

# TARGET DETECTION IN OPTICAL, MICROWAVE AND LIDAR DATA

A DISSERTATION

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE

OF

MASTER OF TECHNOLOGY

IN

CIVIL ENGINEERING

(With Specialization in Geoinformatics Engineering)

By

SUSHMITA GAUTAM

(2K19/GINF/05)

Under the supervision of

DR. (COL) K.C. TIWARI, PROFESSOR



**MULTIDISCIPLINARY CENTRE FOR GEOINFORMATICS  
DEPARTMENT OF CIVIL ENGINEERING  
DELHI TECHNOLOGICAL UNIVERSITY**

(Formerly Delhi College of Engineering)

Bawana Road, Delhi – 110042

November 2021

## CANDIDATE'S DECLARATION

I, Sushmita Gautam, Roll No. - 2K19/GINF/05 student of M. Tech. (Geoinformatics), hereby declare that the Dissertation titled "*Target Detection in Optical, Microwave and LiDAR Data*" which is submitted by me to the Multidisciplinary Centre for Geoinformatics, Department of Civil Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.



SUSHMITA GAUTAM

Place: Delhi

Date: 29- Nov - 2021

## CERTIFICATE

I hereby certify that the Project Dissertation titled, "TARGET DETECTION IN OPTICAL, MICROWAVE AND LIDAR DATA", which is submitted by Sushmita Gautam, 2K19/GINF/05, Multidisciplinary Centre for Geoinformatics, Department of Civil Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is a record of the project work carried out by the student under my supervision. To the best of my knowledge, this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place: Delhi

Date: 29 Nov 21



SUPERVISOR

Dr. (Col) K.C. TIWARI, Professor  
Multidisciplinary Centre for Geoinformatics  
Department of Civil Engineering  
Delhi Technological University  
Delhi - 110042

## **ACKNOWLEDGEMENTS**

First and foremost, I express my deep sense of gratitude to my supervisor, counselor and advisor **Prof. K.C. Tiwari**, Multidisciplinary Centre for Geoinformatics, Department of Civil Engineering for their constant guidance, support, motivation and encouragement throughout the period this work was carried out. Their readiness for consultation at all times, educative comments, concern and assistance have been invaluable.

I also want to thank **Dr. Shalini Gakhar** (Research Associate, IARI) and **Dr. Deepti Soni** (IIT Roorkee alumni) for their support and all the staff of the Multidisciplinary Centre for Geoinformatics, Civil Engineering Department for their fullest cooperation.

I would like to thank my friends and all those who have directly or indirectly helped me in completion of the thesis well in time.

Finally, I wish to thank my parents for their moral support and confidence showed in me to pursue M. Tech at an advanced stage of my academic career.

## **ABSTRACT**

Historically, in the security-defence environment, information is derived through a subjective analytical approach principally based on the experience and the skills of the analyst who visually interprets the image(s). The spatial and contextual way to proceed varies and depends on the objective of the study. Spatial, pattern, texture, and, in general, spectral information is most of the time improved by standard image processing technics (i.e., image enhancement) for increasing the visual distinction between features. Different collateral/ancillary data, spatially and temporally correlated with the imagery, made available through different sources, may complement the analytical process providing worthwhile information, essential in helping, confirming, etc. the interpretation course and its inferences.

For target detection in remotely sensed images, targets can be referred to as man-made or natural object or an event or activity of interest. Detection of an object or activity, such as military vehicles or vehicle tracks, is common in both military and civilian applications. Hyperspectral target identification algorithms look at the spectrum of each pixel to find targets based on the spectral characteristics of the target's surface material. Targets of interest may not be clearly resolved depending on the sensor's spatial resolution, hence the first fundamental characteristic of the hyperspectral target detection problem is that a "target present" versus "target absent" decision must be made individually for each pixel of a hyperspectral image.

In this project, the various target detection algorithms along with the dimensionality reduction methods are explored and presented with valid results and discussions. The work involves the use of multi-platform, multi-dimensional and multi-sensor datasets making this work an important example of the various target detection approaches in remote sensing.

# CONTENTS

<b>Candidates Declaration</b>	i
<b>Certificate</b>	ii
<b>Acknowledgments</b>	iii
<b>Abstract</b>	iv
<b>Contents</b>	vi
<b>List of Figures</b>	ix
<b>List of Tables</b>	xi
<b>List of Abbreviations</b>	xi
1. Introduction.....	1
1.1. Motivation.....	1
1.2. Background to Remote Sensing.....	1
1.3. Approaches in Target Detection.....	2
1.4. Issues in Target Detection.....	2
1.5. Research Gaps.....	2
1.6. Research Objectives.....	3
1.7. Organization of Thesis.....	3
2. Detection of Camouflaged Targets.....	5
2.1. Introduction.....	5
2.2. Experimental Data and Software.....	5
2.2.1. HSI Subsets.....	6
2.2.2. Description of Targets.....	7
2.2.3. Software.....	10
2.3. Methodology.....	11
2.3.1. Georeferencing Image.....	11
2.3.2. Atmospheric Correction.....	12
2.3.3. Dimensionality reduction.....	12
2.3.3.1. PCA.....	12
2.3.3.2. ICA.....	14
2.4. Target Detection Algorithms.....	14

2.4.1. Adaptive Coherence Estimator.....	14
2.4.2. Matched Filter.....	15
2.4.3. Spectral Angle Mapper.....	15
2.5. Implementation.....	15
2.6. Results and Discussions.....	16
2.7. Summary.....	20
3. Identification of Camouflaged Target using Lidar data.....	21
3.1. Introduction.....	21
3.2. Experimental Data and Software.....	21
3.2.1. Subset images.....	21
3.2.2. Description of Targets.....	22
3.2.3. Software.....	22
3.3. Methodology.....	22
3.4. Implementation.....	23
3.5. Results and Discussions.....	24
3.6. Summary.....	26
4. Comparative Assessment of Spectral Target Detection Algorithms.....	27
4.1. Introduction.....	27
4.2. ROC.....	28
4.3. Data and Software.....	28
4.4. Methodology.....	28
4.5. Implementation.....	29
4.6. Results and Discussions.....	29
4.7. Summary.....	32
5. Target object detection in Optical and SAR datasets using Deep Learning Methods.....	33
5.1. Introduction.....	33
5.2. Dataset and Software.....	34
2.2.2. Description of Targets.....	36
2.2.3. Software.....	36
5.3. Methodology.....	36
5.4. Implementation.....	37

5.5. Results and Discussions.....	37
5.6. Summary.....	40
6. Conclusion and Future Scope.....	41
6.1. Conclusion.....	41
6.2. Future Scope.....	42
References	43
Appendix 1: Data Collection	46
Appendix 2: Source Code	47



## LIST OF FIGURES

Figure 1: RIT campus showing all flight lines .....	6
Figure 2: HSI subset 1 .....	7
Figure 3: Green Camo Net target on asphalt road .....	8
Figure 4: Tan camo net target on asphalt .....	9
Figure 5: Green camo tarps targets on asphalt .....	9
Figure 6: A target having set of three chairs .....	10
Figure 7: Flow of methodology used.....	11
Figure 8: subset of full image containing target .....	22
Figure 9: A target having set of three chairs covered with CAMO1 net .....	22
Figure 10: Flow of methodology for target identification .....	23
Figure 11: LiDAR 3D viewer for the scene.....	23
Figure 12: Spectral signature of the tan camouflaged n.....	23
Figure 13: ROC curve results of ICA based ACE algorithm .....	23
Figure 14: ROC curve results of ICA based SAM algorithm .....	25
Figure 15: ROC curve results of ICA based MF algorithm .....	27
Figure 16: ROC curve results of ICA based ACE algorithm .....	28
Figure 17: ROC curve for SAM detection .....	29
Figure 18: ROC curve results of PCA based MF detection .....	30
Figure 19: Ship class image patches .....	34
Figure 20: No- ship class image patches .....	34
Figure 21: Nine sub-images from the SAR dataset .....	35
Figure 22: The CNN results of optical satellite dataset .....	38
Figure 23: CNN Detection results for SAR satellite dataset .....	39

## **LIST OF TABLES**

Table 1: Target considered.....	7
Table 2: Detection of green camo tarps.....	16
Table 3: Detection of tan camo net targets.....	17
Table 4: Detection of green camo net targets.....	17
Table 5: Statistics for detection of camouflaged targets material.....	18
Table 6: Statistics for detection of camouflaged targets material in case of green tarp.....	18
Table 7: Statistics for detection of camouflaged targets material in case of green camo after PCA.....	18
Table 8: Statistics for detection of camouflaged targets material in case of green camo net after ICA.....	19
Table 9: Statistics for detection of camouflaged targets material in case of tan camo net after ICA.....	19
Table 10: Statistics for detection of camouflaged targets material in case of green camo tarp after ICA.....	19
Table 11: Dataset information per image sensor and polarization.....	34

## **LIST OF ABBREVIATIONS**

- I. **HSI** – Hyperspectral Image
- II. **LiDAR** – Light Detection And Ranging
- III. **PCA** – Principle Component Analysis
- IV. **ICA** – Independent Component Analysis
- V. **ACE** – Adaptive Coherence Estimator
- VI. **SAM** – Spectral Angle Mapper
- VII. **MF** – Matched Filter
- VIII. **ROC**- Receiver Operating Characteristics
- IX. **SAR** – Synthetic Aperture Radar
- X. **CNN** – Convolutional Neural Networks

# CHAPTER 1

## INTRODUCTION

---

### 1.1 MOTIVATION

The target detection in remote sensing has been widely studied from many years and is of great interest in many applications. Vital applications such as mineral mapping (for example, exploration and of epithermal ores deposits or recognizing hydrothermally altered rocks) [1], agriculture applications (for example, use of spectral signatures of vegetation and detection of disease and nutrient deficiency allowing monitoring, tracking and controlling of crops health) [2], law enforcement (for example, application of thermography to detect several illegal activities) [3], strategic surveillance and military applications (for example, detection of military objects such as vehicles, weapons, land and sea mines, camouflaged objects using multiple information extracted from hyperspectral imagery) [4] etc. involves the use of target detection and remote sensing.

Target detection algorithms are capable of detecting man-made targets in uncluttered natural background and have been fairly well established [5]. Many of these simply do not work with difficult targets such as camouflaged targets, occluded targets, dim targets without sufficient *a priori* information [6]. In case of camouflaged targets, detection based on spectral information often fails, hence height, shape and surface information may be needed in combination with the spectral information to support detection and identification of such targets [7].

Therefore, with different challenges in target detection and with development of large number of algorithms, the development of approaches for detection and identification of targets will continue to work on similar challenges.

### 1.2 BACKGROUND TO REMOTE SENSING

Remote sensing, also called earth observation, refers in a general sense to the instrumentation, techniques and methods used to observe, or sense, the surface of the earth, usually by the formation of an image in a position, stationary or mobile, at a certain distance remote from that surface. In remote sensing, electromagnetic radiation from an object is measured and converted into information about the object or processes connected to the item

(in the instance of earth observation, this object is the earth's surface).

Electromagnetic radiation can be transmitted, absorbed, or reflected when it hits an object on the earth's surface. The object's attributes define the mutual magnitude of these processes. We can quantify the amount of reflected solar radiation as a function of wavelength in remote sensing, which is known as spectral reflectance.

Passive sensors don't have their own radiation source. They are only sensitive to natural radiation, such as reflected sunlight or energy generated by an earthly object. Active sensors have a built-in radiation source. Radar (radio detection and ranging) and lidar (light detection and ranging) are two examples (light detection and ranging).

### **1.3 APPROACHES IN TARGET DETECTION**

Target detection algorithms based on spectral modelling can be categorized in two groups namely spectral matching and anomaly detection algorithms. Spectral matching algorithms are based on matching of every pixel spectrum with a priori available information. Examples of some spectral matching algorithms are Spectral Angle Mapper (SAM), Constraint Energy Maximization (CEM), Matched Filter (MF), Adaptive Cosine estimator (ACE). While the anomaly detection algorithms compare the pixel spectrum with the background spectrum and declared a target based on noticing certain abrupt spectral variations. Some examples of anomaly detection algorithms are Reed- Xialoi (RX) and UTM algorithms, etc.

### **1.4 ISSUES IN TARGET DETECTION**

Target detection is considered challenging due to following issues:

- i. Size of the target is relatively small with respect to the spatial resolution of the image which creates complexity in its detection.
- ii. Spectral variability, that refers to the variations in measured spectra of different samples of the same material.
- iii. Illumination variation often occurs in natural scenes which effects the spectral properties of the target.
- iv. A target is occluded under other object or vegetation.
- v. A target is camouflaged purposely.
- vi. High dimensionality resulting in high complexity and redundancy.

### **1.5 RESEARCH GAPS**

The research gaps found in this study are:

- i. Optical camouflage used for protection of military equipment confuse the optical detection and surveillance of enemy making it look like the background. Hence, conventional target detection methods, based on spectral separation of background and target, generally fails to detect a camouflaging target.
- ii. Identification in addition to the detection is the demand of the problem which may require spatial information of the target such as shape etc. Multi sensor data can solve the needed.
- iii. Some dimensionality reduction algorithms proposing to reduce the size of data may not consider the importance of target in the band rejected. This results in losing important information. Thus, an approach that reduces redundancy but retains the target information may need to be further explored.
- iv. Target detection is used in social security management, marine vehicle traffic monitoring, etc. There are very outstanding application results in the fields of early warning and national defense security. For deep learning models like Convolutional Neural Networks, computer vision algorithms for marine target recognition may need to be explored.

