°, and 179°) are used in the training dataset.

CHAPTER 4

ELECTROMAGNETIC TARGET CLASSIFIER DESIGN FOR QUADCOPTER TYPE UNMANNED AERIAL VEHICLES BY CONVOLUTIONAL NEURAL NETWORKS

After validating the effectiveness of the CNN-based electromagnetic target classifier design approach in Chapter 3, a second and more challenging target classifier will be designed in this chapter for a set of four quadcopters having similar properties using their measured scattered field database over a relatively narrow bandwidth.

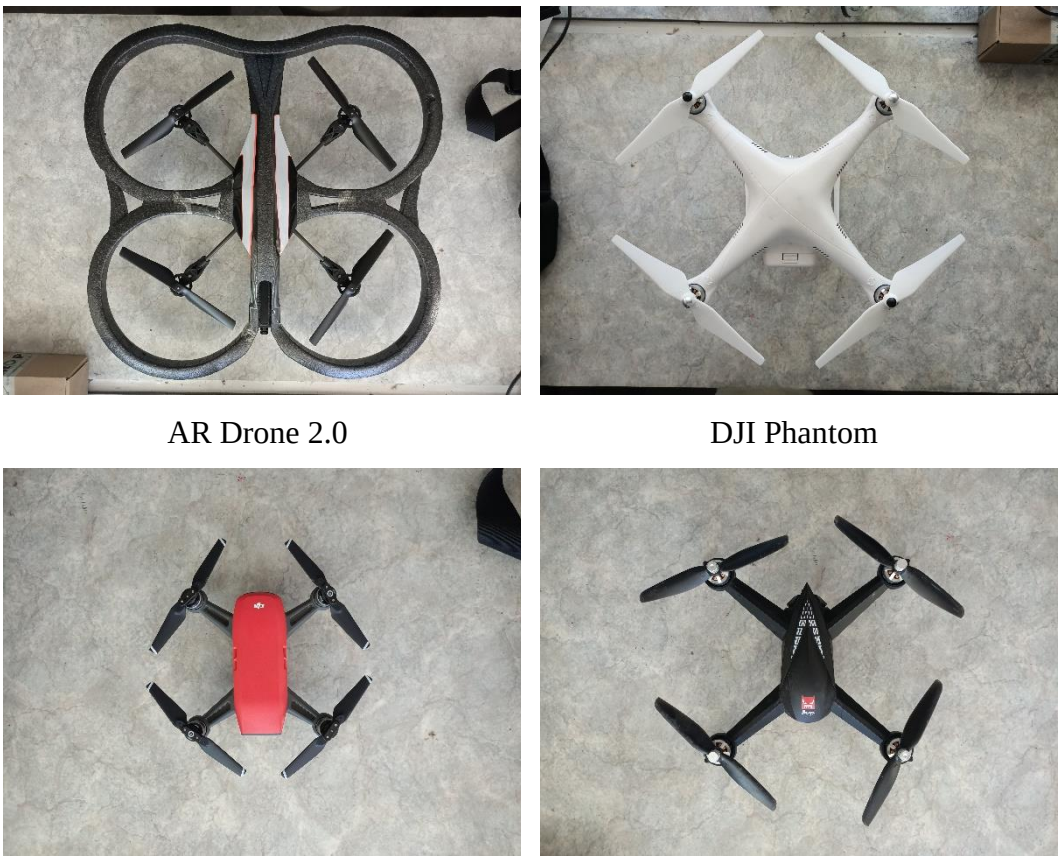
In the present chapter, we will design a CNN-based classifier for commercial quadcopter type UAVs using what we learned from dielectric sphere classification exercise. The target set is composed of four different models of quadcopters. The database of wideband far field scattered signals for these targets are constructed totally by measurements over a very modest bandwidth from 3.1 GHz to 4.8 GHz using a portable wideband radar kit as opposed to the simulated database of spheres with almost nine-octaves bandwidth. Keeping these significant differences between the databases of spheres and quadcopters in mind, the CNN-based classifier will be designed and its performance will be investigated in the rest of this chapter.

4.1 Properties of the Target Library

The target library for this classifier design includes four different quadcopter type commercial drones which are AR Drone 2.0, DJI Phantom, DJI Spark, and MJX Bugs 5W. The picture for each quadcopter is presented in Figure 4.1. As stated in Table 4.1, the targets could be divided into two main groups regarding their dimensions. AR Drone 2.0 and DJI Phantom belong to the “bigger targets” set while DJI Spark and MJX Bugs 5W belong to the “smaller targets” set. In Table 4.1, the diagonal dimension between two motors is given to show the largest physical length of each target.

Table 4.1 Physical properties of the commercial UAV targets

Target Type	*Diagonal Distance*	*Weight*
AR Drone 2.0	40 cm	420 g
DJI Phantom	35 cm	1200 g
DJI Spark	17 cm	300 g
MJX Bugs 5W	25 cm	400 g



AR Drone 2.0 DJI Phantom

DJI Spark MJX Bugs 5W

Figure 4.1. Commercial UAV target set composed of four quadcopters

4.2 Measurement Setup

The scattered field measurements of the quadcopters are collected with the wideband radar kit with brand name ***TimeDomain PulsON 440***. The radar module operates over the frequency band 3.1 to 4.8 GHz having transmitter/receiver antennas with omnidirectional pattern in the H-plane (azimuth plane). Due to this antenna pattern, measurements are strongly affected by environmental cluttering. Therefore, the scattered data had to be collected in an anechoic chamber as seen in Figure 4.2 to obtain almost clutter-free measurements. The characteristics of the excitation pulse generated by the module and details of the measurement setup are explained in Appendix E.

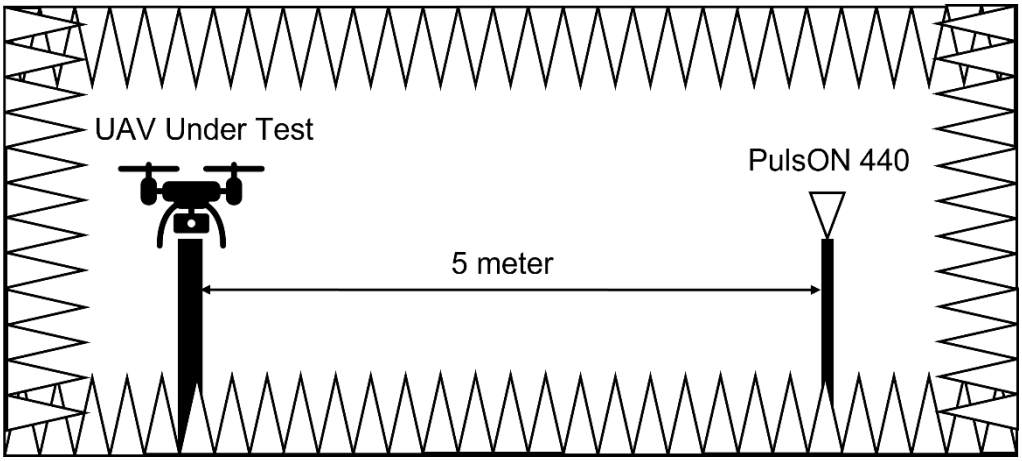


Figure 4.2. Measurement setup in anechoic chamber for UAVs

4.3 UAV Dataset Measured with PulsON 440

The measurements are taken in a total of twenty orientations for each UAV as stated in Table 4.2. Ten orientations are obtained by rotating the UAV under test in azimuth plane from 0° to 90° when the quadcopters are placed parallel to the ground. Since quadcopters have a four-fold symmetry, only the angles between 0° and 90° are

considered as illustrated in Figure 4.3 to eliminate redundant measurements. In addition, the UAVs are tilted by 30° in the elevation plane and rotated again to measure the target responses in different azimuth angles to cover the aspect angles seeing the UAVs from the bottom. For each aspect angle 500 scans are collected. With 20 different incidence angles, the total number of signals measured for each UAV becomes 10,000. As a result, the UAV dataset is composed of 40,000 wideband electromagnetic scattered signals in total.

We set the PulseON 440 radar module to listen for 105 nanoseconds after each pulse is transmitted. However, the raw signals cannot be used directly for the classifier design due to the following reason: The first 20 nanoseconds of the measured signals (corresponding to a two-way propagation range of three meters) are strongly contaminated by systematic clutter signals stemming from the hardware of the radar unit as shown in Figure 4.4. Mutual couplings between the closely placed transmitting and receiving antennas are partially responsible for this contamination. So, we placed our targets 5 meters away from the radar unit in the anechoic chamber to avoid these strong clutter signals. Another undesired echo shows up systematically in the measurements right after 100 nanoseconds as shown in Figure 4.5. This echo is caused by the subsequent transmitted pulse. This is because the pulse repetition frequency of the radar unit is 10 MHz, which corresponds to a repetition time interval of 100 nanoseconds. Therefore, time gating is applied to measured signals to select the time range from 30 nanoseconds to 92.4 nanoseconds that provides a clean measurement segment with a time span of 62.4 nanoseconds sampled in 1024 equally spaced points. This time range is long enough to send the excitation pulse to the target (that is 5 meters away from the radar unit) and to receive the returned signal as shown in Figure 4.6. In this figure, the acquired target response is plotted with 8.6 dB average SNR value. Examples of the scattered signals for all quadcopters are shown in Figure 4.7.

Table 4.2 Details of the Measured UAV datasets

Incident Angles		*Number of Signals per Aspect Angle*	*Total Number of Signals for a Target*
Azimuth	Elevation		
0°, 10°, 20°, 30°, 40°, 50°, 60°, 70°, 80°, 90°	0°, 30°	500	10,000

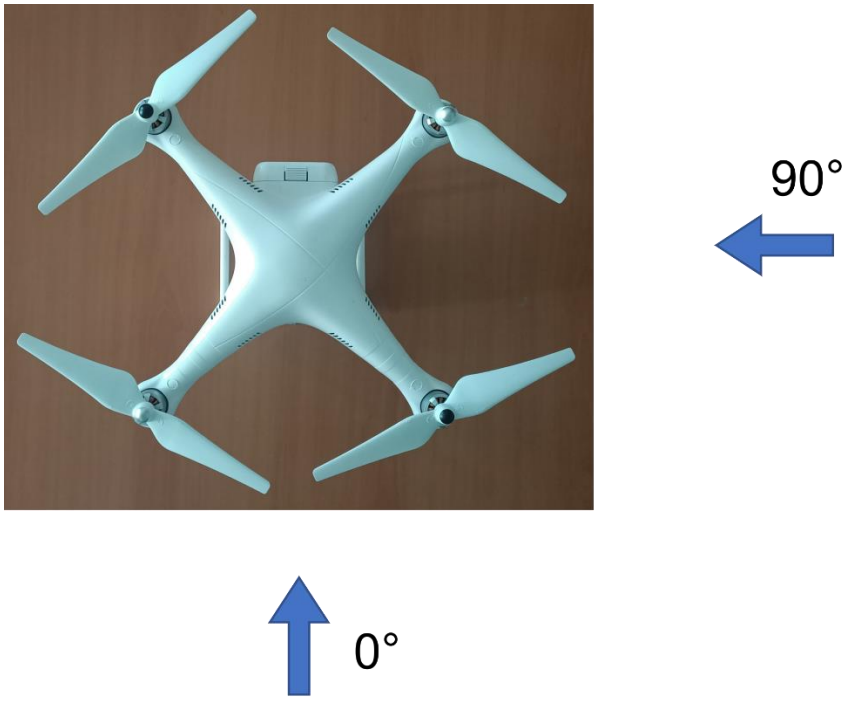


Figure 4.3. The incident angles of the UWB signal to the UAVs

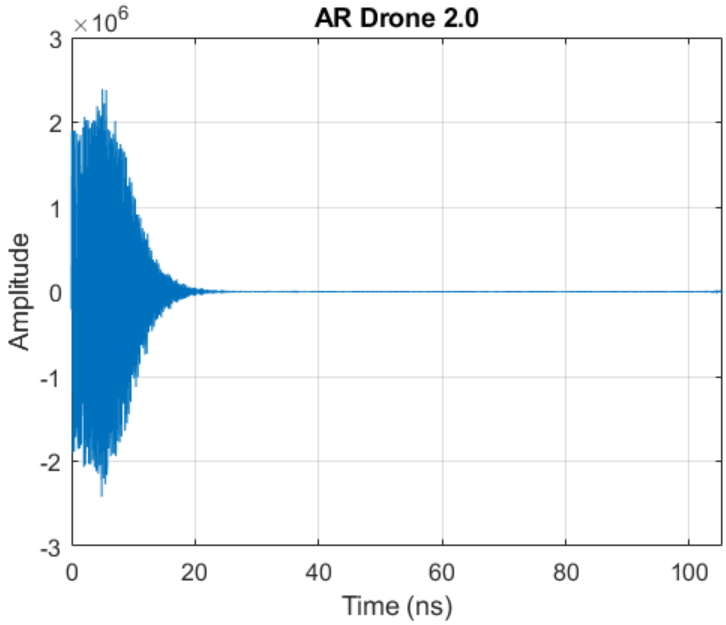


Figure 4.4. An example of the measured signal with PulseON 440 radar module

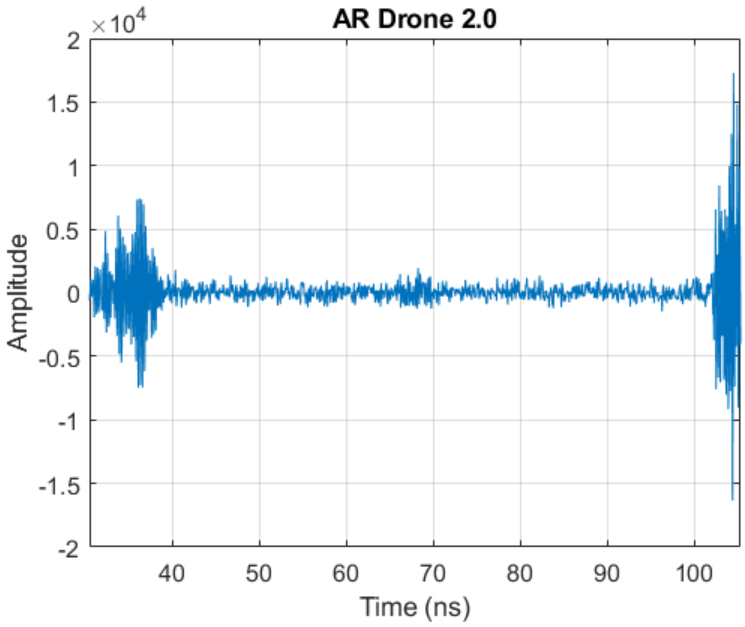


Figure 4.5. The measured signal after the elimination of initial clutter signal.

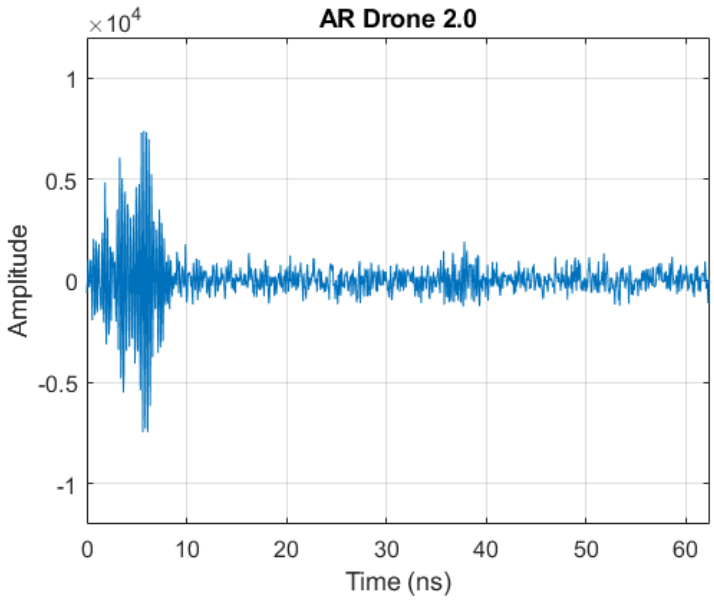


Figure 4.6. The time span of a measured scattered signal selected by time gating to be used in the classifier design.

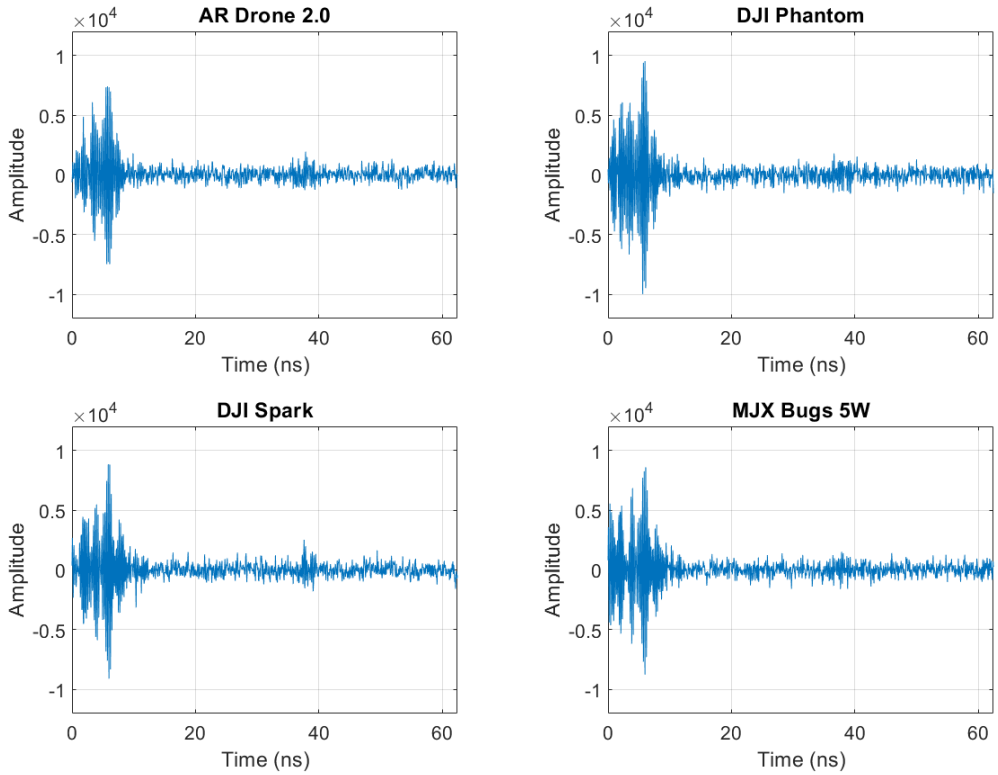


Figure 4.7. Examples of measured scattered signals for each one of four different quadcopters.

In CNN-based classifier, we utilize 5-fold cross validation to evaluate the performance of the neural network. Therefore, the measured dataset is divided into 5 non-overlapping subsets. Before separating the dataset, we randomly shuffle measured scattered signals with respect to aspect angles. Afterwards, the training set is constructed with the scattered signals of 16 different aspect angles in each fold. For the test set, remaining 4 aspect angles are utilized. With this data separation, the number of scattered signals for the training and test sets are given in Table 4.3.

Table 4.3 Number of Samples in Each Set for 5-fold Cross Validation

Training Set	Test Set	Total
32,000	8,000	40,000

4.4 Preprocessing the Input Signal for CNN

Similar to what we did to preprocess the scattered signals of dielectric spheres, two preprocessing methods are applied to the measured UAV signals before they are given as inputs to the CNN. Firstly, we normalize the signals such that the mean becomes zero and the variance is one. Later, shuffling operation is performed to store the input signals in a random order.

4.5 Initial CNN Architecture for UAV Classification

At the first step of UAV classifier design, we start with the network architecture that was proposed in Chapter 3 for the classification of dielectric spheres. The architecture is composed of three convolutional layers, and the hyper-parameters are optimized for the scattered signal dataset of dielectric spheres.

We start to apply 5-fold cross validation to the network. But we finish the cross validation process early after training with the dataset of fold 3 is completed. This is because it is observed that the network could not learn well for 3 different folds. When the training with the dataset of fold 3 is completed, we obtain 83.2% accuracy with the learning curves shown in Figure 4.8. Both in accuracy and loss curves, a gap between training and validation is seen. In addition, validation loss does not decrease over epochs as expected. However, Goodfellow et al. state that a good model is achieved when there is a small gap in between training and validation curves [18]. Even though a relatively good result obtained with the confusion matrix as shown in Figure 4.9, we improve the architecture for UAV classification by deepening the network. The details of the improved CNN architecture are discussed in the following section.

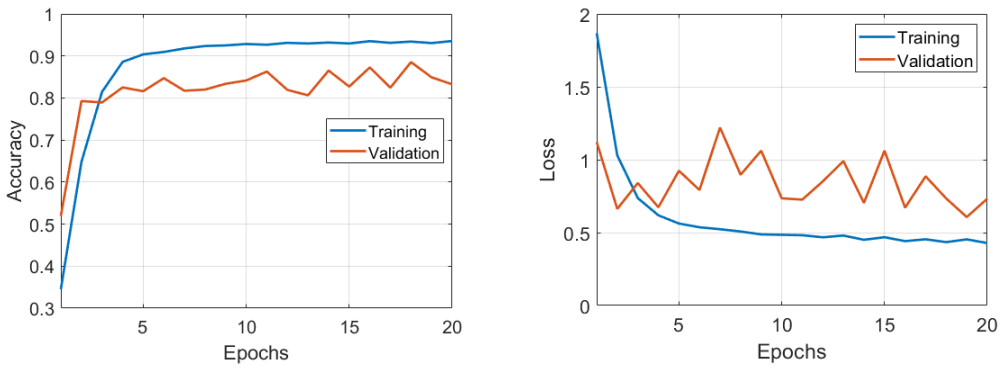


Figure 4.8. The learning curves of the optimized network for the UAV classification

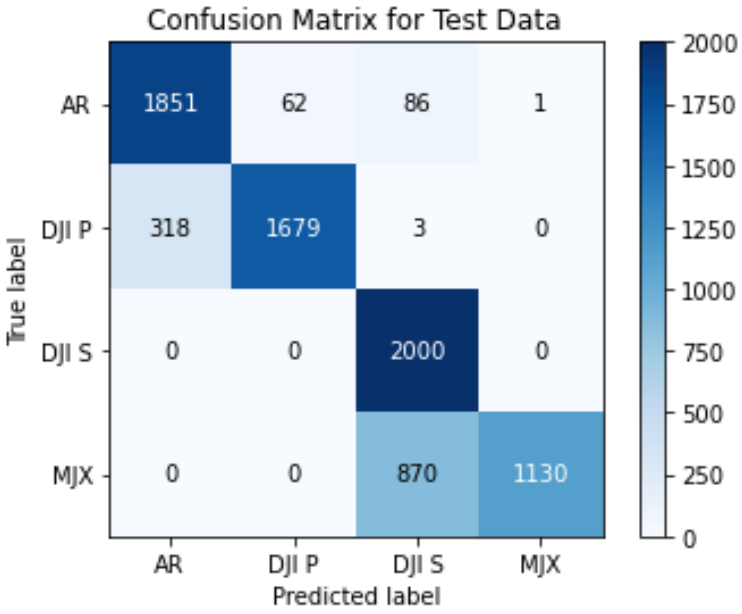


Figure 4.9. Confusion matrix of CNN-based UAV classifier when the architecture optimized for the dielectric spheres is used.

4.6 Improved CNN Architecture for the UAV Classification

Following our first experiments with the initial architecture optimized previously for the database of dielectric spheres, we decided to increase the depth so that the representation power of the network can be enhanced. However, the growth of the total number of parameters is a major concern while deepening a network. This is because a complex model cannot learn the features of the signals from a small dataset, and under-fitting occurs. Although we collect 40,000 scattered signal measurements in total, it is not enough to train a very deep learning model. In this thesis, we propose a CNN architecture inspired by VGG architecture [37] which is a popular image classification network. Simonyan and Zisserman state in [37] that the model performance can be improved by increasing the number of layers. At the same time, the VGG architecture contains stacks of several convolutional layers with small

kernel sizes, so the number of parameters does not grow as much as compared to a network using large convolutional filters.

For our UAV target set, we introduce a CNN architecture with three convolutional layer groups similar to VGG architecture. In the first group, two convolutional layers are stacked while there are three convolutional layers in the second and third groups as illustrated in Figure 4.10. Batch normalization is also added to each convolutional layer. We set 32 filters for the first group, and the remaining groups are composed of 64 filters. After each convolutional layer group, down-sampling is performed with max pooling layers over window size of 2. Following the convolutional layers, a fully connected layer (FCL) with 512 hidden units is placed. For each layer, rectified linear unit is chosen as the activation function. Furthermore, L2 regularization and dropout are defined to avoid overfitting. At the output layer, the activation function is set to softmax function to classify four different UAV targets. We use TensorFlow framework to implement the described CNN architecture on Python. In Appendix G, the code describing the network with TensorFlow functions is given. The training of the CNN model is carried out with NVIDIA Quadro K2200 GPU and Intel Xeon E5-1603 CPU.

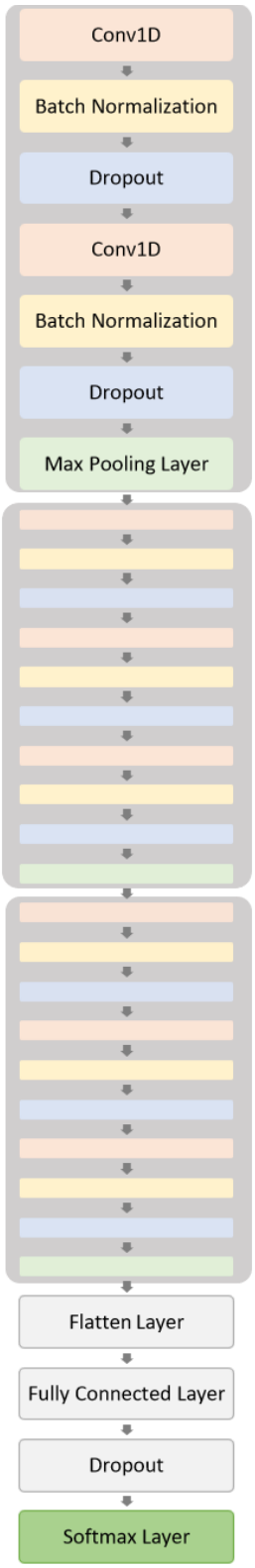


Figure 4.10. CNN architecture used for UAV classification

4.7 Hyper-parameter Optimization for the UAV Classifier

For the architecture defined to classify UAVs in the previous section, we perform a hyper-parameter search to achieve an optimum classifier network. As given in Table 4.4, a total of 27 experiments are conducted to find best values for the kernel size, learning rate and mini-batch size. Similar to dielectric sphere classifier design, the effect of the parameters on the performance of the network is examined with grid search. Accuracy, precision and recall metrics are computed when the network trained with different values of each hyper-parameter, and the results are shown in Figure 4.11.

In each convolutional layer, the kernel size is kept the same to narrow the search space. We work with the kernel size values of 3, 5 and 7 just like in the dielectric sphere classifier design. In addition to kernel size, the examined mini-batch sizes are also selected the same as 32, 64 and 128.

Adam algorithm is utilized as the optimizer of the network. In the initial training practices, we observed that decreasing the learning rate during training process reduces the swing in the learning curves. Therefore, being different from the dielectric sphere classification network, we add a decaying learning rate configuration to the grid search. Besides the constant values of 0.001 and 0.0005 for all epochs, a learning rate scheme starting with 0.001 and reducing to 0.0005 is included in our hyper-parameter search domain. In this case, we train the network for a total of 20 epochs and decrease the learning rate after epoch 10.

As a result of the grid search, the highest performance is achieved in Experiment 18 with the kernel size 5 and the mini-batch size 128. In Experiment 18, the learning rate is set to 0.001 for the first ten epochs and it drops to 0.0005 for the remaining ten epochs. With these hyper-parameter values, an average accuracy of 89.5% is obtained by 5-fold cross validation. In addition to the overall result of the grid search, the influence of each hyper-parameter on the model performance is discussed in the following sections.