## **1.6 RESEACH OBJECTIVES**

The objective of this research are as follows:

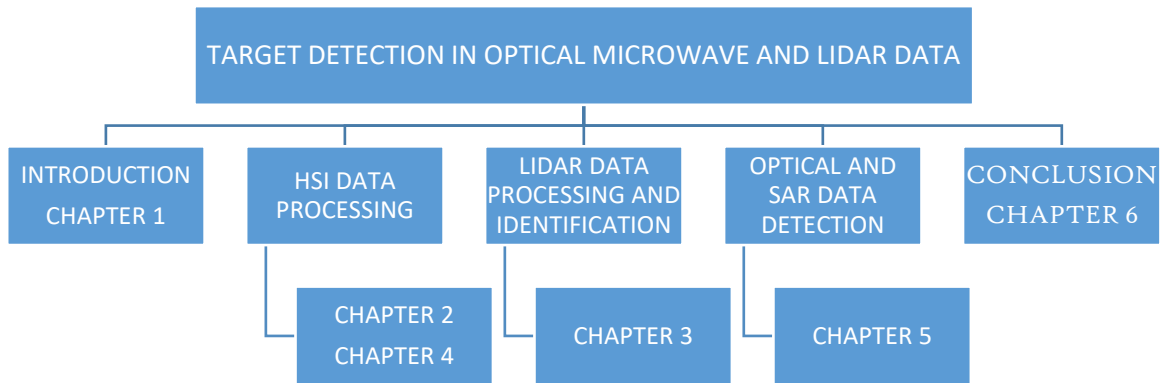
- i. To detect camouflaged targets using different dimensionality reduction and target detection algorithms.
- ii. To identify camouflaged targets using Lidar data
- iii. To perform Comparative assessment of various spectral target detection algorithms after dimensionality reduction.
- iv. To detect target object in marine Optical and SAR satellite datasets using deep learning

## **1.7 ORGANIZATION OF THESIS**

- i. Chapter 1 introduces the motivation leading to the problem formulation for this study and the background to remote sensing and target detection approaches. Next the research gaps and research objectives has been listed.
- ii. Chapter 2 provides the introduction, experimental methods and implementation description for the camouflaged target detection problem in Hyperspectral dataset and the resultant detection maps for different algorithms and discussion

is presented in the chapter.

- iii. Chapter 3 introduces to the LiDAR processing after the HSI data has been explored. Hence, adding to the identification of the camouflaging target.
- iv. Chapter 4 introduces to the use of receiver operating characteristics curve in doing the comparative assessment of various target detection algorithms used.
- v. Chapter 5 is the application of deep learning technique in detecting marine ship target using the optical and SAR satellite datasets.
- vi. Chapter 6 provides the conclusion and the future scope for the various target detection approaches used in this project thesis.



## *CHAPTER 2*

# *DETECTION OF CAMOUFLAGED TARGETS*

---

### **2.1 INTRODUCTION**

It is possible to extract information about the scene from the data after the pre-processing algorithm is applied on the data. This chapter focuses on dimensionality reduction and target detection algorithms. Once again, hyperspectral target detection is the process of locating desired pixels in a scene based on the spectral characteristics of the target and pixel. In this chapter, several different algorithms are introduced and explained.

### **2.2 EXPERIMENTAL DATA AND SOFTWARE**

This experiment focuses on detection of camouflaged targets by detecting the camouflaging material. This experiment particularly uses two subsets of the hyperspectral data acquired over RIT campus described in section 2.2.1. below. Figure 1(a) shows the full image of the RIT campus scene.

The study area comprising of the campus of Rochester Institute of Technology (RIT), New York, USA. Datasets are captured in three different flight lines flown over the campus namely RIT\_1, RIT\_2 and RIT\_3 comprising several buildings of different sizes, roads, trees, parking areas, etc., which usually describes as an urban environment. In addition, several artificial targets have been deployed at various locations within the campus.

The dataset has been collected as part of a data collection campaign titled as SpecTIR Hyperspectral Airborne Rochester Experiment (SHARE), conducted by RIT in conjunction with SpecTIR, LLC, during July 26-29, 2010. Related reference data is available in the campaign dataset and includes detailed information about each target deployed. It includes contextual images, field and laboratory spectra, G.P.S. data and ortho images as a part of ground data.

The data consists of 360 bands with 5 nm spectral resolution and 1 m spatial resolution. For each flight line, Internal Geometry Map (IGM) and Geographic Lookup Table (GLT) have also been provided for creating georeferenced hyperspectral image.





(a)



(b)

Fig 1 (a) RIT campus showing all flight lines of the SHARE campaign 2010. (b) Flight line\_3 used in this experiment.

### 2.2.1. HSI SUBSETS

The airborne Hyperspectral data was captured using ProSpecTIR-VS2 hyperspectral imaging sensor provided by SpecTIR, LLC, to collect data in spectral range from Visible to SWIR (short wave infrared) i.e., 390 to 2450 nm. This work uses RIT\_3 flight line as shown in Figure 1(b).

Two spatial subsets covering the desired targets have been extracted from the original image. The first subset consists of five camouflaging materials, namely green camouflaging

net (GREEN CAMO), tan camouflaging net (TAN CAMO) and three instances of green camouflaging tarp (GREEN TARP1, GREEN TARP2, GREEN TARP3). The second subset contains a camouflaged target which is a set of three chairs (CAM\_CHAIR) placed under a camouflaging net (CAMO1). Refer figure 2 for the subset 1 image.

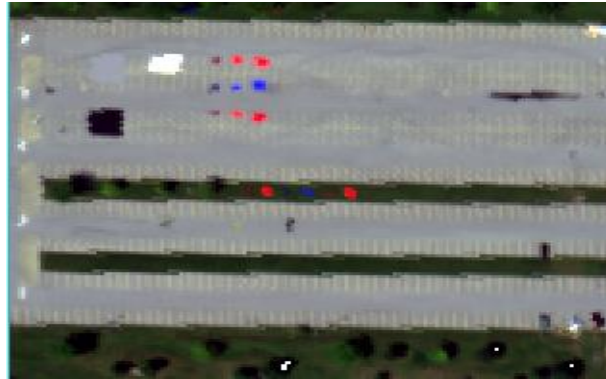


Fig 2. HSI subset 1 (QUAC atmospherically corrected).

## 2.2.2 DESCRIPTION OF TARGETS

Several artificial targets have been placed in study area at multiple locations. The details of each of the target and its reference data have been described in this section. Each target considered has been assigned a unique ID. Refer Table 1 summarizing details of the targets considered in this experiment.

**Table 1: Target considered**

Serial Number	Target Name	Number of Instances and ID	Target Characteristics
1.	Green Camouflaging Net	1 (GREEN CAMO)	(i) A camouflaging material with light and dark colored woodland camouflaged pattern placed on asphalt. (ii) Full pixel (iii) Present in HSI data (iv) Location 43°5'17.64"N, 77°40'40.80" W
2.	Tan Camouflaging Net	1 (TAN CAMO)	(i) A desert tan green colored camouflaging

			material placed on asphalt. (ii) Full pixel (iii) Present in HSI data (iv) Location 43°5'17.61"N, 77°40'41.39" W
3.	Green Camouflaging Tarp	3 (GREEN TARP1, GREEN TARP2, GREEN TARP3)	(i) A green tarpaulin sheet with camouflaging pattern placed on asphalt (ii) Full pixel (iii) Present in HSI data (iv) Location of GREEN TARP1 43°5'17.62"N, 77°40'42.23" W
4.	Camouflaging Chairs	1 (CAM_CHAIR, CAMO1)	(i) Three chairs covered with camouflaging material Placed on grass (ii) Full pixel (iii) Present in HSI data (iv) Location 43°5'9.5"N, 77°40'38.2" W

### 1. Green Camouflaging Net

A thin camouflaging material with light and dark green colored woodland camouflaged pattern mounted on black mesh backing material has been used. It has been placed on asphalt. The field photograph of this target is shown in Figure 3.



Fig 3. Green Camo Net target on asphalt road

## 2. Tan Camouflaging Net

A desert tan green colored camouflaging material mounted on black mesh backing material placed on asphalt has been used. The field photograph of this target is shown in Figure 4.



Fig 4. Tan camo net target on asphalt

## 3. Green Camouflaging Tarps

A green camouflaging tarpaulin sheet with camouflaging pattern on one side and plain green on other side has been used. Three instances of different size has been placed on asphalt. The field photograph of this target is shown in Figure 5.



Fig 5. Green camo tarps targets on asphalt

#### 4. Camouflaging Chairs

Set of three tall purple colored chairs (CAM\_CHAIR) made up of plastic have been placed at known locations. These chairs have been covered with camouflaging net (CAMO1). field photograph of this target is shown in Figure 6.



Fig 6. A target having set of three chairs covered with CAMO1 net

### 2.2.3. SOFTWARE

For implementation of approach developed for detection of targets, ENVI software version 5.3 developed by L3HARRIS Geospatial has been used for visualization, processing and analyzing remote sensing data. This software is used to carry out the following work:

- i. Pre-processing of hyperspectral data such as georeferencing, atmospheric correction, spectral and spatial subsetting etc.
- ii. Viewing HSI image data.
- iii. Target detection etc.

### 2.3. METHODOLOGY

The flow of methodology has been given below in Figure 7. The detailed terms of methodology used are described in further sections.

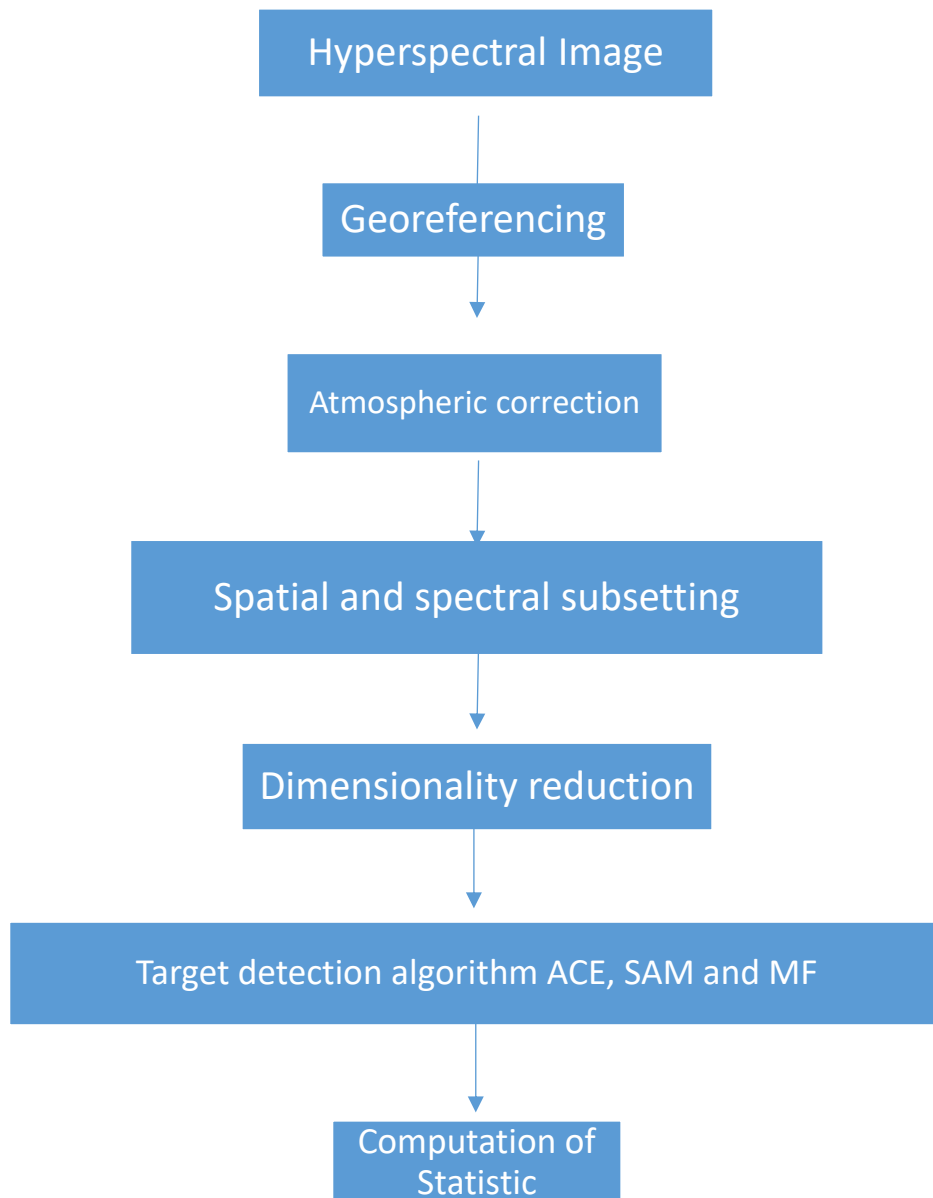


Fig 7: Flow of methodology used.

### **2.3.1. GEOREFERENCING**

The original image has been georeferenced using the geographic lookup table (GLT) provided along with the data.

### **2.3.2. ATMOSPHERIC CORRECTION**

QUick Atmospheric Correction (QUAC) method is used in this experiment. QUAC is a multispectral and hyperspectral atmospheric correction approach that uses the visible and near-infrared through shortwave infrared (VNIR-SWIR) wavelength range. If the following

requirements are met, QUAC can get reasonably accurate reflectance spectra:

- i. A scene contains at least ten different materials.
- ii. There are enough dark pixels in an image to allow for a good baseline spectrum estimation.

### 2.3.3. DIMENSIONALITY REDUCTION

The goal of dimensionality reduction is to reduce the size of a data set while maintaining as much of the information content as possible. Large image files can contain low signal to noise ratio (SNR) bands as well as redundant information due to spectral similarities. Dimensionality reduction algorithms eliminate these redundancies in data based on a variety of different factors. The three algorithms described here utilize different aspects of eigenvector and eigenvalue calculations to represent the data in rotated data spaces.

#### 2.3.3.1. PCA

The Principle Component Analysis (PCA) method of dimensionality reduction attempts to decorrelate the data and maximize the data content in fewer dimensions [Schott 97, Johnson 02]. This is accomplished by finding a different axis to project the data onto that will maximize the variance. The pixel vector in principal component analysis is illustrated in Figure 2.7.

An image pixel vector can be expressed as:

$$\mathbf{x}_i = [x_1, x_2, \dots, x_N]_i^T \quad (1)$$

where,

$x_1, x_2, \dots, x_N$  = Pixel values of Hyperspectral image at corresponding pixel location.

$N$  = The number of the hyperspectral bands equivalent to dimension of the pixel vector.

The mean vector for all image vectors is calculated as:

$$\mathbf{m} = \frac{1}{M} \sum_{i=1}^M [x_1 \quad x_2 \quad \dots \quad x_N]_i^T \quad (2)$$

The covariance matrix of  $\mathbf{x}$  is defined as:

$$\mathbf{Cov}(\mathbf{x}) = \mathbf{E}\{(\mathbf{x}-\mathbf{E}(\mathbf{x}))(\mathbf{x}-\mathbf{E}(\mathbf{x}))^T\} \quad (3)$$

where,

$\mathbf{E}$  = expectation operator;

T = transpose operation; and

Cov = notation for covariance matrix.

The PCA is based on the eigenvalue decomposition of the covariance matrix, which takes the following form:

$$Cx = MDM^T \quad (4)$$

Where  $D = \text{diag}(\lambda_1, \lambda_2 \dots \lambda_N)$  is the diagonal matrix composed of the eigenvalues  $\lambda_1, \lambda_2 \dots \lambda_N$  of the covariance matrix  $C_x$ , and  $M$  is the orthonormal matrix composed of the corresponding  $N$  dimension eigenvectors.