Table 4.4 List of Hyper-parameter Optimization Experiments for UAV Classification

Experiment No	Kernel Size	Learning Rate	Mini-Batch Size	Accuracy
Experiment 1	(3, 3, 3)	0.001	32	0,8878
Experiment 2	(3, 3, 3)	0.001	64	0,8571
Experiment 3	(3, 3, 3)	0.001	128	0,8732
Experiment 4	(3, 3, 3)	0.0005	32	0,8691
Experiment 5	(3, 3, 3)	0.0005	64	0,8471
Experiment 6	(3, 3, 3)	0.0005	128	0,8599
Experiment 7	(3, 3, 3)	[0.001, 0.0005]	32	0,8924
Experiment 8	(3, 3, 3)	[0.001, 0.0005]	64	0,8753
Experiment 9	(3, 3, 3)	[0.001, 0.0005]	128	0,8756
Experiment 10	(5, 5, 5)	0.001	32	0,8630
Experiment 11	(5, 5, 5)	0.001	64	0,8759
Experiment 12	(5, 5, 5)	0.001	128	0,8829
Experiment 13	(5, 5, 5)	0.0005	32	0,8780
Experiment 14	(5, 5, 5)	0.0005	64	0,8718
Experiment 15	(5, 5, 5)	0.0005	128	0,8875
Experiment 16	(5, 5, 5)	[0.001, 0.0005]	32	0,8751
Experiment 17	(5, 5, 5)	[0.001, 0.0005]	64	0,8739
Experiment 18	(5, 5, 5)	[0.001, 0.0005]	128	0,8946
Experiment 19	(7, 7, 7)	0.001	32	0,8356
Experiment 20	(7, 7, 7)	0.001	64	0,8796
Experiment 21	(7, 7, 7)	0.001	128	0,8313
Experiment 22	(7, 7, 7)	0.0005	32	0,8693
Experiment 23	(7, 7, 7)	0.0005	64	0,8528
Experiment 24	(7, 7, 7)	0.0005	128	0,8272
Experiment 25	(7, 7, 7)	[0.001, 0.0005]	32	0,8435
Experiment 26	(7, 7, 7)	[0.001, 0.0005]	64	0,8686
Experiment 27	(7, 7, 7)	[0.001, 0.0005]	128	0,8495

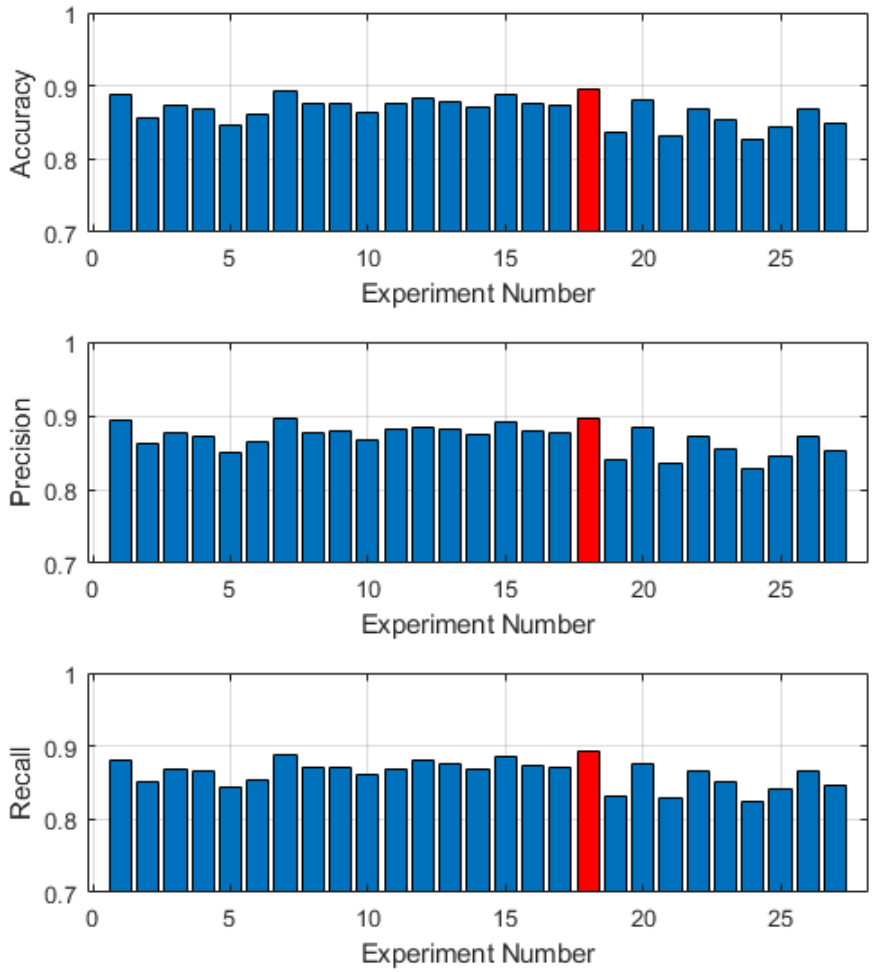


Figure 4.11. The results of hyper-parameter optimization experiments for the UAV classifier.

4.7.1 Impact of the Kernel Size

The variation in the average performance metrics is examined while working with different kernel sizes in the convolutional layers. The results are illustrated in Figure 4.12, and the best scores are obtained for the kernel size 5 where the average accuracy is obtained as 87.8%. For the UAV classifier, we do not observe the positive effect of the kernel size as in the dielectric sphere classifier. When the kernel size is chosen as 7 all performance metrics decrease, and the average accuracy becomes 85%. On contrary, 87% and 87.8% average accuracy is achieved for the kernel sizes 3 and 5 respectively. Therefore, we choose to 5 as the kernel size in the UAV classifier network.

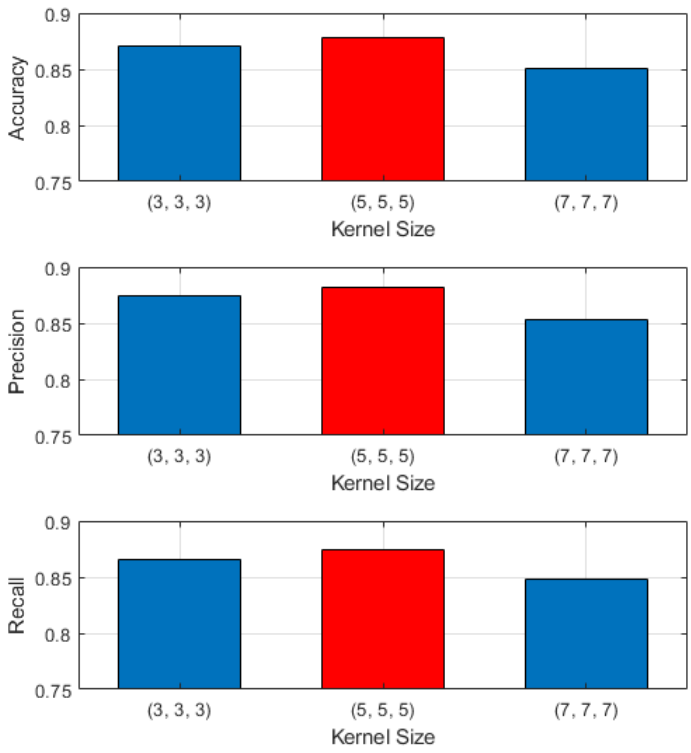


Figure 4.12. Impact of different kernel sizes on the performance of the UAV classifier.

4.7.2 Impact of the Learning Rate

Learning rate is an important hyper-parameter because it is the main multiplier of weight update. Hence, it has a direct influence on the training procedure. In our experiments, we notice that learning rate of 0.001 is a high value for the proposed network architecture. This is because it leads to fluctuations in the learning curves during training. On the other hand, dropping the value to 0.0005 does not solve the problem completely. The training is slowing down, and the validation accuracy does not reach the expected accuracy values as in the case of the learning rate of 0.001. The results shown in Figure 4.13 summarize the situation. The average accuracy for the rate 0.0005 stays at 86% while 86.5% average accuracy is achieved with the rate 0.001.

When we look for another solution to increase classifier performance, it is found that decreasing the learning rate during the training procedure gives better results. Therefore, a learning rate schedule starts with 0.001 and drops to 0.0005 in the middle of the training is utilized. As a result, we obtain an average accuracy of 87.2% with this configuration, which is the highest score among the other learning rate values. The result is also consistent with the outcome of Experiment 18.

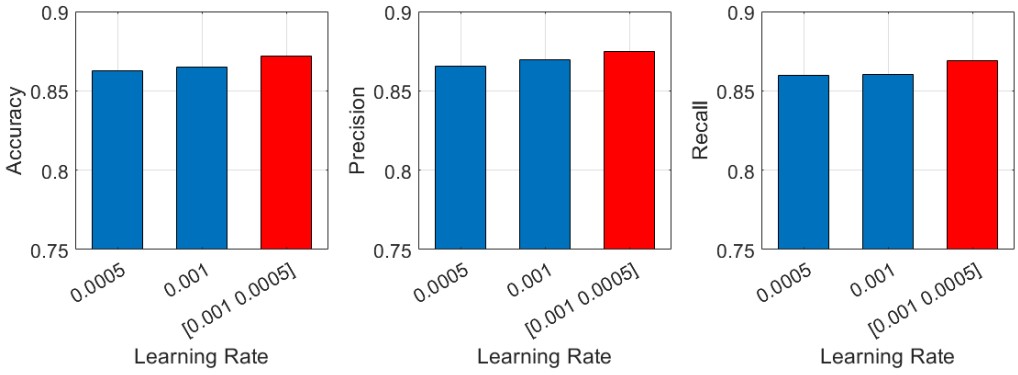


Figure 4.13. Impact of different learning rates on the performance of the UAV classifier.

4.7.3 Impact of the Mini-batch Size

Compared to other hyper-parameters, mini-batch size has no dramatic effect on the UAV classifier network as seen in Figure 4.14. Even though the maximum accuracy of 86.8% is achieved with the mini-batch size 32, the mean accuracy for three different values is received as 86.6%. Besides, the overall best result is obtained with mini-batch size 128 in Experiment 18. This difference could be neglected since the individual effect of the mini-batch size on the performance of the network is negligible. Eventually, we select the mini-batch size of our network as 128 as obtained from the Experiment 18.

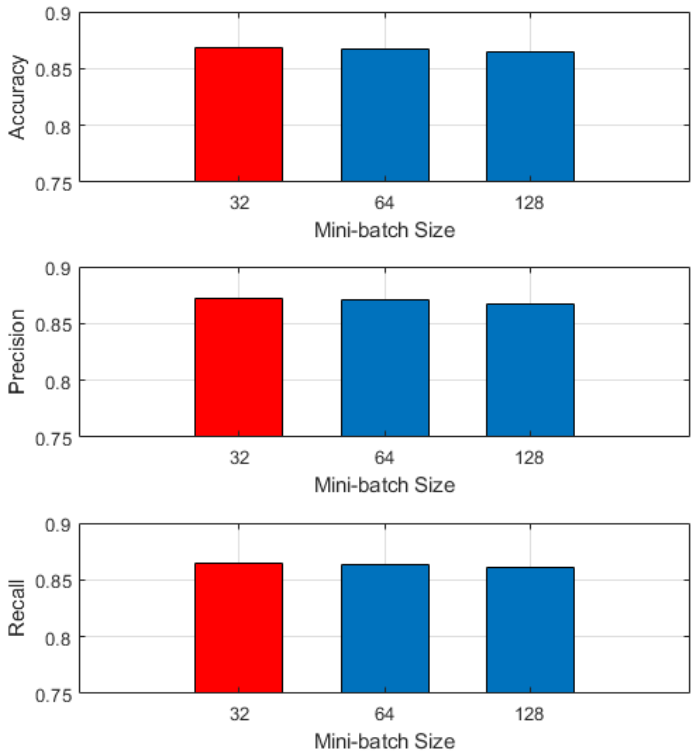


Figure 4.14. Impact of the mini-batch size on the performance of the UAV classifier.

4.8 Training and Evaluation of the Optimized Network

The optimized network for UAV classification is obtained with the grid search for the hyper-parameters stated in the previous section. The best values of the kernel size, learning rate and mini-batch size are determined for the CNN architecture described in section 4.7. As the result of grid search, we achieve the highest performance with the kernel size of 5, decaying learning rate from 0.001 to 0.0005, and mini-batch size of 128. When using these hyper-parameters, the model summary given in Table 4.5 is obtained with a total number of parameters 3,989,348.

We train the optimized network for 20 epochs with the fold 3 to illustrate the performance of the classifier. The learning curves seen in Figure 4.15 arise during the training process. It is easily seen that with the learning rate schedule shown in Figure 4.16 the swing in the curves reduces as the rate decreases to 0.0005. So, the validation curves converge the training curves smoothly. At the end of the training, the confusion matrix shown in Figure 4.17 is obtained. We observe that the classifier makes correct decisions with 92.8% accuracy. In addition, the precision and the recall metrics are obtained as 93% and 92.5% respectively. However, 17% of the samples belong to AR Drone 2 are misclassified as DJI Phantom. Similarly, the classifier predicts 7% of the MJX Bugs 5W measurements as DJI Spark.

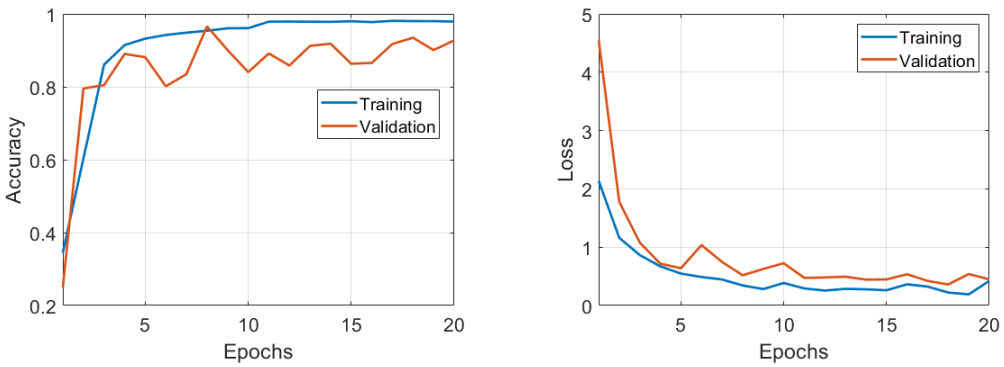


Figure 4.15. The learning curves of the optimized network for the UAV classification.

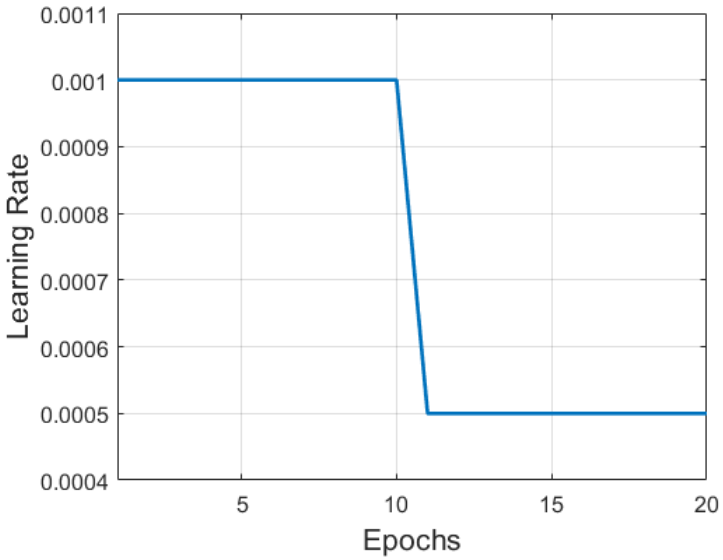


Figure 4.16. The learning rate variation during the training process.

We can explain the confusion between targets with the natural resonance mechanism. This is because the wideband electromagnetic signals are utilized to excite target specific complex natural resonance (CNR) frequencies [3]. As a signal with a larger bandwidth is incident upon a target, a higher number of CNR frequencies could be excited within the selected bandwidth. The CNR frequencies of a target are determined essentially by the material properties as well as the shape and dimensions of a target. Since commercial UAVs are generally made of similar materials and components, the dimensions mainly affect the CNR frequencies. In our target set we observe the confusion between the targets with similar dimension, and the targets are divided into two groups regarding their dimension which stated in Table 4.1. AR Drone 2 and DJI Phantom belong to the bigger target group while the smaller target group is composed of DJI Spark and MJX Bugs 5W.

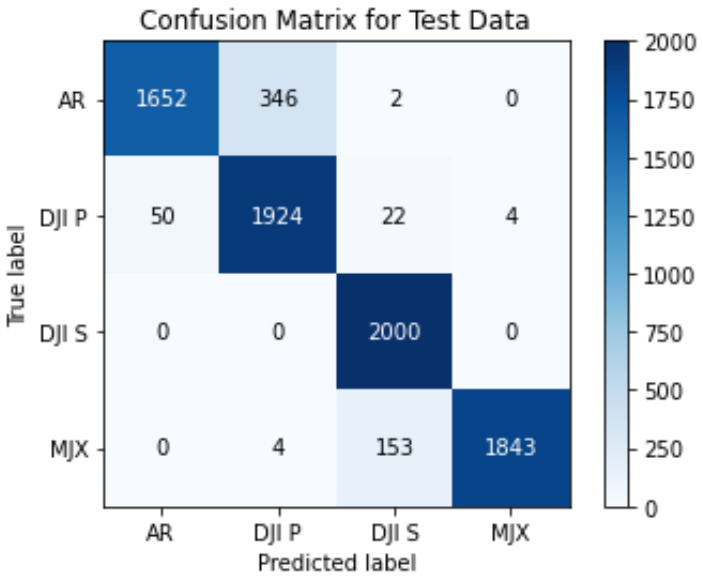


Figure 4.17. Confusion matrix of the optimized network for UAV classification.

Table 4.5 Model Summary of Optimized Network for UAV Classification

Layers	**Output Shape**
Input Layer	1024
Conv 1D, ReLU, L2 Regularization	(1020, 32)
Batch Normalization	(1020, 32)
Dropout	(1020, 32)
Conv 1D, ReLU, L2 Regularization	(1016, 32)
Batch Normalization	(1016, 32)
Dropout	(1016, 32)
Max Pooling 1D	(508, 32)
Conv 1D, ReLU, L2 Regularization	(504, 64)
Batch Normalization	(504, 64)
Dropout	(504, 64)
Conv 1D, ReLU, L2 Regularization	(500, 64)
Batch Normalization	(500, 64)
Dropout	(500, 64)

Table 4.5 (continued)

Conv 1D, ReLU, L2 Regularization	(496, 64)
Batch Normalization	(496, 64)
Dropout	(496, 64)
Max Pooling 1D	(248, 64)
Conv 1D, ReLU, L2 Regularization	(244, 64)
Batch Normalization	(244, 64)
Dropout	(244, 64)
Conv 1D, ReLU, L2 Regularization	(240, 64)
Batch Normalization	(240, 64)
Dropout	(240, 64)
Conv 1D, ReLU, L2 Regularization	(236, 64)
Batch Normalization	(236, 64)
Dropout	(236, 64)
Max Pooling 1D	(118, 64)
Flatten	7552
Fully Connected Layer, L2 Regularization	512
Dropout	512
Softmax Layer	4

4.9 Training of the Optimized Network with a Smaller Training Set

4.9.1 The Dataset Separation

To observe the effect of training set size on the performance of classifier, 50%, 60% and 70% of the dataset is separated as the training subset. The first training set is constructed by the scattered signals from 10 aspect angles, and the remaining 10 aspect angles are left for testing. The selected angles for the training and test sets are summarized in Table 4.6. We prefer to separate the dataset in azimuth angles. So,

the scattered signals measured at 0° and 30° elevation angles are included in both datasets. By adding the scattered signals from 30° azimuth angle, the second training set is established which is 60% of all signals as shown in Table 4.7. Finally, the third training set containing 70% of all data is constructed by including the scattered signals from 70° azimuth angle. The aspect angles composing the third training set are outlined in Table 4.8, and the performance of the resulting networks are compared in the next section.

Table 4.6 Dataset separation: 50% training, 50% test

Dataset	Incident Angles	
	Azimuth	Elevation
Training	0°, 20°, 40°, 60°, 80°	0°, 30°
Test	10°, 30°, 50°, 70°, 90°	0°, 30°

Table 4.7 Dataset separation: 60% training, 40% test

Dataset	Incident Angles	
	Azimuth	Elevation
Training	0°, 20°, 30°, 40°, 60°, 80°	0°, 30°
Test	10°, 50°, 70°, 90°	0°, 30°

Table 4.8 Dataset separation: 70% training, 30% test

Dataset	Incident Angles	
	Azimuth	Elevation
Training	0°, 20°, 30°, 40°, 60°, 70°, 80°	0°, 30°
Test	10°, 50°, 90°	0°, 30°

4.9.2 Training and Evaluation of the Network

The optimized CNN architecture summarized in Table 4.5 is trained with smaller training datasets to examine the effect of training set size. The aspect angles constituting the training and test sets are selected in 3 different ways as stated in Table 4.6, Table 4.7, and Table 4.8. For the first training subset, the dataset is split with 50%-50% ratio. When we train the deep learning model with this training set and use the test set for direct validation only (without using any cross-validation and without making any changes either in the structure or in the hyper-parameters) the learning curves shown in Figure 4.18 are obtained. It is seen in the loss curve that the network is trained without overfitting similar to previous curves in Figure 4.15. However, the same performance could not be achieved. The accuracy is obtained as 84.3% with the training set stated in Table 4.6, and the confusion matrix seen in Figure 4.19 is acquired. When the training of the network is repeated two more times, 82.5% mean accuracy is achieved. Besides, with this classifier 82.7% average precision and 82.4% average recall are received. This performance degradation is expected because the network attempts to learn signal characteristics using a smaller amount of information. Therefore, a gap between the training and validation accuracy occurs and, we reach a weaker classifier compared to the one stated in section 4.8.

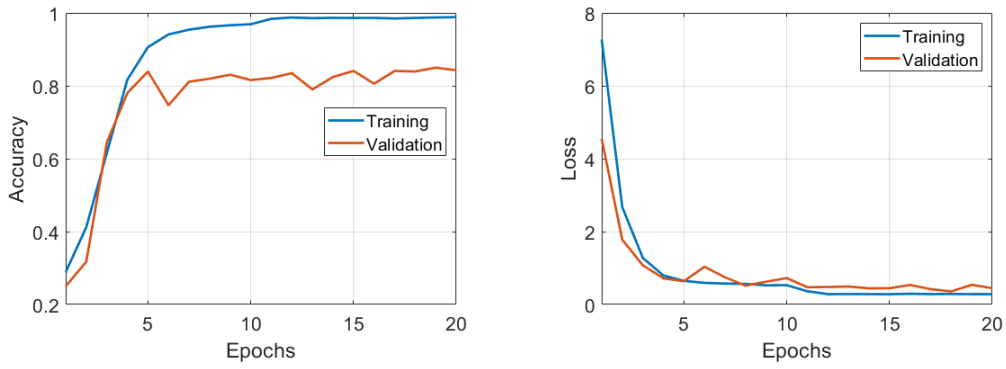


Figure 4.18. The learning curves of the optimized network when the dataset is composed of the aspect angles in Table 4.6 with 10 different aspects.

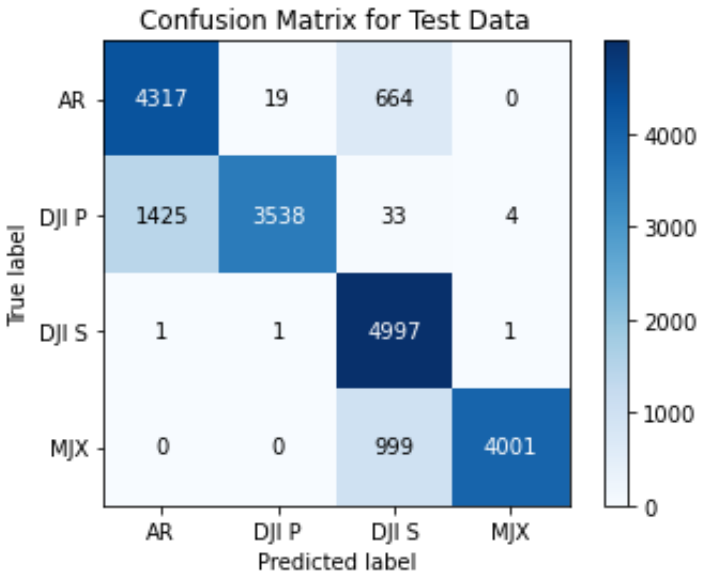


Figure 4.19. Confusion matrix of CNN-based UAV classifier when the training dataset is composed of the aspect angles in Table 4.6 with 10 different aspects.

In the second training set, we increase the number of aspect angles to 12 by including the scattered signals from 30° azimuth angle. As a result of 3 repetition of training the network from scratch, we achieve accuracy values of 81.9%, 86% and 86.6% whose average is 84.8%. So, 2.3% rise in the average accuracy of the classifier is observed by increasing the number of aspect angles to 12. In Figure 4.20, the learning curves are shown for the network with 86% accuracy. 86.1% precision and 85.9% recall is received with this network and the confusion matrix is illustrated in Figure 4.21. We could say that an improvement in the performance of the classifier is occurred with the training set stated in Table 4.7. Furthermore, the performance metrics approaches the values reported in section 4.8.

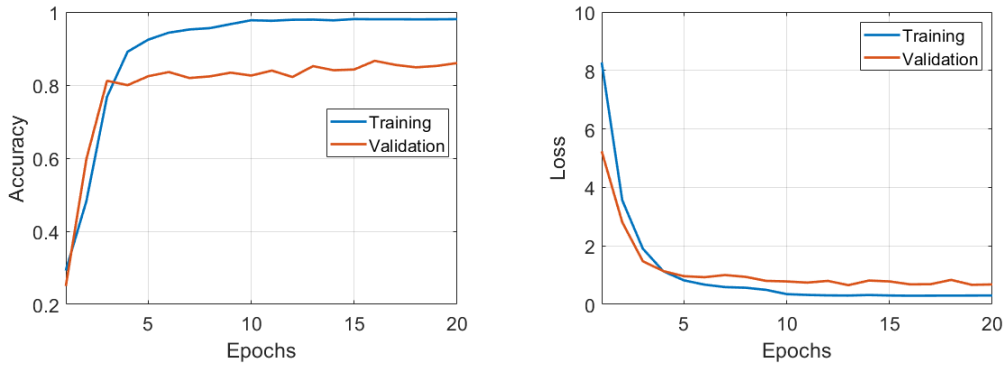


Figure 4.20. The learning curves of the optimized network when the dataset is composed of the aspect angles in Table 4.7 with 12 different aspects.

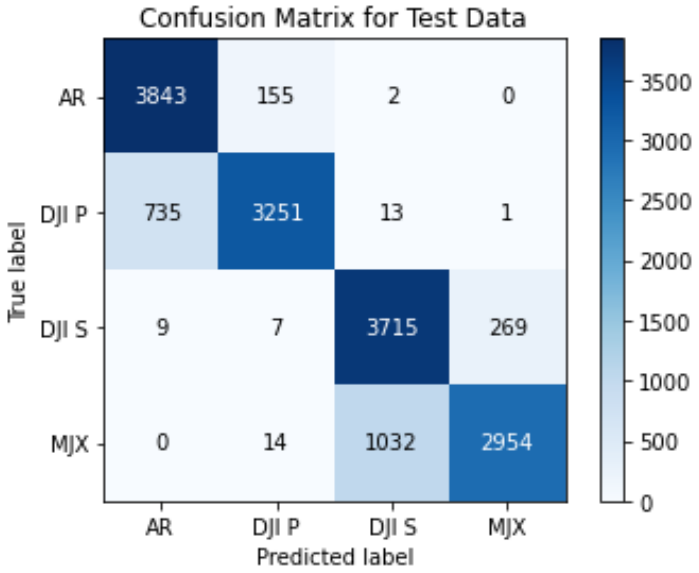


Figure 4.21. Confusion matrix of CNN-based UAV classifier when the training dataset is composed of the aspect angles in Table 4.7 with 12 different aspects.

After examining the performance of the network trained with the dataset containing 12 aspect angles, we add the scattered signals from 70° azimuth angles to the training set also as given in Table 4.8. By increasing the number of aspect angles to 14, 87.3%, 88.9% and 93.7% accuracy values are achieved as a consequence of 3 repeats

of training. For these 3 trials, 89.9% average accuracy is achieved which is 5.1% more than the case for 12 aspect angles. It is observed that the performance of classifier reported in section 4.8 is exceeded with 93.7% accuracy in one of the trials. For this network, the learning curves seen in Figure 4.22 are obtained, and the resulting confusion matrix is shown in Figure 4.23. The precision and recall metrics of the network is received as 94% and 93.4%, respectively. In conclusion, we observe that the performance of the CNN-based quadcopter classifier improves with the increasing number of aspect angles in the training set. It is an expected result because the network could learn better when more information in additional aspect angles is provided.

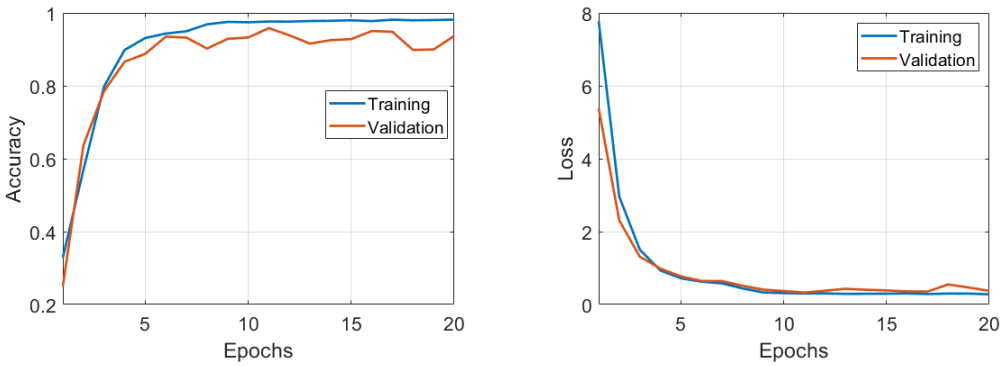


Figure 4.22. The learning curves of the optimized network when the dataset is composed of the aspect angles in Table 4.8 with 14 different aspects.