The linear transformation defined by:

$$y_i = M^T x_i \quad (I = 1, 2, \dots, K) \quad (5)$$

is the PCA pixel vector, and all these pixel vectors form the PCA (transformed) bands of the original images. the first  $K$  PCA bands contain majority of information residing in the original hyperspectral images. These can be used for more effective analyses since the number of bands and the amount of image noise involved are reduced.

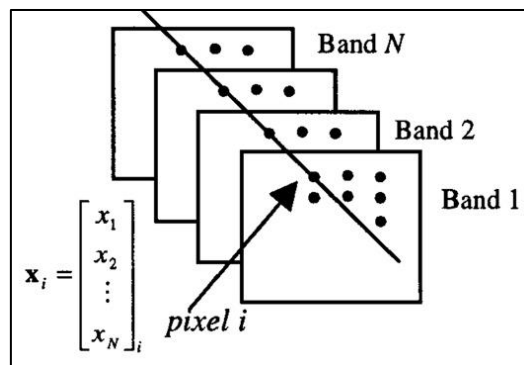


Fig 8. Pixel vector in principal component analysis [adapted from Gonzales and Woods (1993)].

### 2.3.3.2. ICA

The goal of ICA is to find a linear representation of non-Gaussian data so that the components are statistically independent or as independent as possible. In the context of target detection, various bands of hyperspectral image can be considered as linear mixtures composed of  $n$  independent components. These components may consist of both the desired targets and redundant information in the form of undesired targets and noise. Mathematically, any given



mixture says the  $j$ th mixture in this case can be represented as:

$$x_j = a_{j1}S_1 + a_{j2}S_2 \dots a_{jK}S_K \quad (6)$$

Rewriting eq. (6) in matrix notation as:

$$x = As \quad (7)$$

Eq. (7) is also known as the ICA model. Denoting the columns of A, the ICA model can be rewritten as  $A_j$

$$x = \sum_{i=1}^n a_i s_i \quad (8)$$

The ICA aims to find components which are statistically independent meaning thereby that the resulting components will have non-Gaussian distribution.

## 2.4. TARGET DETECTION ALGORITHMS

In this experiment, the target detection is performed by implementing three spectral matching based target detection algorithms namely Adaptive Coherence Estimator (ACE), Matched Filter (MF), Spectral Angle Mapper (SAM).

### 2.4.1. ADAPTIVE COHERENCE ESTIMATOR

The Adaptive Coherence Estimator (ACE) can be derived from the GLRT through some assumptions.

$$\hat{\Sigma}_b = \frac{1}{N} \Sigma_b \quad (9)$$

Where where  $\Sigma^b$  represents the covariance matrix of the background. N represents the number of pixels in the set of sample data. This in turn alters the GLRT equation as

$$\hat{T}_{GLRT}(x) = \frac{(d^T \hat{\Sigma}_b^{-1} x)^2}{(d^T \hat{\Sigma}_b^{-1} d) \left(1 + \frac{1}{N} x^T \hat{\Sigma}_b^{-1} x\right)} \quad (10)$$

Since N is small, GLRT is the ACE algorithm and is written as:

$$T_{ACE}(x) = \frac{(d^T \hat{\Sigma}_b^{-1} x)^2}{(d^T \hat{\Sigma}_b^{-1} d)(x^T \hat{\Sigma}_b^{-1} x)} \quad (11)$$

## 2.4.2. MATCHED FILTER

Matched Filter algorithm finds the abundance of targets using a partial unmixing algorithm. This technique maximizes the response of the known spectra and suppresses the response of the composite unknown background, therefore matching the known signature. It provides a rapid means of detecting specific materials based on matches to target spectra and does not require knowledge of all the endmembers within an image scene.

## 2.4.3. SPECTRAL ANGLE MAPPER

The Spectral Angle Mapper (SAM), developed by Boardman, is a simple detection algorithm that computes the angle between two vectors. According to the SAM algorithm, pixel A is more target like than pixel B due to the smaller angle between it and the target vector. The SAM algorithm is expressed in vector form as

$$T_{SAM}(x) = \frac{d^T x}{(d^T d)^{1/2}(x^T x)^{1/2}} \quad (12)$$

where  $d$  represents the target spectra and  $x$  represents the pixel of interest. This equation will have values between 0 and 1 for a reflectance data set. Similar pixels will have a value near 1 which corresponds to a small angle between the vectors in question. SAM is a simple and fast detection technique with low computing costs. It is only affected by the object's spectral profile and is unaffected by magnitude changes between the target and pixel.

## 2.5. IMPLEMENTATION

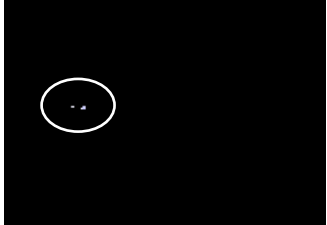
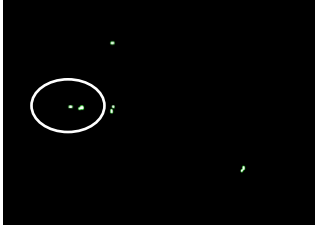
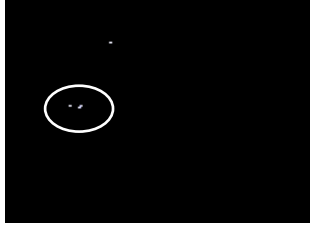
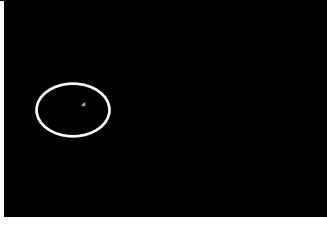
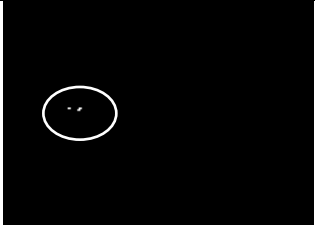
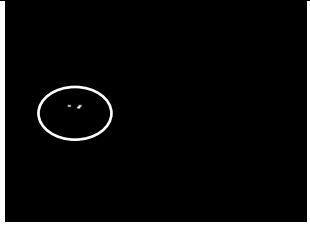


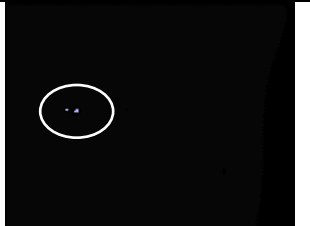
All the implementations for the above discussed algorithms were done using ENVI 5.3 as summarized below:

- i. For each target, reference spectra have been drawn from the image itself and is referred to as in scene spectra.
- ii. PCA and ICA has been applied on Subset 1 and Subset 2. For both subsets, first 100 components have been selected using 360 bands in original image.
- iii. Detection using spectral matching algorithm namely ACE, MF and SAM has been performed
- iv. The results obtained has been thresholded to generate binary images depicting the presence or absence of target.

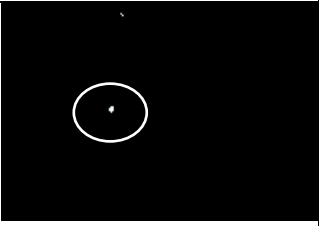
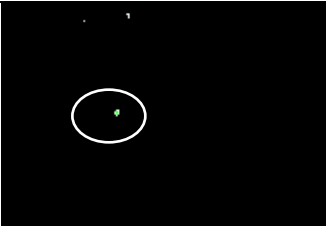
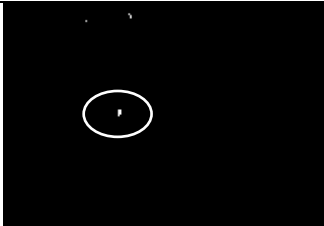
## 2.6. RESULTS AND DISCUSSIONS

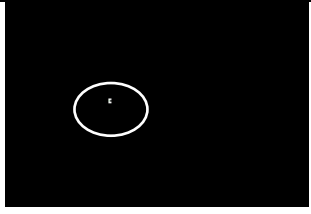
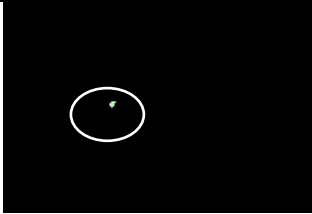
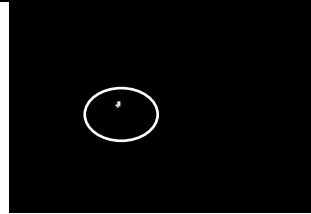
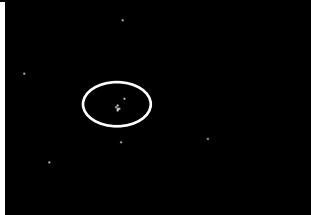

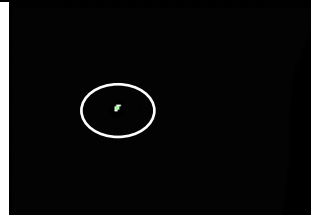
ICA yielded 360 components from which 100 components were retained. All the 100 components were used to detect all 3 camouflaged targets using the above three algorithms, ACE, SAM and MF. The results obtained after applying is shown below.

**Table 2: Detection of green camo tarps**

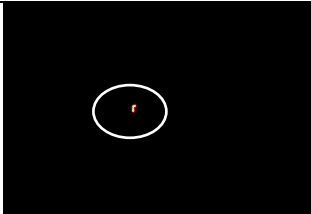
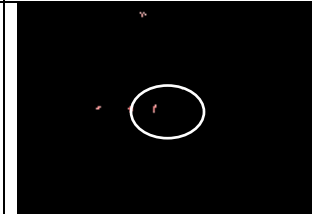
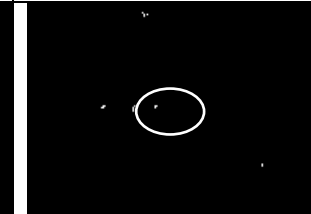
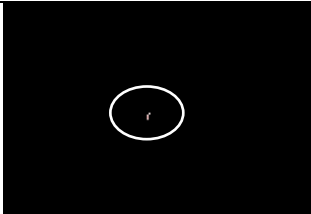
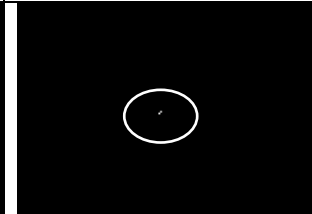
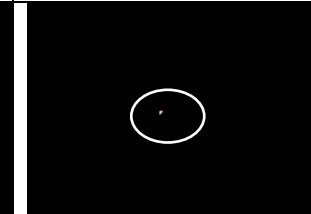
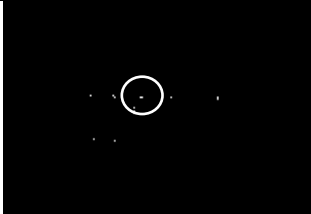

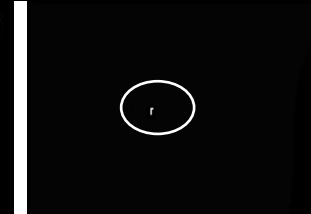
ALGORITHM M	ORIGINAL SUBSET	PCA	ICA
MF			
ACE			
SAM			

**Table 3: Detection of tan camo net targets**

ALGORITHM M	ORIGINAL SUBSET	PCA	ICA
MF			

ACE			
SAM			

**Table 4: Detection of green camo net targets**

ALGORITHM	ORIGINAL SUBSET	PCA	ICA
M			
MF			
ACE			
SAM			

**Table 5: Statistics for detection of camouflaged targets material in case of tan  
camo after PCA on subset**

Algorithm	threshold	Target number of pixels	Detected number pixels
MF	0.70	14	13
ACE	0.545	10	8
SAM	0.216	15	9

**Table 6: Statistics for detection of camouflaged targets material in case of green  
tarp after PCA**

Algorithm	threshold	Target number of pixels	Detected number pixels
MF	0.430	15	14
ACE	0.746	6	2
SAM	0.233	30	18

**Table 7: Statistics for detection of camouflaged targets material in case of green  
camo net after PCA**

Algorithm	threshold	Target number of pixels	Detected number pixels
MF	0.800	24	20
ACE	0.686	2	2
SAM	0.250	33	32

**Table 8: Statistics for detection of camouflaged targets material in case of green  
camo net after ICA**

Algorithm	threshold	Target number of pixels	Detected number pixels
MF	0.908	19	18

ACE	0.554	3	3
SAM	0.850	5	5

**Table 9: Statistics for detection of camouflaged targets material in case of tan camo net after ICA**

Algorithm	threshold	Target number of pixels	Detected number pixels
MF	0.783	11	10
ACE	0.560	8	8
SAM	0.850	11	11

**Table 10: Statistics for detection of camouflaged targets material in case of green camo tarp after ICA**

Algorithm	threshold	Target number of pixels	Detected number pixels
MF	0.686	8	8
ACE	0.650	6	2
SAM	0.754	7	7

Now, the following can be observed from the figures and Tables:

- Table 2 to Table 4 shows the targets in original image subset, PCA applied subset and the ICA applied subset contains camouflaging material as encircled. These figures are results of the three detection algorithms i.e., MF, ACE and SAM. For the thresholds, it was observed that in case of SAM, as the threshold is lower, higher probability of finding the targets.
- Based on the outputs figures in these Tables, and statistics in the Table 5 to Table 10, the following observations can be made:
  - i. Table 5 shows that MF resulted in 13 pixels out of 14 were correctly detected. While in ACE and SAM the number of pixels correctly detected was low in case of PCA.
  - ii. Table 6 and 7 shows MF has resulted sufficient number of correct pixels as compared to ACE and SAM after the thresholds were applied.
  - iii. Table 8 to 10 improves the correct detection results in case of ICA.

From the above observations, following may be concluded

- MF has performed better in case of all the camouflaging targets for PCA and SAM has performed worst.
- MF performance result giving target pixels are much better after ICA was implemented for the original subset.

## **2.7 SUMMARY**

Camouflaging is the process of merging the target with background. This is done for reducing or delaying detection of the target. This experiment used three target different target detection algorithms before and after 2 different dimensionality reduction approaches. Results demonstrated that the detection of camouflaging material can be performed using PCA and ICA approach followed by detection algorithms. Matched Filter algorithm works well. However, identification of targets is also important for which LiDAR data after the HSI processing in this chapter, is used in the next chapter.

**CHAPTER 3**

**IDENTIFICATION OF CAMOUFLAGED TARGET USING  
LIDAR DATA**

---

### **3.1. INTRODUCTION**

Besides HSI data, various other types of remote sensing data such as multispectral, radar and Lidar data, etc, have also been used in camouflaged breaking. Literature suggests that specialized LiDAR sensing systems may be designed for military purposes such as target positioned under trees, camouflaged target etc. Titterton in 2015 performed detection of artificial targets using laser scanning system. Combined usage of data from different sensors HSI and Lidar for detection of camouflaged targets is explored in this chapter.

### **3.2. EXPERIMENTAL DATA AND SOFTWARE**

The hyperspectral data used in earlier chapter detected the targets, while in this chapter LiDAR data has been used for detection and identification of underlying camouflaged target.

#### **3.2.1. SUBSET IMAGE**

Here, in this experiment the second subset, Subset 2 of the HSI image is used and shown below.



(a)





(b)

Fig 8. (a) full image (b) subset of full image containing target.

### 3.2.2. DESCRIPTION OF TARGET

Set of three tall purple colored chairs (CAM\_CHAIR) made up of plastic have been placed at known locations. These chairs have been covered with camouflaging net (CAMO1). field photograph of this target is shown in Figure 9.



Fig 9. A target having set of three chairs covered with CAMO1 net

### 3.2.3. SOFTWARE

The software ENVI 5.3 and ENVI LIDAR 5.3 developed by L3HARRIS Geospatial is used in this experiment.

### 3.3 METHODOLOGY

The flow of methodology is shown below:

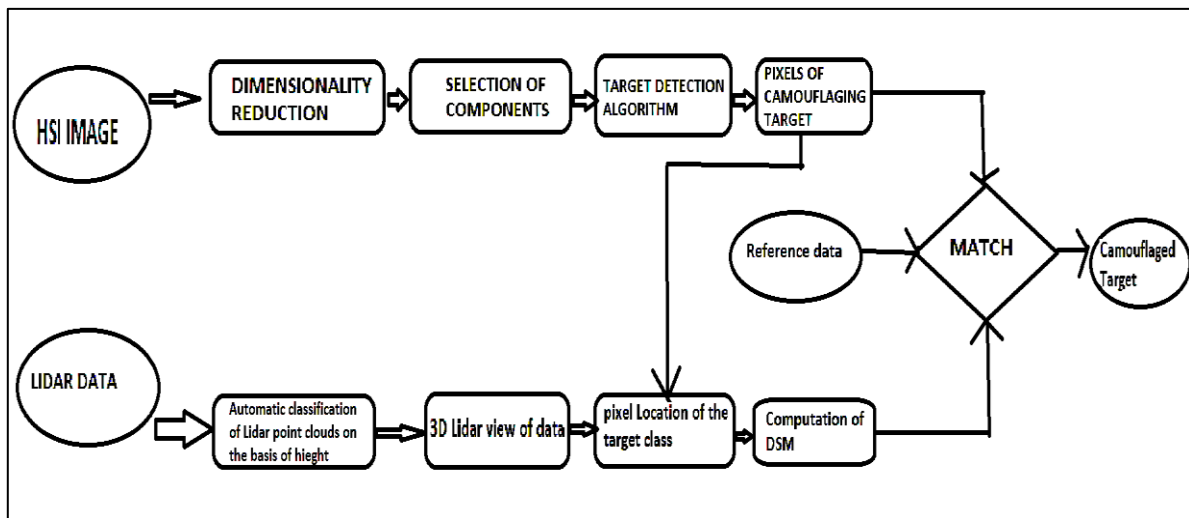


Fig 10. Flow of methodology for target identification using LIDAR data.

### 3.4 IMPLEMENTATION

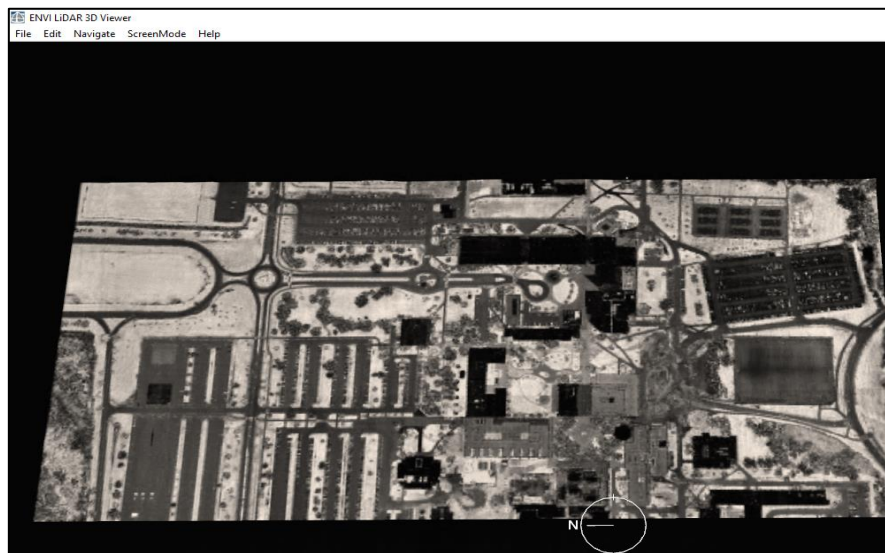
Detection and identification of camouflaged target (set of three chairs) has been performed using LiDAR point clouds. The steps of implementation are described below:

- i. The HSI subset 2 is used to conduct the target detection, the pixel location is used as seed location for LiDAR interpretation.
- ii. Automatic point cloud classification is done on the basis of height.
- iii. A 3D LiDAR view is created for the data.
- iv. Pixel location of the target from HSI detection is pointed on the 3D LiDAR view.
- v. A digital surface model is created to confirm the target surface property and finally identify the set of three chair underlying the camouflaged net target.
- vi. Reference data such as contextual images provided in dataset at the site of target is use to confirm the presence of target in the image data of the RIT campus area.

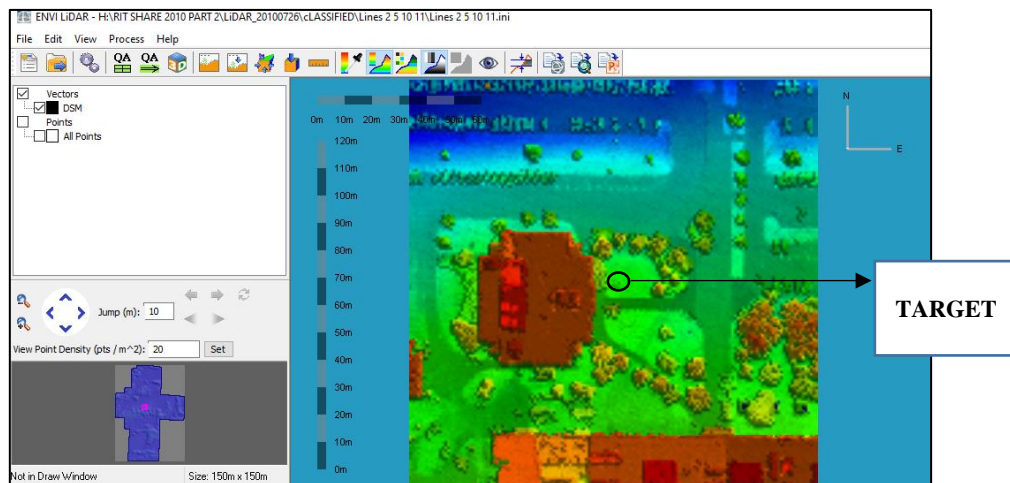
### 3.5. RESULTS AND DISCUSSIONS

Results of LiDAR data processing:

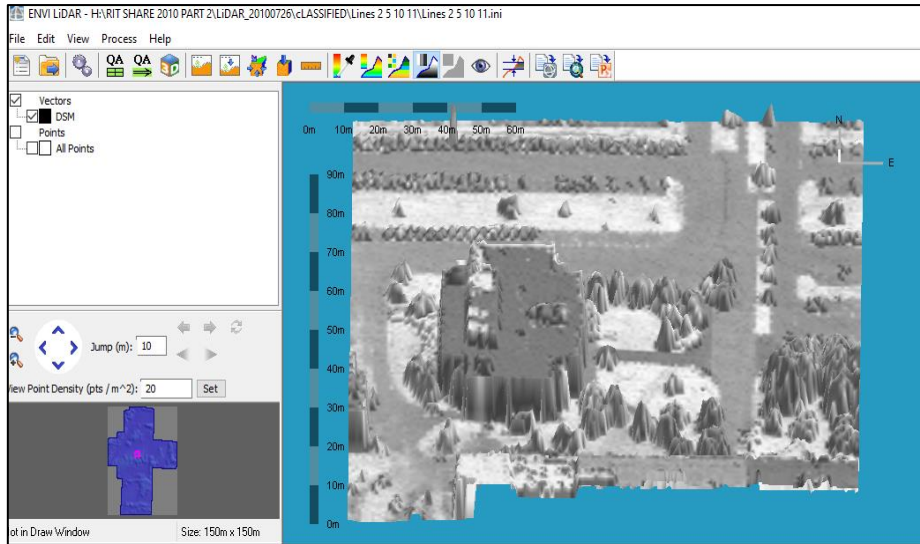
The 3D view generated and the DSM created for the target is shown below. The DSM shows 3 peaks of valid chair height of the target is visible in the figure. The location of the target is matched with the HSI detection results as well as the reference contextual images.



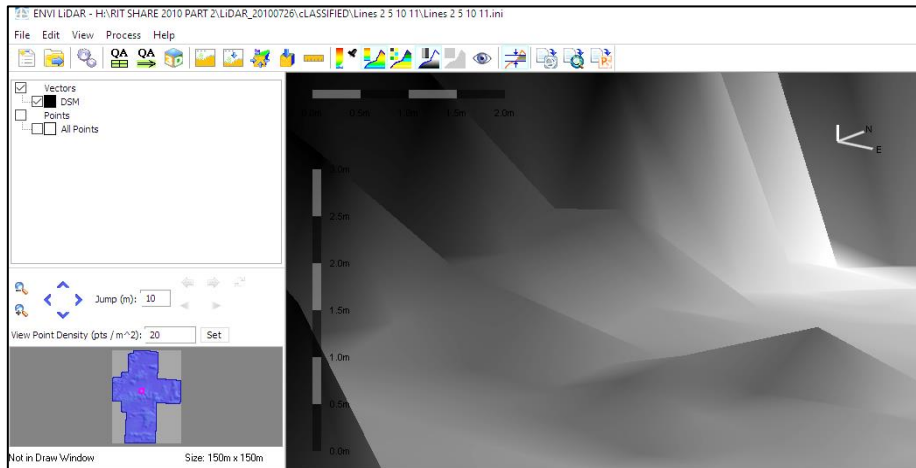
(a)



(b)



(c)



(d)

Figure 11. (a) LiDAR 3D viewer for the scene. (b) Subset scene near the target. (c) and (d) DSM generated for the scene and the target respectively.

From the above figures, the following observations and conclusion can be made.

- The Digital Surface Model shows in figure (d) the three peaks indicating the presence of three chairs target that were covered with the camouflaged net.
- The contextual images as reference confirmed that the target location associated with the its surroundings
- Hence, it can be concluded that the LiDAR point cloud, DSMs and the reference data can aid in identification of the target to be detected.

### **3.6 SUMMARY**

The results demonstrated that the detection on camouflaging material can be performed using various Target detection algorithm and the identification part can be performed using the LiDAR data with the combination of reference data. Furthermore, the results obtained by processing LiDAR point demonstrate the use of DSM of the potential target at the seed locations generated using the HSI processing.

**CHAPTER 4**

**COMPARATIVE ASSESSMENT OF SPECTRAL TARGET  
DETECTION ALGORITHMS**

---

## 4.1 INTRODUCTION

A receiver operator characteristic (ROC) curve is a standard approach for assessing detection results. The false alarm rate (FAR) vs the detection rate (DR) plot for a particular target is known as an ROC curve. A log FAR axis can be used to expand this figure and highlight differences at low FAR values. When employing log ROC curves, a minimum FAR value must be set because the minimum cannot be zero. These curves can be used for performance analysis on their own, or metrics can be calculated from them. The DR at a certain FAR is one such measure that may be taken from a ROC curve. This threshold metric reduces the ROC curve to a single point on the curve specified by the user. This is typically a low FAR on the order of  $10^{-3}$  or  $10^{-4}$ .

## 4.2 ROC

Four outcomes of the binary hypothesis test are possible.

- i. TT: True target labeled as a target (correct detection).
- ii. TB: True target labeled as background (missed detection).
- iii. BT: True background labeled as target (false alarm).
- iv. BB: True background labeled as background (correct non detection).

The ROC curve describes detection performance with  $P_D$  plotted as a function of  $P_{FA}$ . When applied to a known empirical situation (with ground truth), these probabilities are estimated by the maximum-likelihood estimates.

$$\hat{P}_D = \frac{\text{Number of observed true detections } (N_{TT})}{\text{Number of possible targets } (N_{TT} + N_{TB})} \quad (13)$$

$$\hat{P}_{FA} = \frac{\text{Number of observed false detections } (N_{BT})}{\text{Number of background pixels } (N_{BT} + N_{BB})} \quad (14)$$

The ROC curve is created by shifting the threshold across the test statistic's range, with

points on the curve calculated by counting the number of true and false detections at each threshold and computing (13) and (14) at each threshold. When the test statistic for the target and background samples overlaps, increasing the number of true detections will inevitably result in an increase in the number of false alarms. This tradeoff is represented by the ROC curve.

The genuine positive rate (Sensitivity) is presented as a function of the false positive rate (100-Specificity) at various cut-off points in a Receiver Operating Characteristic (ROC) curve. A sensitivity/specificity pair corresponding to a specific decision threshold is represented by each point on the ROC curve. The ROC curve of a test with perfect discrimination (no overlap between the two distributions) passes through the upper left corner (100 percent sensitivity, 100 percent specificity). As a result, the higher the overall accuracy of the test, the closer the ROC curve is to the upper left corner.

### **4.3 DATA AND SOFTWARE**

This chapter uses the data of the worked upon in chapter 2 and presents the valid ROC curve analysis for the various algorithm used in the chapter 1, i.e., MF, ACE and SAM preferably for the tan camouflaged target.

All the implementations are performed using the ENVI version 5.3 and the MATLAB r2021a software.

### **4.4 METHODOLOGY**

The methods are described in below steps:

- i. The spectra from the image for the tan camouflage target is saved with the header file using ENVI.
- ii. The atmospherically corrected subset 1 image is input in MATLAB. The implementation in MATLAB is depended on the image processing toolbox and Hyperspectral image processing toolbox of the MATLAB version.
- iii. The code, refer appendix section, is ran for PCA and ICA pre-processing of the subset image and for further applying MF, ACE and SAM on the same subset separately for PCA and ICA results and ROC curves for each are generated.