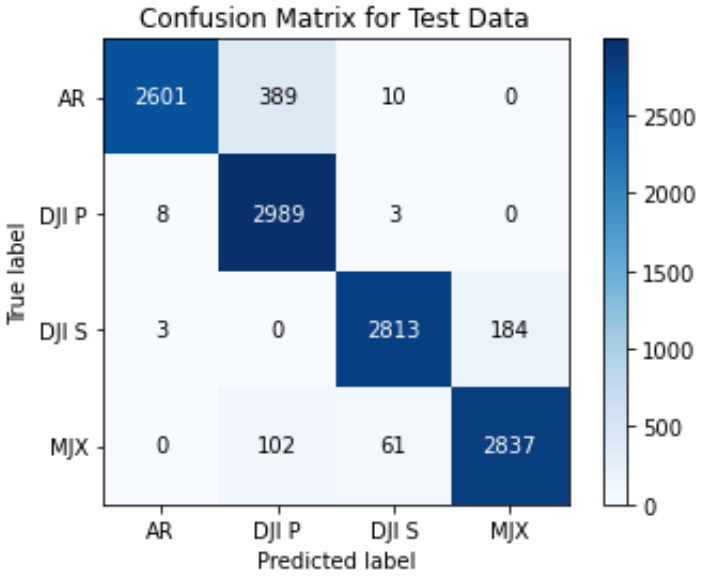


Figure 4.23. Confusion matrix of CNN-based UAV classifier when the training dataset is composed of the aspect angles in Table 4.8 with 14 different aspects.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

In this thesis, the convolutional neural network (CNN) type deep learning approach is applied to the electromagnetic target classification problem for two different target set libraries; for a library of four lossless dielectric spheres of the same size but with slightly different dielectric constants, and a library of four quadcopter type drones of similar sizes. The datasets used for training, validation and testing in these classifier design problems are composed of one-dimensional, broadband, time-domain scattered signals. Aspect and polarization dependence of these scattered signals makes the target recognition problem complicated calling for the use of ultra-wide band data, because using wider bandwidth means better characterization of the targets as explained by the SEM theory. While there exist various physics-based methods for electromagnetic target recognition, most of them rely on the direct or indirect use of targets' complex natural resonance frequencies. One of the recent and well-established methods falling into this group is the WD-PCA method that utilizes target features extracted by the help of Wigner distribution and principal component analysis [11]. The WD-PCA method is used as the comparison case to verify the feasibility of the CNN-based classifier design when it is applied to a target set of lossless dielectric spheres using an ultra-wide band simulation data.

As shown in Chapter 3, when both WD-PCA based classifier and the CNN-based classifier are trained using exactly the same 5-aspect sphere database (composed of the bi-static aspect angles of 5°, 45°, 90°, 135° and 179°), the WD-PCA classifier reached 100% accuracy rate at 20 dB SNR level (and also at noise-free case) but the CNN-based classifier could only reach an average accuracy rate of 93.2 %. While the classifier design by the WD-PCA method is perfectly repeatable, the classifier design by the CNN method shows fluctuating performances, the accuracy rate ranges from 90.4% to 97.4% in 5 repetitions of the classifier design. The CNN-based

classifier design method reaches the performance level of WD-PCA based classifier design method only when additional information on aspect angles are provided in the training phase, including data at 7 aspects (5°, 30°, 60°, 90°, 120°, 150°, and 179°).

Regarding the noise performances of the classifiers, the accuracy rate of the CNN-based classifier (designed in Section 3.7) remains over 90% while the WD-PCA classifier's accuracy drops to 75% at 5 dB SNR level. Although the CNN classifier's noise performance looks much better than that of the WD-PCA classifier, this comparison is biased due to the fact that a large amount of noisy database signals at 15, 17 and 20 dB SNR values are used for CNN training due to need for data augmentation. The WD-PCA classifier on the other hand, uses a very small amount of noise-free training data. More research is needed to make unbiased comparisons in this respect.

As the novel part of this thesis, the CNN-based target classifier design approach is applied to a set of four similar commercial quadcopters using an experimentally constructed scattered database over an extremely limited bandwidth. Various classifiers are designed for different network structures and hyper-parameter values using training databases at varying combinations of aspect angles. Successful quadcopter classifier designs are realized attaining the correct decision rates up to 93.7% in this challenging problem demonstrating the feasibility and usefulness of the CNN-based electromagnetic target classification technique. Provided that a sufficiently large database is available, even when the bandwidth of the database is extremely restricted as in the case of quadcopter database, the classifier still succeeds in attaining accuracy levels above 90% with careful tuning of the hyper-parameters of the CNN-training process.

Most of the previous research papers reporting CNN-based classifiers using one-dimensional time-domain signals in ATR design have used very low frequency ECG signals [20] which do not suffer from physically dictated complications of aspect/polarization dependencies. Only a few studies use electromagnetic scattered

data like we do but they mostly make use of ultra-wideband simulation data as well as ultra-wideband measurement data [21-22]. Another distinct aspect of the CNN-based classifier design approach presented in this thesis is that our design is completed in a single stage rather than needing multiple stages as reported in [21-22].

The major drawback of the CNN-based classification method is the need for a huge training dataset. The CNN based design also needs data with more variation in aspect angle as compared to the WD-PCA classifier approach. While the WD-PCA method is consistently repeatable to design classifiers with very high correct decision rates using a very small amount of noise-free scattered dataset (at five aspects for each one of four spherical targets, i.e. composed of only 20 scattered signals), the CNN based classifier design can reach this level of consistency and accuracy after using data at 7 aspects for each spherical target with a high level of data augmentation involving three different SNR levels with 30 repetitions at each target/aspect/SNR level combination, with a total of 2520 scattered signals.

Although the need for a large database size sounds to be a disadvantage of the CNN-based classifier design approach, each method has its own advantages and disadvantages depending upon the characteristics of the available scattered database. On one hand, the physics based WD-PCA approach is superior if the size and aspect variation of the training dataset are extremely restricted. On the other hand, the CNN based classifier design approach is superior if a sufficiently large database with enough aspect variation is available. The need for physics-based feature extraction in the WD-PCA approach is eliminated in the CNN based classifier design method. This highly appreciated asset of the CNN approach is made possible by the convolutional layers of the CNN structure, which learn the target features hidden in the one-dimensional electromagnetic scattered database.

Design of a WD-PCA based classifier for the quadcopter target set using the same limited bandwidth data is planned as a future work to better evaluate the capacity of

the CNN-based electromagnetic classifier design approach as compared to the WD-PCA classifier design technique.

REFERENCES

[1] E.M. Kennaugh, R.L. Cosgriff, “The use of impulse response in electromagnetic scattering problems,” IRE National Convention Record—Part 1, pp. 72–77, 1958.

[2] M. L. Van Blaricum, R. Mittra, “A technique for extracting the poles and residues of a system directly from its transient response,” IEEE Trans. Antennas and Propagation, vol. 23, no. 6, pp. 777-781, Nov. 1975.

[3] C.E. Baum, “Toward an engineering theory of electromagnetic scattering: the singularity and eigenmode expansion methods,” in: P.L.E. Uslenghi (Ed.), Transient Electromagnetic Fields, Academic Press, New York, pp. 571-651, 1978.

[4] E. M. Kennaugh, “The K-pulse concept,” IEEE Trans. Antennas and Propagation, vol. 29, no. 2, pp. 327-331, Mar. 1981.

[5] G. Turhan-Sayan and D. L. Moffatt, “K-Pulse Estimation Using Legendre Polynomial Expansions and Target Discrimination,” Journal of Electromagnetic Waves and Applications, Vol.4, No. 2, pp. 113-128, 1990.

[6] G. Turhan-Sayan and M. Kuzuoglu, "Pole Estimation for Arbitrarily-Shaped Dielectric Targets by a Genetic Algorithm-Based Resonance Annihilation Technique," Electronics Letters, Vol. 37, No. 6, pp.380-381, March 2001.

[7] C. Chen, "Electromagnetic resonances of immersed dielectric spheres," IEEE Transactions on Antennas and Propagation, vol. 46, no. 7, pp. 1074-1083, July 1998, doi: 10.1109/8.704811.

[8] M. Secmen and G. Turhan-Sayan, “Radar Target Classification Method with Reduced Aspect Dependency and Improved Noise Performance Using Multiple Signal Classification Algorithm,” IET Radar, Sonar & Navigation, Vol. 3, Issue 6, pp. 583-595, December 2009.

[9] M. Secmen and G. Turhan-Sayan, “The MUSIC algorithm-based electromagnetic target classification for isolated targets from incomplete frequency domain data,” 2007 Antennas and Propagation Symposium, pp. 4601-4604, Honolulu, Hawai, USA, June 2007.

[10] G. Turhan-Sayan, “Natural resonance-based feature extraction with reduced aspect sensitivity for electromagnetic target classification,” Pattern Recognition, vol. 36, no. 7, pp. 766-776, July 2003.

[11] G. Turhan-Sayan, “Real time electromagnetic target classification using a novel feature extraction technique with PCA-based fusion,” IEEE Trans. Antennas and Propagation, vol. 53, no. 2, pp. 766-776, Feb. 2005.

[12] B. K. Kim, H. S. Kang and S. O. Park, "Drone classification using convolutional neural networks with merged Doppler images," IEEE Geoscience and Remote Sensing Letters, vol. 14, no. 1, pp. 38-42, Jan. 2017, doi: 10.1109/LGRS.2016.2624820.

[13] H. Han et al., "Object classification on raw radar data using convolutional neural networks," 2019 IEEE Sensors Applications Symposium (SAS), 2019, pp. 1-6, doi: 10.1109/SAS.2019.8706004.

[14] M. W. Roth, “Survey of neural network technology for automatic target recognition,” IEEE Transactions on Neural Networks, vol. 1, no. 1, pp. 28–43, 1990.

[15] T. Kohonen, “Self-Organization and Associative Memory,” Springer-Verlag, Berlin,1989.

[16] G. Turhan-Sayan, K. Leblebicioglu, T. Ince, “Electromagnetic target classification using time-frequency analysis and neural networks,” Microwave Opt. Tech. Lett., vol. 21, no. 1, pp. 63-69, April 1999.

[17] T. Katılmış, E.Ekmekçi and G. Turhan-Sayan, “Unsupervised Electromagnetic Target Classification by Self-Organizing Map Type Clustering,” PIERS 2010 in Cambridge, MA, USA, July 5-8, 2010.

[18] I. Goodfellow, Y. Bengio, and A. Courville, “Deep Learning,” MIT Press, 2016.

[19] Z. Geng, H. Yan, J. Zhang and D. Zhu, "Deep-Learning for Radar: A Survey," IEEE Access, vol. 9, pp. 141800-141818, 2021.

[20] S. Kiranyaz, T. Ince and M. Gabbouj, "Real-Time Patient-Specific ECG Classification by 1-D Convolutional Neural Networks," IEEE Transactions on Biomedical Engineering, vol. 63, no. 3, pp. 664-675, March 2016.

[21] M. Alper Selver, T. Toprak, M. Seçmen and E. Y. Zoral, "Transferring Synthetic Elementary Learning Tasks to Classification of Complex Targets," IEEE Antennas and Wireless Propagation Letters, vol. 18, no. 11, pp. 2267-2271, Nov. 2019.

[22] T. Toprak, M. Alper Selver, M. Seçmen and E. Y. Zoral "Utilizing Resonant Scattering Signal Characteristics Via Deep Learning for Improved Classification of Complex Targets," Turkish Journal of Electrical Engineering and Computer Sciences: vol. 29, no. 1, article 22, 2021.

[23] S. Rahman and D.A. Robertson, “Classification of Drones and Birds using Convolutional Neural Networks Applied to Radar Micro-Doppler Spectrogram Images,” IET Radar, Sonar & Navigation, vol. 14, no. 5, pp. 653-661, May 2020, doi: 10.1049/iet-rsn.2019.0493.

[24] D. Raval, E. Hunter, S. Hudson, A. Damini, and B. Balaji, “Convolutional Neural Networks for Classification of Drones Using Radars,” Drones, vol. 5, no. 4, p. 149, Dec. 2021, doi: 10.3390/drones5040149.

[25] D. Kawaguchi, R. Nakamura and H. Hadama, "Evaluation on a Drone Classification Method Using UWB Radar Image Recognition with Deep Learning," 2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring), pp. 1-5, 2021, doi: 10.1109/VTC2021-Spring51267.2021.9448946.

[26] F. Hlawatsch and G.F. Boudreaux-Bartels, “Linear and Quadratic Time-Frequency Signal Representations,” IEEE Signal Processing Magazine, pp. 21-67, 1992.

[27] M. Secmen, “Novel music algorithm based electromagnetic target recognition method in resonance region for the classification of single and multiple targets,” Ph.D. dissertation, METU, Ankara, Turkey, 2008.

[28] M. Kerker, “The Scattering of Light and Other Electromagnetic Radiation,” Academic Press, New York, 1969.

[29] A. Mueller, S. Guido, “Introduction to machine Learning with Python: A guide for Data Scientists,” O’Reilly Media, Newton, MA, USA, 2017.

[30] S. Ioffe, C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," International conference on machine learning, PMLR, 2015.

[31] P440 Datasheet / User Guide, TimeDomain, Huntsville, AL, 2017.

[32] *P440 Operation Description / Theory of Operation*, TimeDomain, Huntsville, AL, 2015.

[33] G. Turhan-Sayan and E. Ergin, “Non-Destructive Recognition of Dielectric Coated Conducting Objects by Using WD Type Time-Frequency Transformation and PCA Based Fusion,” International Journal of RF and Microwave Computer-Aided Engineering, Vol. 23, Issue 4, pp. 403-409, (DOI 10.1002/mmce.20726) July 2013.

[34] Tufan Taylan Katılmış, “Design of Self Organizing Map Type Electromagnetic Target Classifiers for Dielectric Spheres and Conducting Aircraft Targets with Investigations of Their Noise Performances,” Master Thesis, METU, Ankara, Turkey, November 2009.

[35] “CS231n: Deep Learning for Computer Vision,” [Online]. Available: https://cs231n.github.io. [Accessed: 22-Nov-2021)

[36] “What is a convolutional neural network?: 3 things you need to know,” MATLAB&Simulink. [Online]. Available: https://www.mathworks.com/discovery/convolutional-neural-network-matlab.html. [Accessed: 10-Mar-2022].

[37] K. Simonyan, A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” 2014, arXiv: 1409.1556.

[38] C. Shorten, T.M Khoshgoftaar, “A survey on Image Data Augmentation for Deep Learning,” in J Big Data, vol. 6, article 60, 2019, doi: 10.1186/s40537-019-0197-0.

[39] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” Information Processing & Management, vol. 45, no. 4, pp. 427–437, 2009.

[40] J.E. Jackson, A User’s Guide to Principal Components. New York, Wiley, 1991.