### **4.5 IMPLEMENTATION**

A source code in the appendix section is attached at last of this thesis that can be referred for the implementation part of this chapter. The code is an implementation that uses MATLAB programming and Image Processing Toolbox add on. The purpose of the implementation is to generate ROC curves for the mentioned target detection algorithm in the methodology.

## 4.6. RESULTS AND DISCUSSIONS

The spectra extracted from the image subset is saved and then imported in MATLAB. The visualization of the spectra for tan camouflaging net is shown below.

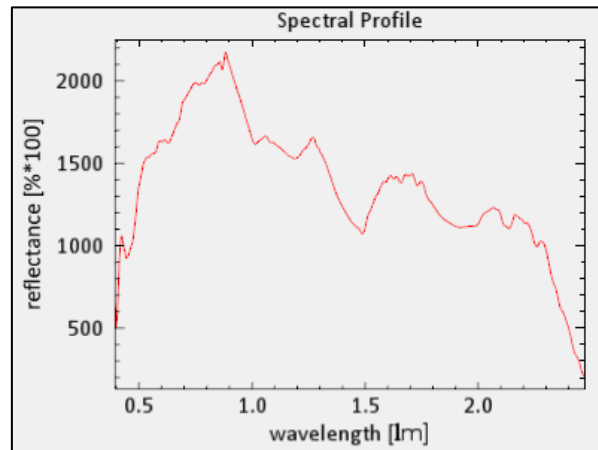


Fig 12. Spectral signature of the tan camouflaged net target.

Next the various resultant ROC curves after the dimensionality reduction (PCA and ICA) and the target detection algorithm (MF, SAM and ACE) are shown next.

### 4.6.1 ICA based Target detection ROC curves

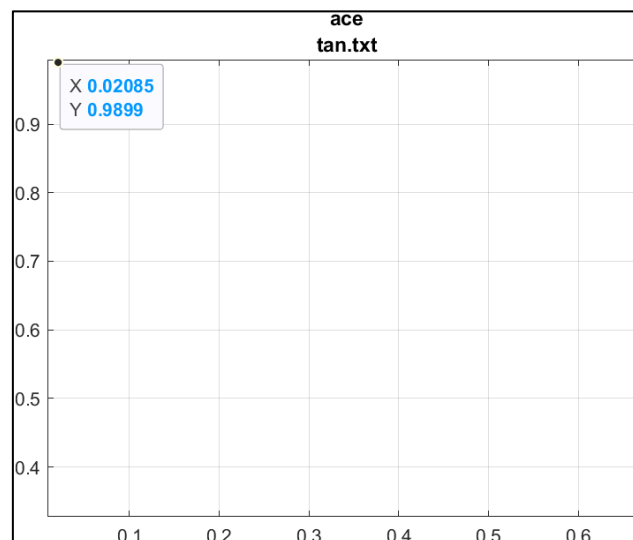


Fig. 13. ROC curve results of ICA based ACE algorithm



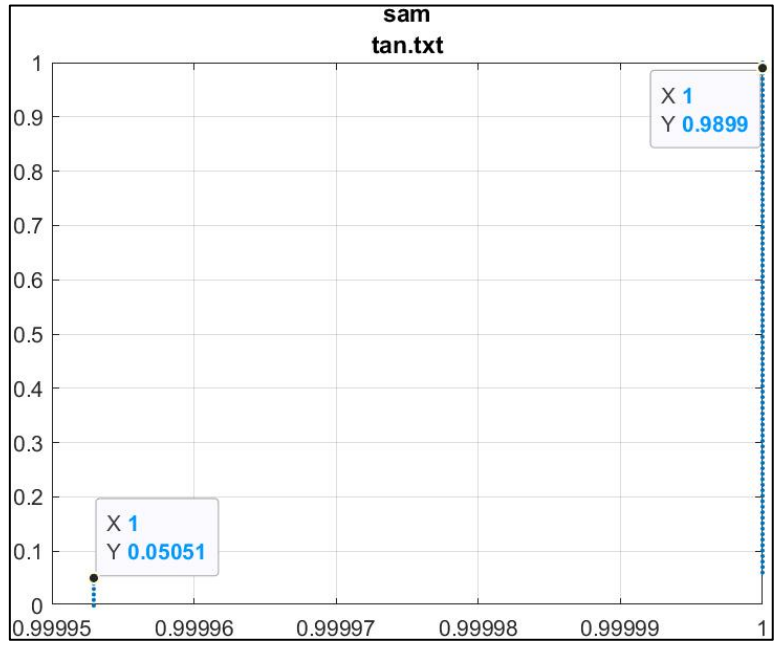


Fig. 14. ROC curve results of ICA based SAM algorithm

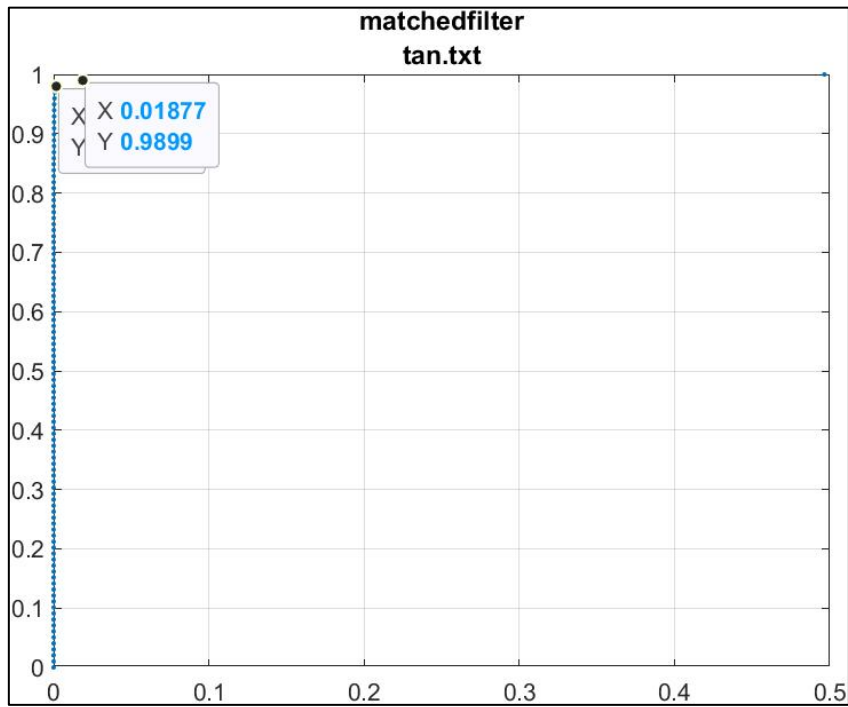


Fig. 15. ROC curve results of ICA based MF algorithm

### 4.6.2. PCA based Target Detection ROC curves

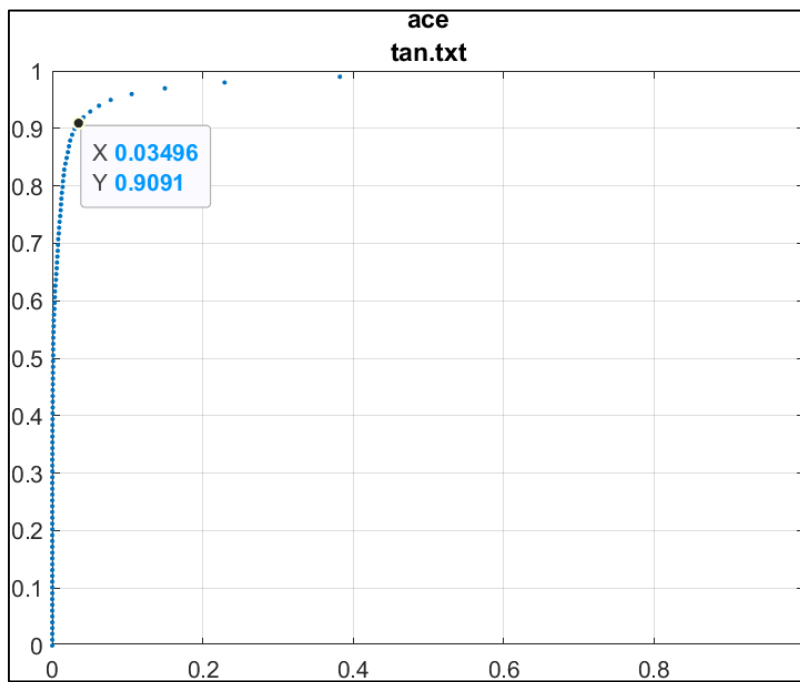


Fig. 16. ROC curve results of ICA based ACE algorithm

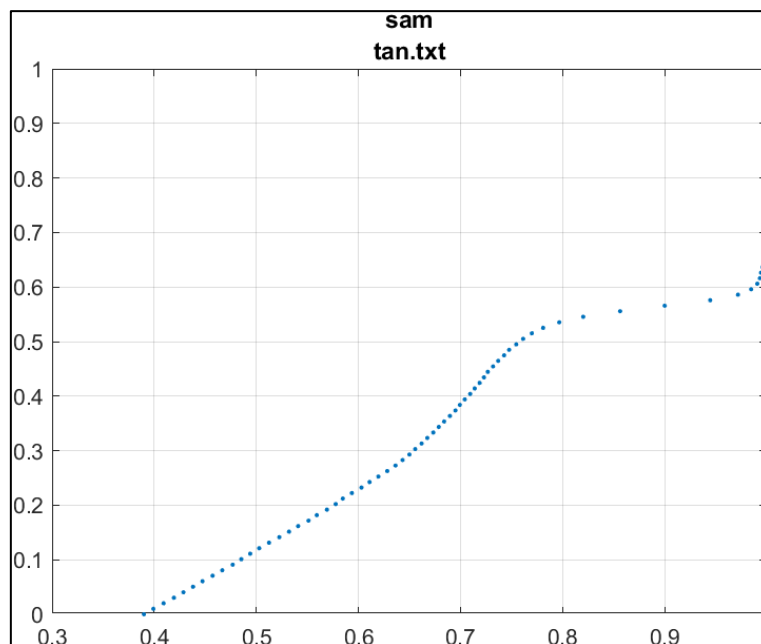


Fig. 17. ROC curve for SAM detection

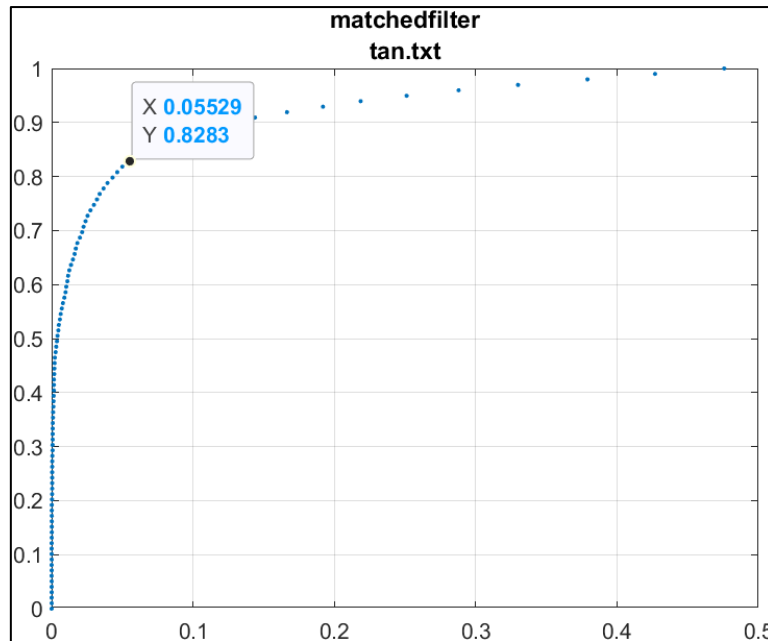


Fig. 18. ROC curve results of PCA based MF detection

Following is observed from the ROC curves:

- From the above discussions about the ROC curve analysis, it can be seen in the results that the ICA based Matched Filter (MF) algorithm has performed best in the detection of the camouflaged target.
- Secondly, the ICA based ACE algorithm has shown second best algorithm in detection of camouflaged target. While the PCA and the ICA based SAM (spectral angle mapper) algorithm has performed badly in detection of the camouflaged target in the experiment.

#### 4.7. SUMMARY

This experiment focuses on comparing the performance of target detection algorithms on the basis and inclusion of two dimensionality reduction algorithm with the RIT SHARE 2010 hyperspectral dataset. From the results it can be seen that the ICA based ACE and MF algorithms has performed well as compared to the PCA based ACE and MF results of ROCs.

# **TARGET OBJECT DETECTION IN OPTICAL AND SAR DATASETS USING DEEP LEARNING METHODS**

---

## **5.1 INTRODUCTION**

Computer vision technology has been integrated into all aspects of life in the continuous development of today's society. Target detection is a very basic but very important task in computer vision technology. Target detection is used in social security management, traffic vehicle monitoring, environmental pollution detection, and forest disasters. There are very outstanding application results in the fields of early warning and national defense security. The task of target detection mainly includes the recognition and location of single or multiple targets of interest in digital images. Target detection algorithms have been studied for many years. In the 1990s, many effective traditional target detection algorithms appeared. They mainly used traditional feature extraction algorithms to extract features and then combined with template matching algorithms or classifiers for target recognition. However, traditional algorithms have encountered a bottleneck in their development due to the lack of strong semantic information and complex calculations. In 2014, Ross Girshick proposed a convolutional network-based target detection model RCNN with high detection accuracy and strong specific robustness and generalization ability, making people pay more attention to the use of convolutional neural networks to extract high-level semantic information of images and many excellent detection models of convolutional neural networks have been proposed.

## **5.2. DATASETS AND SOFTWARE**

Image chips from Planet satellite imagery collected over the San Francisco Bay and San Pedro Bay areas of California make up dataset 1. It contains 4000 80x80 RGB photos with a classification of "ship" or "no-ship." PlanetScope full-frame visual scene products, orthorectified to a 3 m pixel scale, were used to create image chips.

Shipsnet.json, a JSON-formatted text file, is also included with the dataset. Data, label, scene ids, and location lists are all included in the loaded object.

- i. "label": Valued 1 or 0, representing the "ship" class and "no-ship" class, respectively.
- ii. "longitude\_latitude": The longitude and latitude coordinates of the image center point, with values separated by a single underscore.

The dataset 2 which was obtained from IEEE Dataport, is "A SAR SHIP DATASET FOR DETECTION, DISCRIMINATION AND ANALYSIS". The dataset contains 43 Sentinel-1 Extended Wide Swath images and three RADARSAT-2 ScanSAR Narrow images from October 6, 2014, to July 22, 2015. Using multiple polarizations and three different resolutions, these photos cover a substantial portion of the South African Exclusive Economic Zone. Synthetic Aperture Radar is a surveillance system that is particularly well suited to marine surveillance. Large swath widths, time-independent observations, and weather resistance can be particularly beneficial for detecting ships that are generally unseen to conventional means of monitoring. The entire generation process from the initial compressed data to the geo - coded images is described in detail, as well as the dataset organizational structure tailored to each phase of the ship detection process, ship referencing and attribute extraction practices, additional Automatic Identification System (AIS) transponder information and matching, and ship reference and Automatic Identification System matched ship attribute analysis.

TABLE 11: Dataset information per image sensor and polarization.

Attributes	Sentinel-1		RADARSAT-2
	GRDH	GRDM	SCNA
Acquisitions	6	16	3
Images Total	12	31	3
Type	EW	EW	SGF
Incidence Angle (°)	19.0 – 47.0	19.0 – 47.0	20.0 – 39.0
Swath Width (km)	400	400	300
Resolution [rg x az] (m)	50 × 50	93 × 97	81 × 30
Pixel Spacing [rg x az] (m)	25 × 25	40 × 40	25 × 25
Number of Looks [rg x az]	3 × 1	6 × 2	2 × 2

### 5.2.1. DESCRIPTION OF TARGETS

#### Class Labels of Dataset 1:

There are 1000 photos in the "ship" class. Images in this category are near-centered on a single ship's body. There are ships of various sizes, orientations, and atmospheric collection conditions. Below are some examples of photographs from this lesson.



Fig 19. Ship class image patches of the optical satellite dataset.

There are 3000 photos in the "no-ship" class. A third of them are random samples of various landcover features, such as water, flora, bare dirt, buildings, and so on, that do not include any part of a ship. The third category is "partial ships," which contain only a fraction of a ship but not enough to match the "ship" class's full definition.

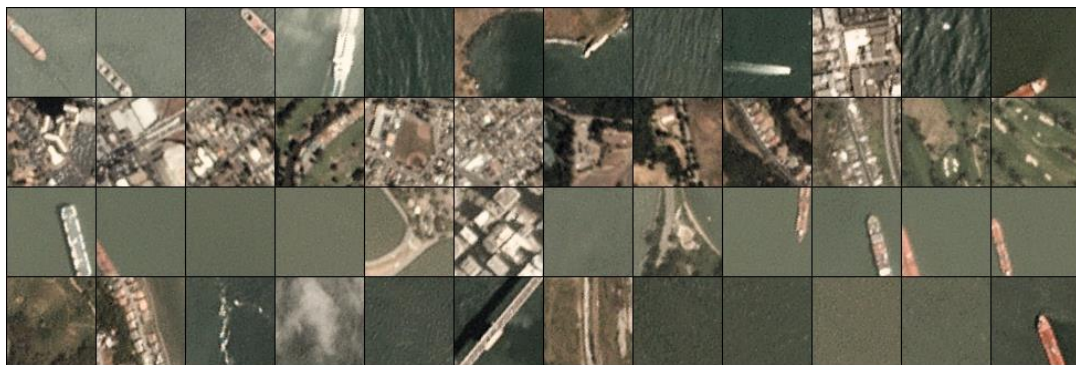


Fig 20. No- ship class image patches of the optical satellite dataset.

### Dataset 2:

A ship is defined as any item that is sufficiently brighter than its surrounding ocean backscatter in SAR intensity imagery covering the ocean. A ship was presumed to be an area of ocean with significantly higher backscatter than its neighbours and is at least 2 pixels in length for the sake of simplicity and due to the resolution of the SAR imagery for this dataset. Ships with dimensions less than these are outside the scope of this dataset and would be better analysed with higher resolution SAR imagery. As a result, single pixels with strong backscatter due to speckle noise are ignored by this definition.

There are four images associated with each ship in the dataset: a "ship patch," a "reference patch," a "ship sub-image," and a "reference sub-image." Patches are big pictures centred on the ship that cover a large region of pixel (101 x 101). Sub-images are smaller images (21 x 21) with little to no information 5 other than the ships in the image's centre. A reference image is a binary image that shows "true" for pixels related with the ship in the centre and "false" for

all other pixels. The sizes of the reference photos were selected to match the SAR ship patches and sub-images (101 x 101 and 21 x 21 respectively).

The Automatic Identification System (AIS) is a transponder system that is installed aboard ships and sends their positions at regular intervals. Ship tracks can be formed by tracking these places over time. AIS messages are received in one of two ways: by terrestrial AIS receivers on the coast or inland satellite-based receivers. Coastal-based AIS receivers have a range of around 74 kilometres from the coast and receive messages in near real time, but satellite-based AIS communications are received every 6-12 hours and have a near-global range. The SAR dataset includes roughly 220 million AIS signals obtained from the 6th of October 2014 to the 22nd of July 2015, and it covers the full of South Africa's EEZ.

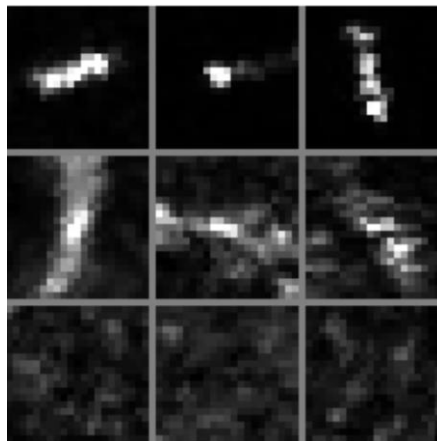


Fig. 21. Nine sub-images from the SAR dataset

### 5.2.2. SOFTWARE

All the programming regarding this experiment were performed on Google Colab. Colaboratory, or "Colab" for short, allows one to write and execute Python in browser.

## 5.3. METHODOLOGY

Convolutional neural network target detection also has certain similarities with traditional detection, which can be regarded as feature extraction and use of features to identify targets. Both backbone feature extraction network and "detection head".

Convolutional neural networks use convolutional networks to extract high-level semantic features of images, which are generally the backbone of the network, and then process the feature maps, such as connecting a fully connected network and softmax to form a classification head to complete the classification task, or use a small volume The product core is processed into feature dimensions and a position loss function is used to form target positioning.

The network judges the error through the loss function and updates the network weight parameters by the reverse gradient propagation of the network, thereby continuously reducing the value of the loss function to improve the detection accuracy. The detection network calculates multiple times through a large amount of training data and can learn a set of optimal weight values from the set of data to predict the detection target.

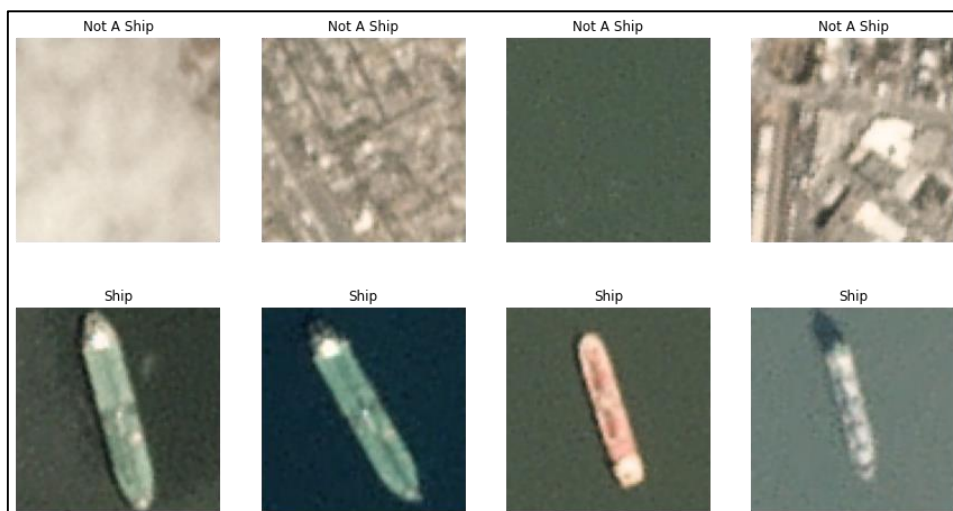
The model model is based on four Conv2D layers: an input layer with a 2x2 MaxPool2D pooling layer, two dense layers of 64 and 128 neurons each with a ReLU activation function, and a final layer with two neurons and the softmax activation function for producing final class predictions.

#### 5.4. IMPLEMENTATION

For this Section, code can be referred from the appendix attached at the end of the thesis.

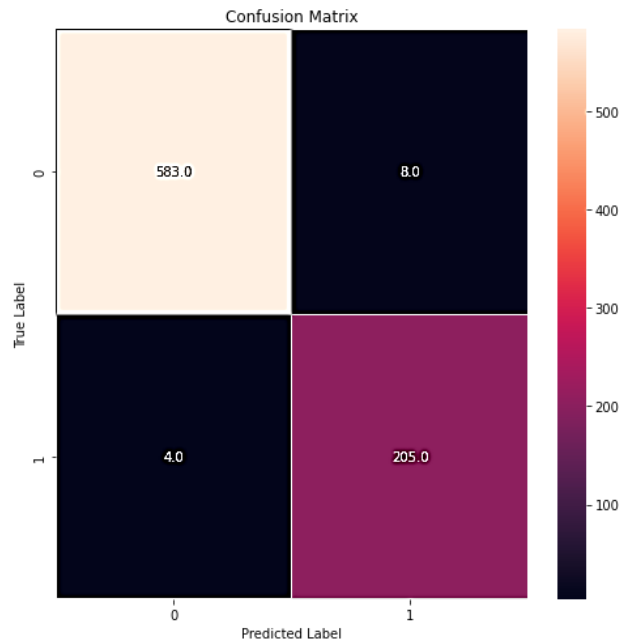
#### 5.5. RESULTS AND DISCUSSIONS

The following results were obtained as a result of the CNN model used for Dataset 1 and Dataset 2.



(a)





(b)

Model: "sequential\_8"

Layer (type)	Output Shape	Param #
conv2d_24 (Conv2D)	(None, 80, 80, 64)	3136
max_pooling2d_24 (MaxPooling)	(None, 16, 16, 64)	0
dropout_56 (Dropout)	(None, 16, 16, 64)	0
conv2d_25 (Conv2D)	(None, 16, 16, 32)	18464
max_pooling2d_25 (MaxPooling)	(None, 14, 14, 32)	0
dropout_57 (Dropout)	(None, 14, 14, 32)	0
conv2d_26 (Conv2D)	(None, 14, 14, 16)	2064
max_pooling2d_26 (MaxPooling)	(None, 12, 12, 16)	0
dropout_58 (Dropout)	(None, 12, 12, 16)	0
flatten_8 (Flatten)	(None, 2304)	0
dense_40 (Dense)	(None, 200)	461000
dropout_59 (Dropout)	(None, 200)	0
dense_41 (Dense)	(None, 100)	20100
dropout_60 (Dropout)	(None, 100)	0
dense_42 (Dense)	(None, 100)	10100
dropout_61 (Dropout)	(None, 100)	0
dense_43 (Dense)	(None, 50)	5050
dropout_62 (Dropout)	(None, 50)	0
dense_44 (Dense)	(None, 2)	102

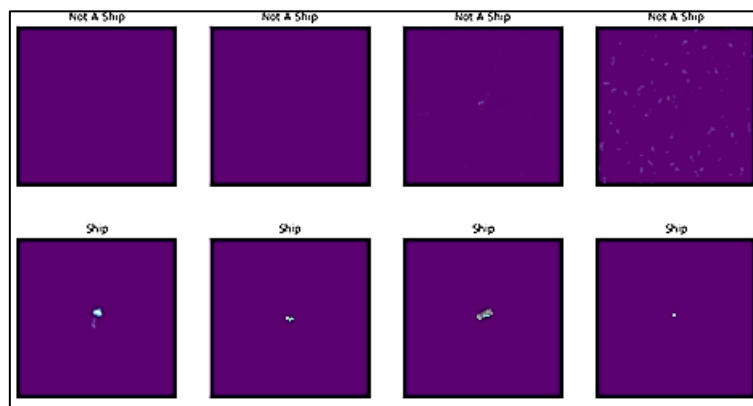
-----  
 Total params: 520,016  
 Trainable params: 520,016  
 Non-trainable params: 0

(c)

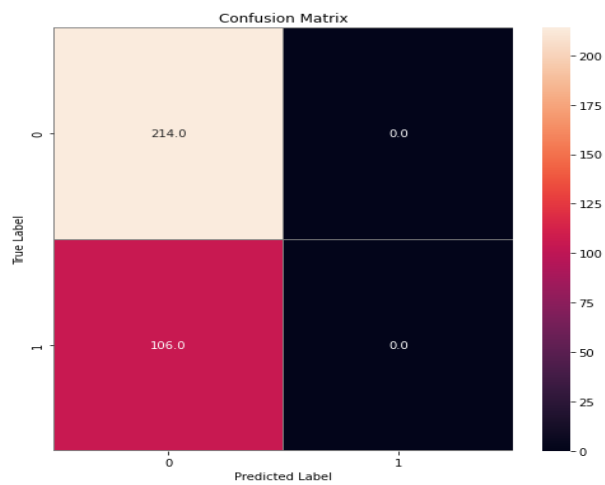
	precision	recall	f1-score	support
1. No Ship	0.99	0.99	0.99	587
2. Ship	0.98	0.96	0.97	213
accuracy			0.98	800
macro avg	0.98	0.98	0.98	800
weighted avg	0.98	0.98	0.98	800

(d)

Fig 22. The CNN results of optical satellite dataset in detection of ship class. (a) visualization of the ship and non-ship class labels. (b) Confusion Matrix (c) Model summary (d) Detection report.



(a)



(b)

Fig 23. CNN Detection results for SAR satellite dataset. (a) Visualization of the ship and non-ship class. (b) Confusion Matrix.

## **5.6. SUMMARY**

Synthetic Aperture Radar is a surveillance system that is particularly well suited to marine surveillance. Large swath widths, time-independent observations, and weather resistance can be particularly beneficial for detecting ships that are generally unseen to conventional means of monitoring. The accuracy of the CNN model used in this experiment was obtained as 98% accurate for the optical satellite dataset.