APPENDICES

A. Singularity Expansion Method (SEM) and Representation of the System Response for an Electromagnetic Target

An isolated object with a finite size can be represented as a linear time-invariant (LTI) system with an aspect and polarization dependent system function in complex frequency domain (i.e. the Laplace Transform of the object's impulse function) as [1-4]

$$H(s,\theta_d,\phi_d) = E(s,\theta_d,\phi_d) + \sum_{n=1}^{\infty} \frac{R_n(s,\theta_d,\phi_d)}{(s-s_n)(s-s_n^*)} \qquad \text{(A.1)}$$

where $s = \alpha + j\omega$ is the complex frequency with $\omega = 2\pi f$; θ_d and ϕ_d are the elevation and azimuth angles in spherical coordinates; $H_{forced}(s,\theta_d,\phi_d)$ is the forced component of the system function formed basically by the specular returns from the object, hence it is highly aspect/polarization dependent; (s_n, s_n^*) are the complex-conjugate system pole pairs (i.e. the complex natural resonance (CNR) frequencies which are independent of aspect/polarization variations) having the aspect/polarization dependent residues $R_n(s,\theta_d,\phi_d)\ for\ n = 1,2,3,\ldots$.

The inverse Laplace transform of $H(s,\theta_d,\phi_d)$ can be expressed as

$$h(t,\theta_d,\phi_d) = h_{forced}(t,\theta_d,\phi_d) + \sum_{n=1}^{\infty} b_n(\theta_d,\phi_d)e^{\alpha_n t}\cos(\omega t + \delta_n) \qquad \text{(A.2)}$$

where $h_{forced}(t,\theta_d,\phi_d)$is the forced response component that exists only during the direct interaction of the excitation signal with the object, hence it is time limited and effective only at early times. The summation terms, on the other hand represents the

superposition of damped sinusoidal signal components created by complex conjugate pole pairs. These sinusoidal terms are excited as the excitation signal passes through the object, they partially overlap with the forced response but exist until very late times while exponentially decaying in time. Oscillation frequencies and decaying rates of damped sinusoids are determined by the target poles (i.e. CNR frequencies) and their magnitudes depend on the values of residues R_n's defined in Equation (A.1). When the attenuation constant α of a pole pair is small, then the associated sinusoidal signal is called "dominant" as its oscillations persist for a longer time. As the strength of such dominant modes are determined by aspect dependent residues, a dominant mode becomes effective at varying orders at different aspect angles. Therefore, a target can be characterized more completely by using its scattered signals at a collection of many different aspects and/or polarizations.

The transient response of a target could be divided into two parts as early-time response and late-time response as illustrated in Figure A.1. Mostly, the direct reflections from the target are observed along with just starting damping sinusoids in the early-time response. The early-time portion of the scattered signal is not distinctive for physics-based feature extraction purposes because the effects of CNR frequencies are not dominant yet. On the other hand, the use of target specific late-time response is preferable in target classifiers such as those designed by the WD-PCA method after the time-limited forced response vanishes.

However, the CNN-based classifier design presented in this thesis uses both early-time and late-time portions of the scattered target response signals without isolating any part.

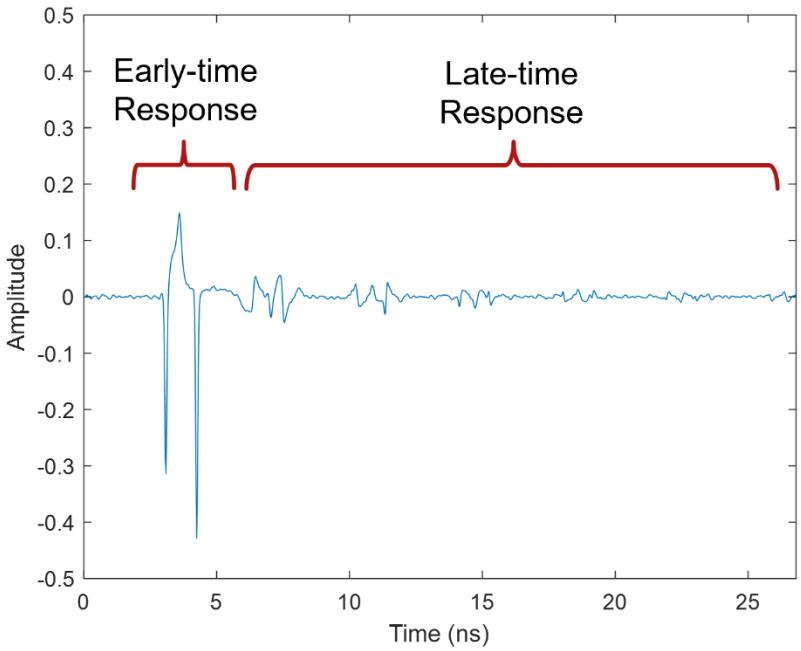


Figure A.1. Early-time and late-time responses of a scattered signal

B. Scattered Far-Field Time-Domain Signals of the Dielectric Sphere Database

Details of the data simulation procedure for the database of dielectric spheres are outlined in section 3.1.2.1. The resulting time-domain far field scattered signals (computed from Mie series solutions) are presented in this Appendix B for four different dielectric sphere targets are given below in Figures B.1, B.2, B.3 and B.4.

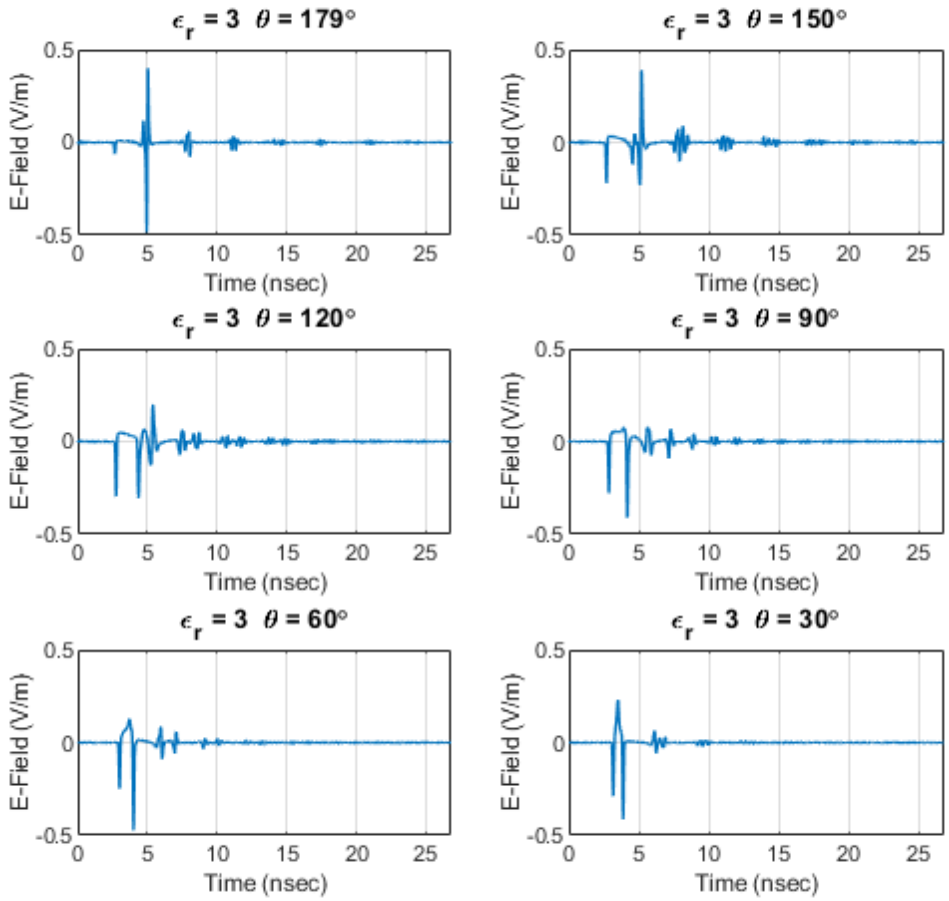


Figure B.1. The simulated time domain scattered signals of the dielectric sphere with $\epsilon_r = 3$ at the bi-static angles 179°, 150°, 120°, 90°, 60°, 30°.

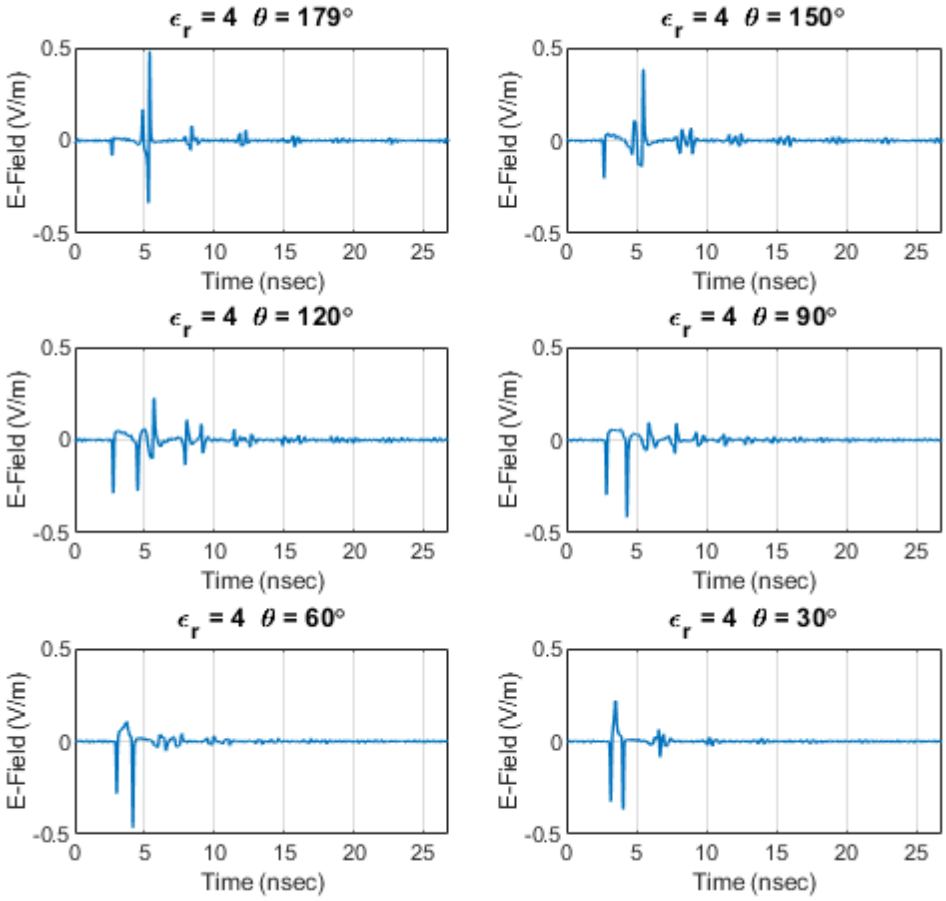


Figure B.2. The simulated time domain scattered signals of the dielectric sphere with $\epsilon_r = 4$ at the bi-static angles 179°, 150°, 120°, 90°, 60°, 30°.

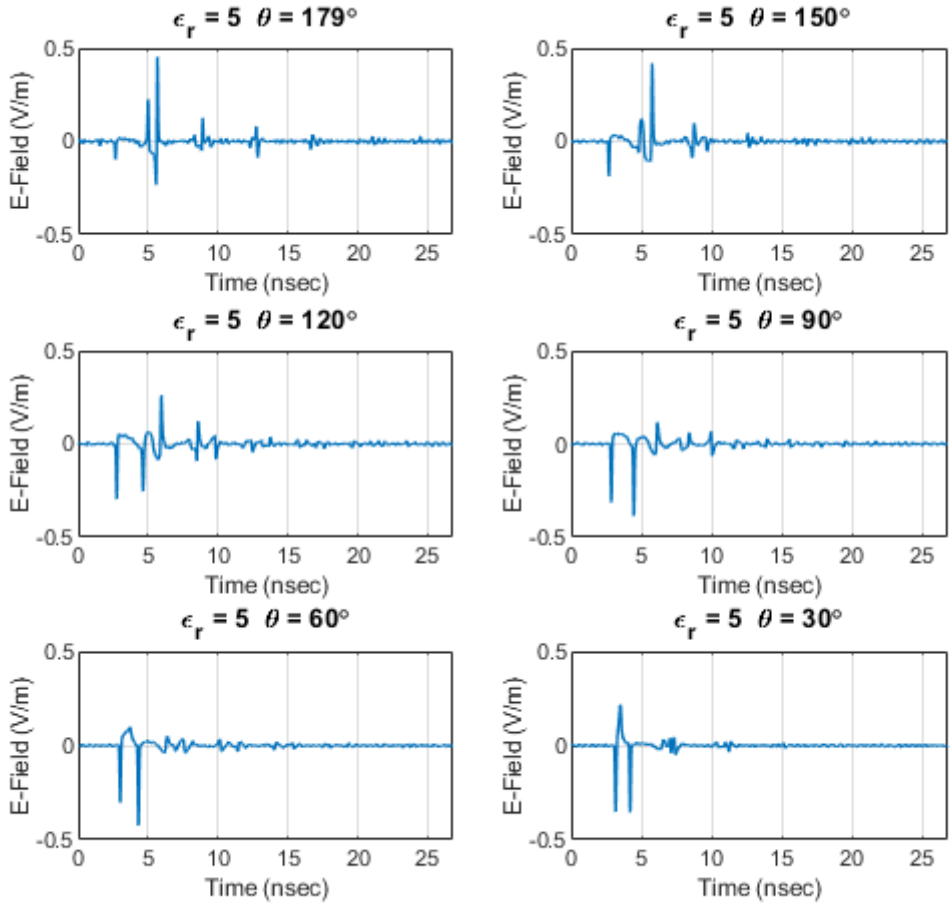


Figure B.3. The simulated time domain scattered signals of the dielectric sphere with $\epsilon_r = 5$ at the bi-static angles 179°, 150°, 120°, 90°, 60°, 30°.