#### 6.1 CONCLUSION

An important component of Earth Observation in supporting the primary aims of the space and security and defence domain is the provision of image and geospatial intelligence products and services resulting from the exploitation of remotely sensed data acquired by sensors mounted on spaceborne assets. The reflections or emissions measured by the different types of sensors are depicted in images that need to be converted into meaningful information. For target detection in remotely sensed images, targets can be referred to as man-made or natural object or an event or activity of interest. Many applications of military and civilian applications involve the detection of an object or activity such as military vehicles or vehicles track. Hyperspectral target identification algorithms look at the spectrum of each pixel to find targets based on the spectral characteristics of the target's surface material. Combination of Hyperspectral data and LiDAR data can be found extremely useful in detection and also identification of the target such as camouflaged target used in this study.

Artificial Intelligence (AI) methods are progressively demonstrating the potential to get information out faster with more thorough and complete analysis. In recent years, neural network applications increasingly demonstrated better capability to automatically discover relevant contextual features in remotely sensed images. Data volume and computational capacity increased exponentially, boosting precisely the application of neural network computing to satellite image. Combining satellite radar imagery with Automatic Identification System (AIS), coastal radars, Vessel Monitoring System (VMS), and any available intelligence data provide useful information to build a database behavior concerning the vessel tracks in specific area. Any deviation from recognized track patterns might be considered as an anomaly to be further investigated. A thorough exploitation of SAR imagery strengths would enlarge the use of SAR imagery alone and/or in combined use with electro-optical images, thus taking full advantage of its unique 24/7 and all-weather characteristics, therefore raising the effectiveness of investments made by several European Ministries of defence on SAR satellites.

In this thesis, various target detection methodologies is used to detect targets in optical, microwave and Lidar data successfully jointly with the rich reference data available from multi-

platform and multi-sensors.

## **6.2 FUTURE SCOPE**

There is a great need for better performing target detection algorithms in real time data processing. Hence the development of better performing algorithms can revolutionize the remote sensing target detection application more importantly for defence and strategic interest targets such as camouflaged, hiding and disguising targets.

In case of applying deep learning artificial intelligence it should be known that with the diversification of detection task requirements, the target detection model is no longer a single task model, which adds instance segmentation (similar to multi-target detection, but uses edge contours instead of bounding boxes (target boxes)) and some are also added Panoramic segmentation (it is a combination of semantic segmentation and instance segmentation: semantic segmentation refers to assigning a category to each pixel on the image (can be distinguished by color) but does not distinguish between individuals). Hence, in order to pursue faster and more accurate target detection algorithm models, the algorithm model will incorporate more other advanced model algorithms, and single-stage and two-stage methods will gradually merge. For example, the target position estimation proposed by the single-stage CornerNet-Lite model is pseudo. The two-stage model adopts the idea of two-stage target detection.

## ***REFERENCES***

---

- [1] Mohamed El Zalaky. Target Detection Using Matched Filtering Analysis of the ASTER Data for Exploration of Uranium Mineralization in the Northern Part of El-Erediya Pluton , Central Eastern Desert , Egypt
- [2] Ariolfo Camacho, A comparative study of target detection algorithms in hyperspectral imagery applied to agricultural crops in Colombia
- [3] M Lega 1, C Ferrara, G Persechino, P Bishop Remote sensing in environmental police investigations: aerial platforms and an innovative application of thermography to detect several illegal activities
- [4] Chen Ke, Military object detection using multiple information extracted from hyperspectral imagery
- [5] Dimitris Manolakis and Gary Shaw, “Detection algorithms for hyperspectral Imaging applications”
- [6] Dimitris Manolakis, “Hyperspectral Image Processing for Automatic Target Detection Applications
- [7] D. Manolakis and G. A. Shaw, Detector algorithm for hyper spectral Imaging application.
- [8] C.I. Chang, Hyperspectral Imaging -Techniques for Spectral Detection and Classification,
- [9] A.R. Stefan , P. K. Varshney, “Further Results in Use of Independent Components Analysis for Target Detection in Hyperspectral Images
- [10] A. Hyvarinen A and E. Oja E, Independent Component Analysis : Algorithms and Applications
- [11] S. A. Robila, P. K. Varshney, “Target Detection in Hyperspectral Images Based on Independent Component Analysis”, Proceedings of SPIE AeroSense, 2002, 173-182
- [12] S-S. Chiang, C-I. Chang , and I.W. Ginsberg, “Unsupervised target detection in hyperspectral images using projection pursuit”, IEEE Transactions on Geoscience and Remote Sensing, 39, no. 7, 2001, 1380-1391
- [13] A. Ifarraguerri , and C.-I Chang, “Unsupervised hyperspectral image analysis with

- projection pursuit”, IEEE Transactions on Geoscience and Remote Sensing, 38, no. 6, 2000, 2529-2538
- [14] C.H. Chen, X. Zhang, “Independent component analysis for remote sensing”, Proceedings of SPIE Image and Signal Processing Conference for Remote Sensing V, 3871, 1999, 150-157
- [15] T.M. Tu, P.S. Huang, P.Y. Chen, “Blind separation of spectral signatures in hyperspectral imagery”, IEE Proceedings on Vision, Image and Signal Processing, 148, no. 4, 2001, 217-226
- [16] S-S. Chiang, C-I. Chang, and I.W. Ginsberg, “Unsupervised hyperspectral image analysis using independent component analysis”, Proceedings of IEEE Geoscience and Remote Sensing Symposium, 7, 2000, 3136-3138
- [17] A. J. Bell, and T. J. Sejnowski, “The 'Independent Components' of natural scenes are edge filters”, Vision Research, 37, no. 23, 1999, 3327-3338
- [18] S. A. Robila, P. K. Varshney, “A Fast Source Separation Algorithm for Hyperspectral Imagery”, Proceedings of IEEE IGARSS 2002, 3516-3518
- [19] Hyperspectral Digital Imagery Collection Experiment Documentation, August, 1995.
- [20] J. Bowles, J. Antoniadis, and all, “Real time analysis of hyperspectral data sets using NRL’s ORASIS algorithm”, Proceedings SPIE, 1997, 38-45.
- [21] Bell, A. and Sejnowski, T. (1997). The 'independent components' of natural scenes are edge filters. Vision Research, 37:3327–3338.
- [22] Cardoso, J.-F. (1997). Infomax and maximum likelihood for source separation. IEEE Letters on Signal Processing, 4:112–114.
- [23] Comon, P. (1994). Independent component analysis—a new concept? Signal Processing, 36:287–314
- [24] Delfosse, N. and Loubaton, P. (1995). Adaptive blind separation of independent sources: a deflation approach. Signal Processing, 45:59–83
- [25] Gonzales, R. and Wintz, P. (1987). Digital Image Processing. Addison-Wesley.
- [26] Hyvärinen, A. (1998b). New approximations of differential entropy for independent component analysis and projection pursuit. In Advances in Neural Information Processing Systems, volume 10, pages 273–279. MIT Press.
- [27] Kiviluoto, K. and Oja, E. (1998). Independent component analysis for parallel financial time series. In Proc. Int. Conf. on Neural Information Processing (ICONIP’98), volume 2, pages 895–898, Tokyo, Japan.
- [28] Vigário, R., Jousmäki, V., Hämäläinen, M., Hari, R., and Oja, E. (1998). Independent

- component analysis for identification of artifacts in magnetoencephalographic recordings. In *Advances in Neural Information Processing Systems*, volume 10, pages 229–235. MIT Press.
- [29] Jin, X. & Davis, C.H., 2005. Automated building extraction from high-resolution satellite imagery in urban areas using structural, contextual, and spectral information, *EURASIP Journal on Applied Signal Processing*, 2005(14): 2196-2206.
- [30] Meng, X., 2005. A slope- and elevation-based filter to remove non-ground measurements from airborne LIDAR Data, *UCGIS Summer Assembly*, June 28-July 1, Jackson Hole, Wyoming-USA.
- [31] Morgan, M., & Tempfli, K., 2000. Automatic building extraction from airborne laser scanning data, In *Proc. 19th ISPRS Congr., Amsterdam-The Netherlands*, book 3B, pp. 616-623.
- [32] Rottensteiner, F., Trinder, J., Clode, S., & Kubik, K., 2003. Building detection using LIDAR data and multi-spectral images, *Proc. VIIth Digital Image Computing: Techniques and Applications*, Sydney-Australia, pp.10-12.
- [33] Sohn, G. & Dowman, I., 2003. Building Extraction. Using LIDAR DEMs and IKONOS Images, *ISPRS, Volume XXXIV, PART 3/W13. Dresden-Germany*, pp. 8-10.
- [34] Peng, J., Zhang, D., & Liu, Y., 2005. An improved snake model for building detection from urban aerial images, *Pattern Recognition Letters*, 26(5): 587-595.
- [35] Rottensteiner, F., Trinder, J., Clode, S., & Kubik, K., 2003. Building detection using LIDAR data and multi-spectral images, *Proc. VIIth Digital Image Computing: Techniques and Applications*, Sydney-Australia, pp.10-12.



## ***APPENDIX 1: DATA COLLECTION***

---

### **Chapter Wise Data Collection sources or weblinks:**

#### **CHAPTER 2:**

1. <http://dirsapps.cis.rit.edu/share-2010/cgi-bin/share-2010.pl>
  - SpecTIR Hyperspectral Airborne Rochester Experiment (SHARE)
2. [http://dirsapps.cis.rit.edu/wwwdata/share-2010/SpecTIR\\_20100729/](http://dirsapps.cis.rit.edu/wwwdata/share-2010/SpecTIR_20100729/)
  - SpecTIR Imagery (non-Geo Corrected) (29 July 2010)

#### **CHAPTER 3:**

1. [http://dirsapps.cis.rit.edu/wwwdata/share-2010/LiDAR\\_20100726/](http://dirsapps.cis.rit.edu/wwwdata/share-2010/LiDAR_20100726/)
  - Kucera LiDAR Imagery (26 July 2010)
2. [http://dirsapps.cis.rit.edu/wwwdata/share-2010/GroundTruth/ContextPhotos\\_Ground/](http://dirsapps.cis.rit.edu/wwwdata/share-2010/GroundTruth/ContextPhotos_Ground/)
  - Ground truth contextual imagery

#### **CHAPTER 5:**

1. <https://www.kaggle.com/rhammell/ships-in-satellite-imagery>
  - Dataset for optical satellite image for ship detection
2. <https://ieee-dataport.org/documents/sar-ship-dataset-detection-discrimination-and-analysis>
  - A SAR SHIP DATASET FOR DETECTION, DISCRIMINATION AND ANALYSIS
  - IEEE Dataport <https://ieee-dataport.org/datasets>
  - Categories: Geoscience and Remote Sensing, Keywords: SAR, Machine Learning

### Chapter Wise source code that can be referred for implementation:

#### CHAPTER 4.

#### The Source Code for the developed ROC analysis using MATLAB for various algorithms

15/10/21 6:23 PM C:\Users\Sushmita Gautam\D...\QUAC\_TD.m

```
% Read in the data
w = 161;
h = 132;
p = 360;
M = multibandread('QUAC.dat', [w h p], 'uint16', 0, 'bsq', 'ieee-le');
IData = hyperGetHymapWavelengthsNm();
% Read in target signatures
[sig1,lsig] = hyperGetEnviSignature( 'tan.txt');
figure; imagesc(M(:, :, 18)); axis image; colormap(gray);
% Get signature from data for comparison
%fsig1 = squeeze(M(295,1139,:));
%sig1 = fsig1;
% Resample data to commone wavelength set
desiredLambdas = IData;
sig1 = squeeze(hyperResample(sig1, lsig, desiredLambdas));
%figure; plot(sig1); grid on; title('Signature 1');
xlabel('Wavelength [nm]'); ylabel('Reflectance [%]');
hold on; plot(fsig1, '--');
legend('Recorded', 'From Image');
sig1=transpose(sig1);
% Display data
M = hyperConvert2d(M);
%figure; imagesc(M_pct); axis image; title('Scene');
algorithm = lower('matchedfilter');
tic
switch algorithm
case 'matchedfilter'
r = hyperConvert3d(hyperMatchedFilter(M, sig1), w, h, 1);
case 'ace'
r = hyperConvert3d(hyperAce(M, sig1), w, h, 1);
case 'sid'
r = hyperConvert3d(hyperSid(M, sig1), w, h, 1);
case 'cem'
r = hyperConvert3d(hyperCem(M, sig1), w, h, 1);
end
toc
% Display results and write to file
figure; imagesc(r); axis image; colorbar;
title(algorithm);
```

```

[a,b]=sort(r(:),'descend');
tmpicamf = a(1:20);
figure; plot(tmpicamf./tmpicamf(1)); grid on;
[x, y, val] = hyperMax2d(r);
tmpmf = (hyperNormalize(r)*2^10);
multibandwrite(tmpicamf,'tmpicamf.tif','bsq');
[pd,fa] = hyperRoc(r);
figure; plot(fa,pd,'); grid on; title(sprintf('%s\n%s',algorithm, 'tan.txt'));

algorithm = lower('sam');
tic
switch algorithm
case 'matchedfilter'
r = hyperConvert3d(hyperMatchedFilter(M, sig1), w, h, 1);
case 'ace'
r = hyperConvert3d(hyperAce(M, sig1), w, h, 1);
case 'sid'
r = hyperConvert3d(hyperSid(M, sig1), w, h, 1);
case 'cem'
r = hyperConvert3d(hyperCem(M, sig1), w, h, 1);
case 'sam'
r = (1./(eps+hyperConvert3d(hyperSam(M, sig1), w, h, 1)));
end
toc
% Display results and write to file
figure; imagesc(r); axis image; colorbar;
title(algorithm);
[a,b]=sort(r(:),'descend');
tmksam2 = a(1:20);
figure; plot(tmksam2./tmksam2(1)); grid on;
[x, y, val] = hyperMax2d(r);
tmksam2 = (hyperNormalize(r)*2^10);
multibandwrite(tmksam2,'tmksam2.tif','bsq');
[pd,fa] = hyperRoc(r);
figure; plot(fa,pd,'); grid on; title(sprintf('%s\n%s',algorithm, 'tan.txt'));
algorithm = lower('cem');
tic
switch algorithm
case 'matchedfilter'
r = hyperConvert3d(hyperMatchedFilter(M, sig1), w, h, 1);
case 'ace'
r = hyperConvert3d(hyperAce(M, sig1), w, h, 1);
case 'sid'
r = hyperConvert3d(hyperSid(M, sig1), w, h, 1);
case 'cem'
r = hyperConvert3d(hyperCem(M, sig1), w, h, 1);
case 'sam'
r = (1./(eps+hyperConvert3d(hyperSam(M, sig1), w, h, 1)));
end
toc
% Display results and write to file
figure; imagesc(r); axis image; colorbar;
title(algorithm);
[a,b]=sort(r(:),'descend');
tmpcem = a(1:20);
figure; plot(tmpcem./tmpcem(1)); grid on;
[x, y, val] = hyperMax2d(r);