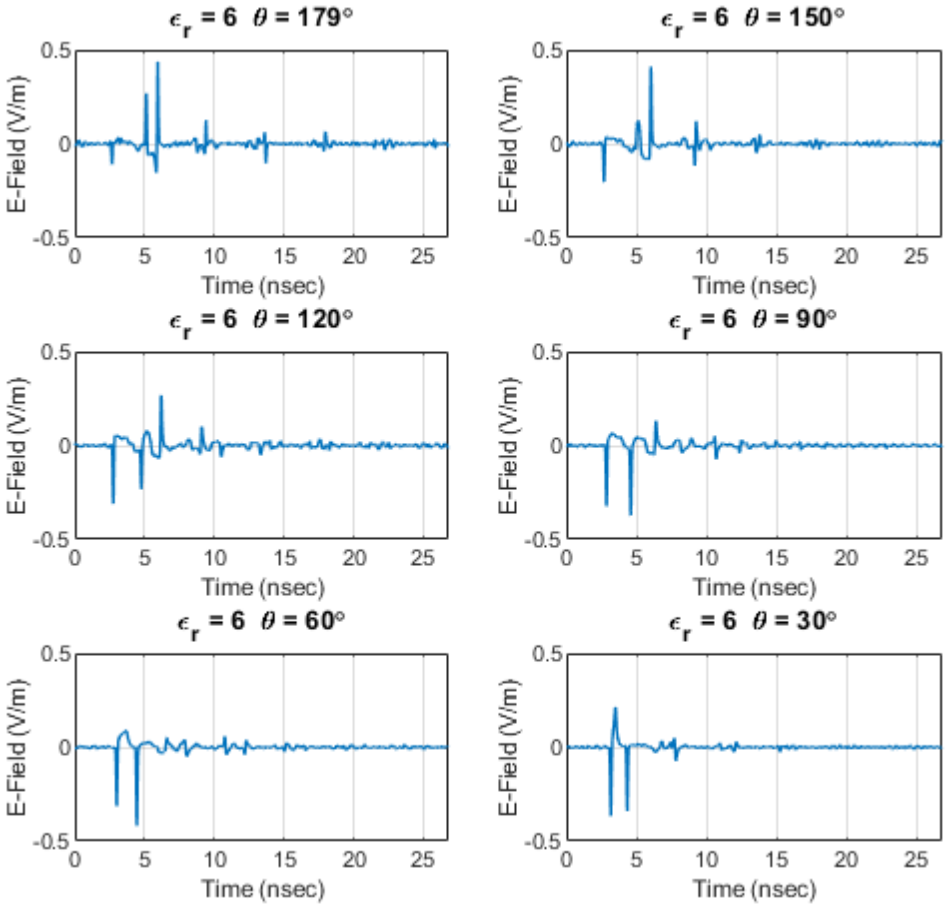


Figure B.4. The simulated time domain scattered signals of the dielectric sphere with $\epsilon_r = 6$ at the bi-static angles 179°, 150°, 120°, 90°, 60°, 30°.

C. Performance Metrics of the Neural Network Based Classifiers

In this Appendix section, the performance metrics to evaluate the CNN-based classifiers are stated. We select three performance measures in this thesis as accuracy, precision, and recall. These metrics could be defined by the help of the confusion matrix of binary classification as seen in Table C.1. The accuracy of a classifier is the ratio of correct decisions ($tp + tn$) to all predictions as formulated in Equation (C.1). In [39], the accuracy is defined as the overall effectiveness of a classifier. Whereas precision is the metric corresponding to the correct decisions among the positive predictions as seen in Equation (C.2). How many of the actually positive cases are predicted correctly is also defined as recall by Equation (C.3).

Table C.1 Confusion Matrix for Binary Classification

		Prediction	
		Positive	Negative
Actual Data Class	Positive	tp	fn
	Negative	fp	tn

$$Accuracy = \frac{tp + tn}{tp + fn + fp + tn} \tag{C.1}$$

$$Precision = \frac{tp}{tp + fp} \tag{C.2}$$

$$Recall = \frac{tp}{tp + fn} \tag{C.3}$$

D. Confusion Matrices for Noisy Dielectric Sphere Signals

In section 3.7.3, the noise performance of the CNN-based classifier for dielectric spheres is discussed. The network is trained with a smaller training set as stated in section 3.7.1. The resulting confusion matrices for different sets of testing signals at the SNR levels of 0 dB, 5 dB, 10 dB, 15 dB, and 20 dB are shown in figures D.1 thru D.5.

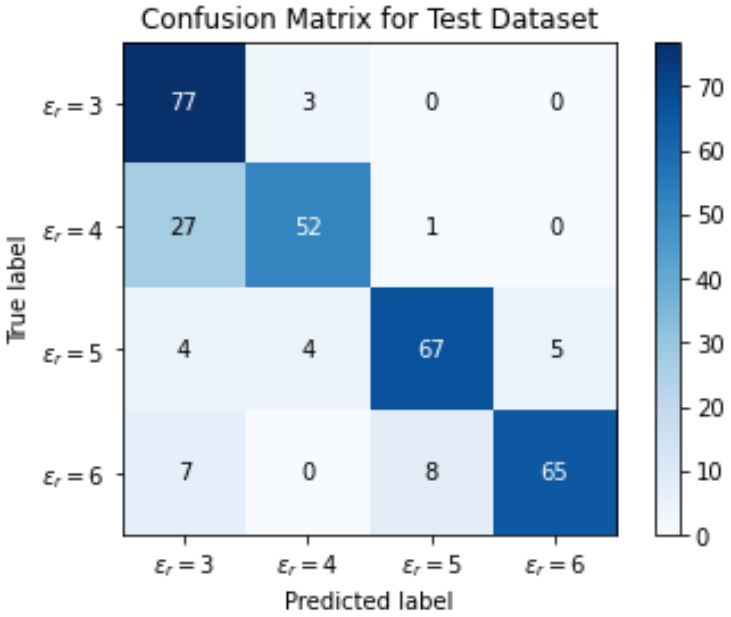


Figure D.1. Confusion matrix of CNN-based dielectric sphere classifier when SNR level of the signal is 0 dB

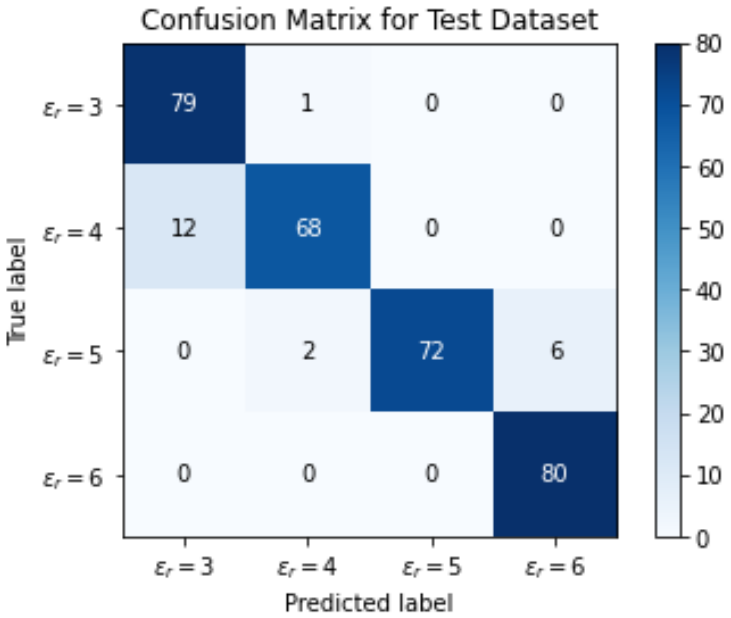


Figure D.2. Confusion matrix of CNN-based dielectric sphere classifier when SNR level of the signal is 5 dB

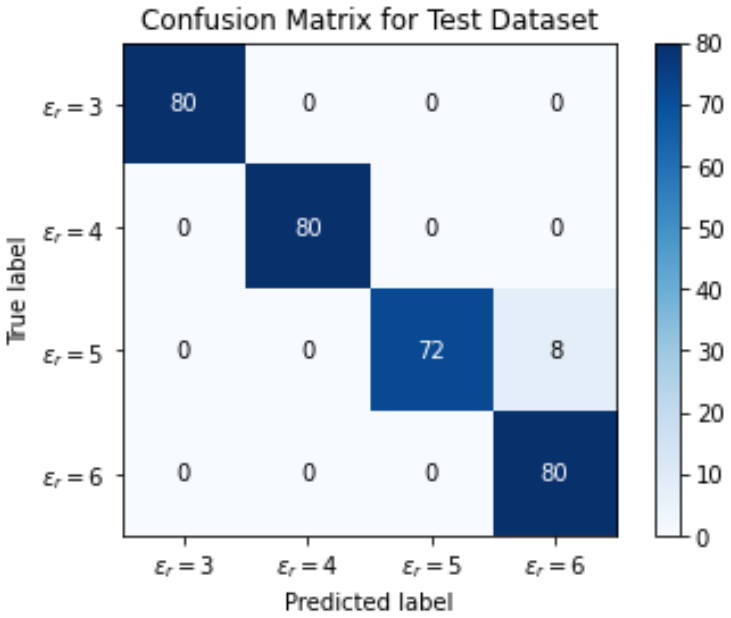


Figure D.3. Confusion matrix of CNN-based dielectric sphere classifier when SNR level of the signal is 10 dB

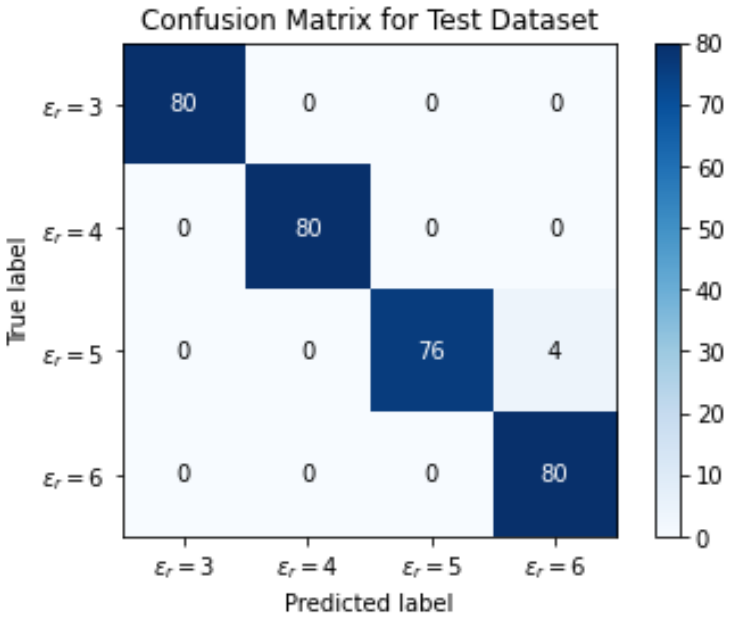


Figure D.4. Confusion matrix of CNN-based dielectric sphere classifier when SNR level of the signal is 15 dB

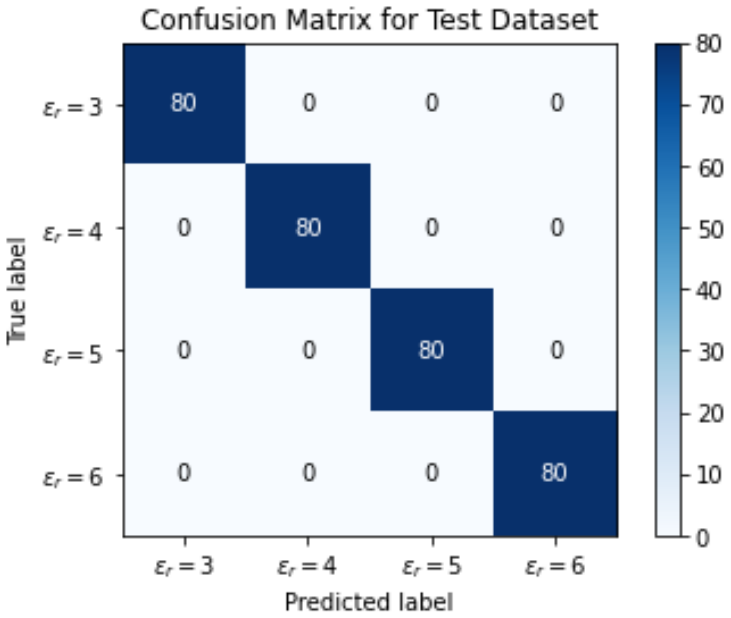


Figure D.5. Confusion matrix of CNN-based dielectric sphere classifier when SNR level of the signal is 20 dB

E. Details of Experimental Setup used for the scattered field measurements of Quadcopter Type UAVs

The UWB Radar Module: "TimeDomain PulsON 440" Radar Kit

TimeDomain PulsON 440 radar kit is used to collect wideband scattered signals from the quadcopters. The device is portable and easily operated with both USB and Ethernet interfaces as shown in Figure E.1. The radar kit operates between 3.1 and 4.8 GHz band as given in Table E.1. The transmitted pulse has a duration of 1 ns, and the characteristics of the pulse could be seen in Figure E.2.

Figure E.1. TimeDomain PulsON 440 Radar Kit [31]

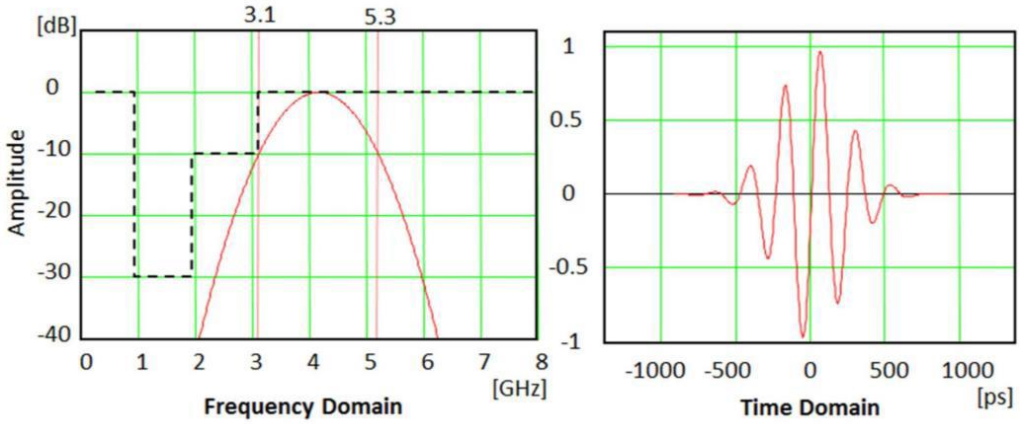


Figure E.2. PulsON 440 UWB pulse plotted in both time and frequency domains [32]

Table E.1 Technical Specifications of PulsON 440 [31]

Property	*Value*
Operating Band	3.1 to 4.8 GHz
Center Frequency	4.3 GHz
Average Transmit Power	-13 dBm
Nominal Pulse Repetition Rate	10 MHz
Receive Noise Figure	4.8 dB

Measurement Interfaces

The communication of the TimeDomain PulsON 440 UWB radar module with a computer could be accomplished over USB or Ethernet interfaces. For both interfaces, MATLAB and C functions are provided by TimeDomain. In addition, the software called Monostatic Radar Module Reconfiguration and Evaluation Tool (MRM RET) is supplied to configure and acquire data from the module from a graphical user interface as illustrated in Figure E.3. We use MRM RET software to gather measurement for the UAV dataset. However, for initial examinations of the radar module, a MATLAB GUI is prepared as a part of this thesis work as shown in Figure E.4. The different configurations and the operational features of the module are evaluated with the prepared GUI.

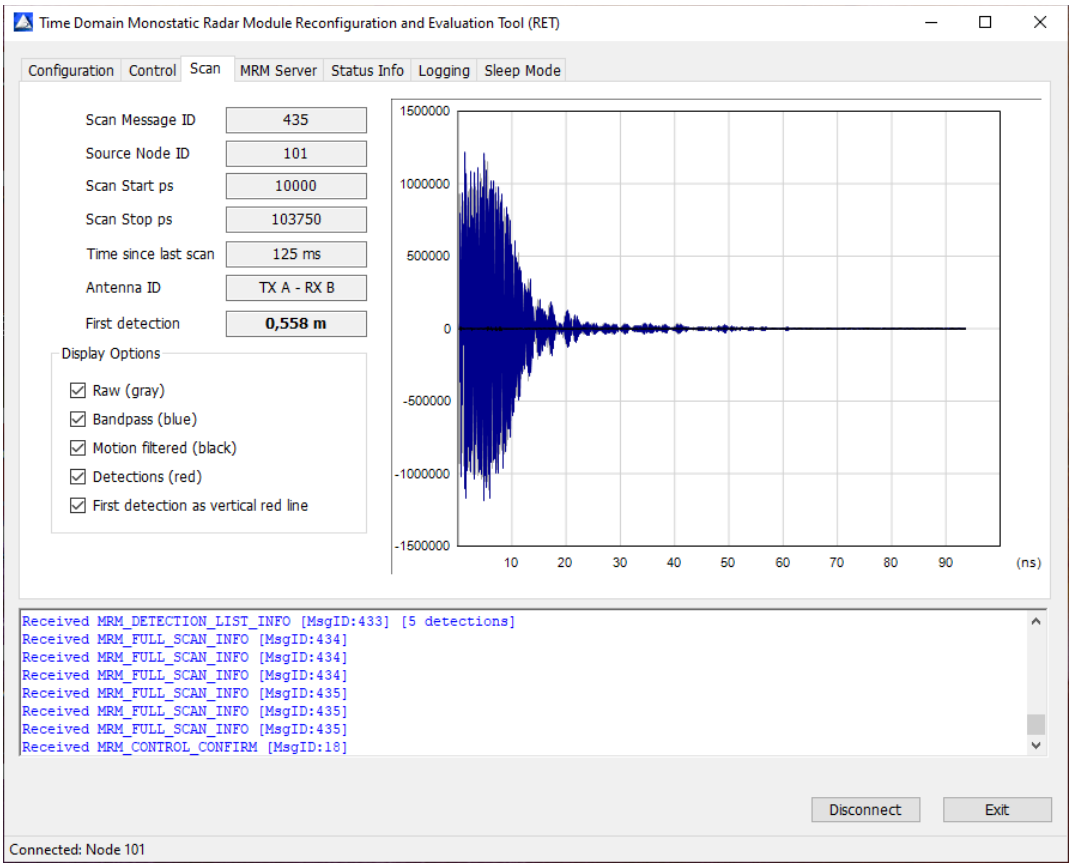


Figure E.3. TimeDomain MRM RET Software

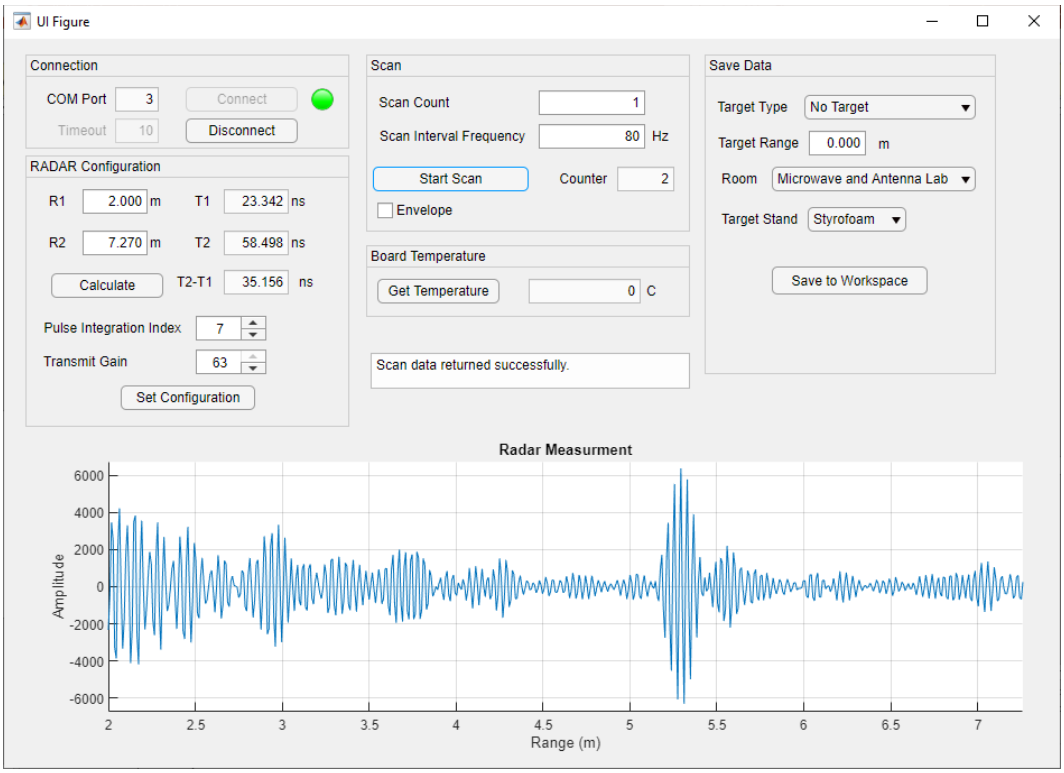


Figure E.4. MATLAB GUI prepared in this thesis work to take measurements by the PulsON440 Radar Kit.

D221 Anechoic Chamber

To measure the wideband scattered response of the quadcopters by avoiding unknown scattered mechanisms as much as possible, the anechoic chamber in the Laboratory D-221 of the Electrical-Electronics Department (METU) is used. This electromagnetically isolated chamber is normally used to measure antenna patterns. The PulsON 440 radar kit is placed at the position normally reserved for the transmitting antenna in the anechoic chamber for this study as illustrated in Figure E.5. The quadcopter type UAV target under test is placed on the opposite side of the anechoic chamber where normally antenna under test (AUT) is positioned. The placement of two quadcopters AR Drone 2.0 and DJI Phantom is shown in Figure D.6 and Figure E.7.

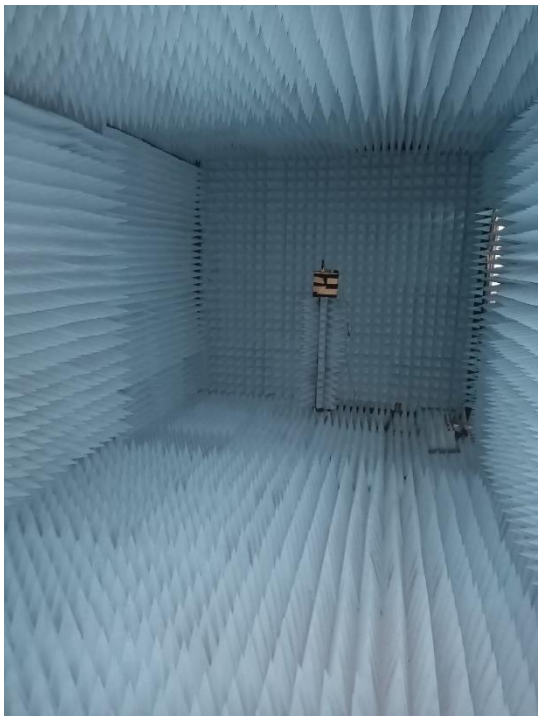


Figure E.5. Placement of PulsON 440 radar kit inside the anechoic chamber

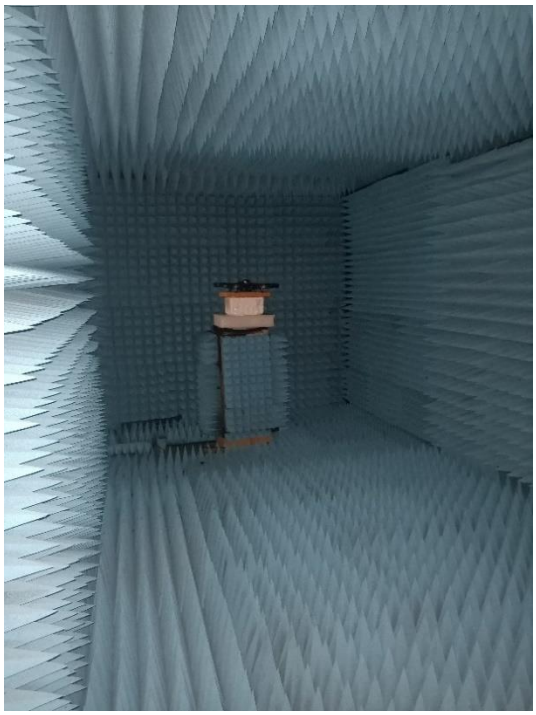


Figure E.6. Placement of AR Drone 2.0 drone inside the anechoic chamber

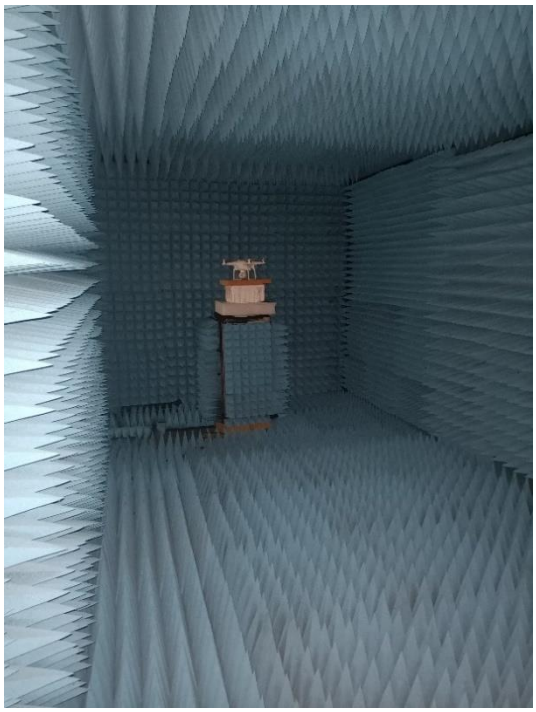


Figure E.7. Placement of DJI Phantom drone inside the anechoic chamber

F. Python Code for Dielectric Sphere Classification

In this Appendix section, the Python code produced for the implementation of the CNN architecture described in section 3.4 is given. Tensorflow framework is utilized to define the neural network layers. Furthermore, all training and testing procedures are performed by using the functions in Tensorflow libraries.

```
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras import optimizers

model = keras.Sequential( [
  layers.Conv1D(filters=16, kernel_size=7, activation="relu",
                        input_shape=(1024,1), kernel_regularizer=('l2')),
  layers.BatchNormalization(),
  layers.Dropout(0.5),
  layers.MaxPooling1D(pool_size=2),
  #-------------------------------------------------------------------------------
  layers.Conv1D(filters=32, kernel_size=7, activation="relu",
                        kernel_regularizer=('l2')),
  layers.BatchNormalization(),
  layers.Dropout(0.5),
  layers.MaxPooling1D(pool_size=2),
 #-------------------------------------------------------------------------------
  layers.Conv1D(filters=64, kernel_size=7, activation="relu",
                        kernel_regularizer=('l2')),
 layers.BatchNormalization(),
 layers.Dropout(0.5),
 layers.MaxPooling1D(pool_size=2),
#-------------------------------------------------------------------------------
```

```
layers.Flatten(),
layers.Dense(64, activation="relu", kernel_regularizer=('l2')),
layers.Dropout(0.5),
layers.Dense(num_classes, activation="softmax")
] )

batch_size = 32
epochs = 20

opt = optimizers.Adam(learning_rate=0.001);

model.compile(loss="categorical_crossentropy", optimizer=opt,
                    metrics=["Accuracy" ,"Precision","Recall"])

model.fit(train_data, train_label, batch_size=batch_size, epochs=epochs,
          validation_data=(test_data, test_label), verbose=1, shuffle=True,
          callbacks=[tensorboard])

model.evaluate(test_data, test_label, verbose=1)
```

G. Python Code for Quadcopter Type UAV Classification

In this Appendix section, the Python code defines classifier for UAV targets is given. The CNN architecture described in section 4.6 is implementing with Tensorflow framework. All training and testing procedures are performed by using the functions in Tensorflow libraries.

```
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras import optimizers

dropout_rate = 0.5
model = keras.Sequential( [
  # Conv-1
  layers.Conv1D(filters=32, kernel_size=5, activation="relu",
                      input_shape=(1024,1), kernel_regularizer=('l2')),
  layers.BatchNormalization(),
  layers.Dropout(dropout_rate),

  layers.Conv1D(filters=32, kernel_size=5, activation="relu",
                      kernel_regularizer=('l2')),
  layers.BatchNormalization(),
  layers.Dropout(dropout_rate),

  layers.MaxPooling1D(pool_size=2),
  #-----------------------------------------------------------------------------
  # Conv-2
  layers.Conv1D(filters=64, kernel_size=5, activation="relu",
                      kernel_regularizer=('l2')),
  layers.BatchNormalization(),
```

```
layers.Dropout(dropout_rate),

layers.Conv1D(filters=64, kernel_size=5,
        activation="relu",kernel_regularizer=('l2')),
layers.BatchNormalization(),
layers.Dropout(dropout_rate),

layers.Conv1D(filters=64, kernel_size=5,
        activation="relu",kernel_regularizer=('l2')),
layers.BatchNormalization(),
layers.Dropout(dropout_rate),

layers.MaxPooling1D(pool_size=2),
#-------------------------------------------------------------------------------
# Conv-3
layers.Conv1D(filters=64, kernel_size=5,
        activation="relu",kernel_regularizer=('l2')),
layers.BatchNormalization(),
layers.Dropout(dropout_rate),

layers.Conv1D(filters=64, kernel_size=5,
        activation="relu",kernel_regularizer=('l2')),
layers.BatchNormalization(),
layers.Dropout(dropout_rate),

layers.Conv1D(filters=64, kernel_size=5,
        activation="relu",kernel_regularizer=('l2')),
layers.BatchNormalization(),
layers.Dropout(dropout_rate),
```

```
layers.MaxPooling1D(pool_size=2),
#-------------------------------------------------------------------------------
# FCL
layers.Flatten(),
layers.Dense(512, activation="relu",kernel_regularizer=('l2')),
layers.Dropout(0.5),
layers.Dense(num_classes, activation="softmax")
] )

batch_size = 128
epochs = 20

boundaries = [(32e3/batch)*10]
values = [0.001, 0.0005]
lr_schedule = optimizers.schedules.PiecewiseConstantDecay(boundaries, values)
opt = optimizers.Adam(learning_rate=lr_schedule);
model.compile(loss="categorical_crossentropy", optimizer=opt,
                    metrics=["Accuracy" ,"Precision","Recall"])

model.fit(train_data, train_label,
          batch_size=batch_size, epochs=epochs,
          validation_data=(test_data, test_label),
          verbose=1, shuffle=True, callbacks=[tensorboard])

model.evaluate(test_data, test_label, verbose=1)
```