```

```

tmpcem = (hyperNormalize(r)*2^10);
multibandwrite(tmpcem,'tmpcem.tif','bsq');
[pd,fa] = hyperRoc(r);
figure; plot(fa,pd, '.'); grid on; title(sprintf('%s\n%s',algorithm, 'tan.txt'));
algorithm = lower('sam');
tic
switch algorithm
case 'matchedfilter'
r = hyperConvert3d(hyperMatchedFilter(M, sig1), w, h, 1);
case 'ace'
r = hyperConvert3d(hyperAce(M, sig1), w, h, 1);
case 'sid'
r = hyperConvert3d(hyperSid(M, sig1), w, h, 1);
case 'cem'
r = hyperConvert3d(hyperCem(M, sig1), w, h, 1);
case 'sam'
r = (1./(eps+hyperConvert3d(hyperSam(M, sig1), w, h, 1)));
end
toc
% Display results and write to file
figure; imagesc(r); axis image; colorbar;
title(algorithm);
[a,b]=sort(r(:),'descend');
tmpsam2 = a(1:20);
figure; plot(tmpsam./tmpsam(1)); grid on;
[x, y, val] = hyperMax2d(r);
tmpsam2 = (hyperNormalize(r)*2^10);
multibandwrite(tmpsam2,'tmpsam2.tif','bsq');
[pd,fa] = hyperRoc(r);
figure; plot(fa,pd, '.'); grid on; title(sprintf('%s\n%s',algorithm, 'tan.txt'));
algorithm = lower('sid');
tic
switch algorithm
case 'matchedfilter'
r = hyperConvert3d(hyperMatchedFilter(M, sig1), w, h, 1);
case 'ace'
r = hyperConvert3d(hyperAce(M, sig1), w, h, 1);
case 'sid'
r = hyperConvert3d(hyperSid(M, sig1), w, h, 1);
case 'cem'
r = hyperConvert3d(hyperCem(M, sig1), w, h, 1);
case 'sam'
r = (1./(eps+hyperConvert3d(hyperSam(M, sig1), w, h, 1)));
end
toc
% Display results and write to file
figure; imagesc(r); axis image; colorbar;
title(algorithm);
[a,b]=sort(r(:),'descend');
tmpsid = a(1:20);
figure; plot(tmpsam./tmpsam(1)); grid on;
[x, y, val] = hyperMax2d(r);
tmpsid = (hyperNormalize(r)*2^10);
multibandwrite(tmpsid,'tmpsid.tif','bsq');
[pd,fa] = hyperRoc(r);
figure; plot(fa,pd, '.'); grid on; title(sprintf('%s\n%s',algorithm, 'tan.txt'))

```

```

algorithm = lower('ica-eea');
tic
switch algorithm
case 'ica-eea'
[U, X] = hyperIcaEea(M, 50, sig1);
r = X(1,:);
r = hyperConvert3d(r, w, h, 1);
end
toc
% Display results and write to file
figure; imagesc(r); axis image; colorbar;
title(algorithm);
[a,b]=sort(r:),'descend');
tmpica = a(1:20);
figure; plot(tmpica./tmpica(1)); grid on;
[x, y, val] = hyperMax2d(r);
tmpica = (hyperNormalize(r)*2^10);
multibandwrite(tmpica,'tmpica.tif','bsq');
[pd,fa] = hyperRoc(r);
figure; plot(fa,pd, '.'); grid on; title(sprintf('%s\n%s',algorithm, 'tan.txt'));
algorithm = lower('sam');
tic
switch algorithm
case 'matchedfilter'
r = hyperConvert3d(hyperMatchedFilter(M, sig1), w, h, 1);
case 'ace'
r = hyperConvert3d(hyperAce(M, sig1), w, h, 1);
case 'sid'
r = hyperConvert3d(hyperSid(M, sig1), w, h, 1);
case 'cem'
r = hyperConvert3d(hyperCem(M, sig1), w, h, 1);
case 'sam'
r = (1./(eps+hyperConvert3d(hyperSam(M, sig1), w, h, 1)));
end
toc
% Display results and write to file
figure; imagesc(r); axis image; colorbar;
title(algorithm);
[a,b]=sort(r:),'descend');
tmpsam2 = a(1:20);
figure; plot(tmpsam2./tmpsam2(1)); grid on;
[x, y, val] = hyperMax2d(r);
tmpsam2 = (hyperNormalize(r)*2^10);
multibandwrite(tmpsam2,'tmpsam2.tif','bsq');
[pd,fa] = hyperRoc(r);
figure; plot(fa,pd, '.'); grid on; title(sprintf('%s\n%s',algorithm, 'tan.txt'));

```

## CHAPTER 5.

### The Source Code for the developed CNN model implementation on optical satellite data.

```
# -*- coding: utf-8 -*-
```

```
"""Copy of optical_ship.ipynb
```

```
Automatically generated by Colaboratory.
```

```
Original file is located at
```

```
https://colab.research.google.com/drive/1XpF87kGonQZvejMfSQ6CnR\_Vw\_Vgwa1\_
```

```
"""
```

```
from google.colab import drive
```

```
drive.mount('/content/drive')
```

```
!ls "/content/drive/My Drive/SHIP/shipsnet.json"
```

```
import numpy as np
```

```
from numpy import expand_dims
```

```
import pandas as pd
```

```
import json
```

```
import matplotlib.pyplot as plt
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.model_selection import cross_val_score
```

```
from sklearn.metrics import confusion_matrix
```

```
import tensorflow as tf
```

```
from tensorflow.keras.utils import to_categorical
```

```
import keras
```

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

```
from keras.wrappers.scikit_learn import KerasClassifier
```

```
from tensorflow.keras.models import Sequential
```

```
from tensorflow.keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D
```

```
from tensorflow.keras.optimizers import RMSprop, Adam
```

```
from keras.preprocessing.image import ImageDataGenerator
```

```

from tensorflow.keras.callbacks import EarlyStopping

with open('/content/drive/My Drive/SHIP/shipsnet.json') as data_file:
    dataset = json.load(data_file)
shipsnet= pd.DataFrame(dataset)
shipsnet.head()

shipsnet = shipsnet[["data", "labels"]]
shipsnet.head()

ship_images = shipsnet["labels"].value_counts()[0]
no_ship_images = shipsnet["labels"].value_counts()[1]
print("Number of the ship_images :{}".format(ship_images),"n")
print("Number of the ship_images :{}".format(no_ship_images))

# Turning the json information into numpy array and then assign it as x and y variables
x = np.array(dataset['data']).astype('uint8')
y = np.array(dataset['labels']).astype('uint8')

x.shape

x_reshaped = x.reshape([-1, 3, 80, 80])

x_reshaped.shape

x_reshaped = x.reshape([-1, 3, 80, 80]).transpose([0,2,3,1])
x_reshaped.shape

y.shape

y_reshaped = to_categorical(y, num_classes=2)

y_reshaped.shape

y_reshaped

image_no_ship = x_reshaped[y==0]
image_ship = x_reshaped[y==1]

def plot(a,b):

```

```

plt.figure(figsize=(15, 15))
for i, k in enumerate(range(1,9)):
    if i < 4:
        plt.subplot(2,4,k)
        plt.title('Not A Ship')
        plt.imshow(image_no_ship[i+2])
        plt.axis("off")
    else:
        plt.subplot(2,4,k)
        plt.title('Ship')
        plt.imshow(image_ship[i+15])
        plt.axis("off")

plt.subplots_adjust(bottom=0.3, top=0.7, hspace=0.25)

#Implementation of the function

plot(image_no_ship, image_ship)

x_reshaped = x_reshaped / 255

x_reshaped[0][0][0] # Normalized RGB values of the first pixel of the first image in the dataset.

n_bins = 30
plt.hist(x_reshaped[y == 0][0][:,:,0].flatten(), bins = n_bins, lw = 0, color = 'r', alpha = 0.5);
plt.hist(x_reshaped[y == 0][0][:,:,1].flatten(), bins = n_bins, lw = 0, color = 'g', alpha = 0.5);
plt.hist(x_reshaped[y == 0][0][:,:,2].flatten(), bins = n_bins, lw = 0, color = 'b', alpha = 0.5);
plt.ylabel('Count', fontweight = "bold")
plt.xlabel('Pixel Intensity', fontweight = "bold")
plt.title("Histogram of normalized data")
plt.show()

x_train_1, x_test, y_train_1, y_test = train_test_split(x_reshaped, y_reshaped,
                                                    test_size = 0.20, random_state = 42)

x_train, x_val, y_train, y_val = train_test_split(x_train_1, y_train_1,
                                                    test_size = 0.25, random_state = 42)

```



```
print("x_train shape",x_train.shape)
print("x_test shape",x_test.shape)
print("y_train shape",y_train.shape)
print("y_test shape",y_test.shape)
print("y_train shape",x_val.shape)
print("y_test shape",y_val.shape)
```

```
x_train.shape
```

```
model = Sequential()
#
model.add(Conv2D(filters = 64, kernel_size = (4,4),padding = 'Same',
                activation = 'relu', input_shape = (80,80,3)))
model.add(MaxPool2D(pool_size=(5,5)))
model.add(Dropout(0.25))
#
model.add(Conv2D(filters = 32, kernel_size = (3,3),padding = 'Same',
                activation = 'relu'))
model.add(MaxPool2D(pool_size=(3,3), strides=(1,1)))
model.add(Dropout(0.25))
#
model.add(Conv2D(filters = 16, kernel_size = (2,2),padding = 'Same',
                activation = 'relu'))
model.add(MaxPool2D(pool_size=(3,3), strides=(1,1)))
model.add(Dropout(0.25))

# Fully connected
model.add(Flatten())
model.add(Dense(200, activation = "relu"))
model.add(Dropout(0.5))
model.add(Dense(100, activation = "relu"))
model.add(Dropout(0.5))
model.add(Dense(100, activation = "relu"))
model.add(Dropout(0.5))
model.add(Dense(50, activation = "relu"))
model.add(Dropout(0.5))
model.add(Dense(2, activation = "softmax"))

optimizer = Adam(lr=0.001, beta_1=0.9, beta_2=0.999)
```

```

model.compile(optimizer = optimizer , loss = "categorical_crossentropy", metrics=["accuracy"])
from tensorflow.keras import callbacks
earlystopping = callbacks.EarlyStopping(monitor ="val_loss", mode ="min", patience = 10,
restore_best_weights = True)
history = model.fit(x_train, y_train, epochs = 100, validation_data=(x_val, y_val), callbacks = [earlystopping])

model.evaluate(x_test, y_test)

pd.DataFrame(history.history).plot();

```

```

datagen = ImageDataGenerator(
    featurewise_center=False,
    samplewise_center=False,
    featurewise_std_normalization=False,
    samplewise_std_normalization=False,
    zca_whitening=False,
    rotation_range=5,
    zoom_range = 0.1,
    width_shift_range=0.1,
    height_shift_range=0.1,
    horizontal_flip=False,
    vertical_flip=False)

```

```

datagen.fit(x_train)

```

```

data = x_resaped[y==1][15]
# expand dimension to one sample
samples = expand_dims(data, 0)
# create image data augmentation generator
datag = ImageDataGenerator(brightness_range=[0.2,1.0],
    zoom_range=[0.5,1.0],
    horizontal_flip=True,
    rotation_range=90)
# prepare iterator
it = datag.flow(samples, batch_size=1)
# generate samples and plot
plt.figure(figsize = (10,10))
for i in range(9):
    # define subplot

```

```

plt.subplot(3,3,i+1)
# generate batch of images
batch = it.next()
# convert to unsigned integers for viewing
image = batch[0].astype('uint8')
# plot raw pixel data
plt.imshow(image)
# show the figure
plt.show()

history = model.fit(datagen.flow(x_train, y_train), epochs = 100,
                    validation_data=(x_val, y_val), callbacks = [earlystopping])

model.evaluate(x_test, y_test)

from sklearn import metrics
import seaborn as sns
Y_pred = model.predict(x_test)
# Convert predictions classes to one hot vectors
Y_pred_classes = np.argmax(Y_pred,axis = 1)
# Convert validation observations to one hot vectors
Y_true = np.argmax(y_test,axis = 1)
# Compute the confusion matrix

print("\n""Test Accuracy Score : ",metrics.accuracy_score(Y_true, Y_pred_classes),"\n")

fig, axis = plt.subplots(1, 3, figsize=(20,6))
axis[0].plot(history.history['val_accuracy'], label='val_acc')
axis[0].set_title("Validation Accuracy")
axis[0].set_xlabel("Epochs")
axis[0].set_ylabel("Val. Acc.")
axis[1].plot(history.history['accuracy'], label='acc')
axis[1].set_title("Training Accuracy")
axis[1].set_xlabel("Epochs")
axis[0].set_ylabel("Train. Acc.")
axis[2].plot(history.history['val_loss'], label='val_loss')
axis[2].set_title("Test Loss")
axis[2].set_xlabel("Epochs")
axis[2].set_ylabel("Loss")
plt.show()

```

```

confusion_mtx = confusion_matrix(Y_true, Y_pred_classes)
# Plot the confusion matrix
f,ax = plt.subplots(figsize=(7, 7))
sns.heatmap(confusion_mtx, annot=True, linewidths=0.01,linecolor="gray", fmt= '.1f',ax=ax)
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix")
plt.tight_layout()

plt.show()

pd.DataFrame(history.history).plot();

prediction = model.predict(x_test)
pd.Series(prediction[0], index=["Not A Ship", "Ship"])

with open('modelsummary.txt', 'w') as f:

    model.summary(print_fn=lambda x: f.write(x + '\n'))

plt.rc('figure', figsize=(12, 7))
#plt.text(0.01, 0.05, str(model.summary()), {'fontsize': 12}) old approach
plt.text(0.01, 0.05, str(model.summary()), {'fontsize': 10}, fontproperties = 'monospace') # approach improved
by OP -> monospace!
plt.axis('off')
plt.tight_layout()
plt.savefig('output_model_Summary.png')

from sklearn.metrics import classification_report, accuracy_score

pred = np.argmax(model.predict(x_test), axis=1)

# Classification Report
print(classification_report(pred, np.argmax(y_test, 1),
    target_names = ['1. No Ship', '2. Ship']))

```

## The Source Code for the developed CNN model implementation on SAR satellite data.

```
# -*- coding: utf-8 -*-
```

```
""""SAR SHIP.ipynb
```

Automatically generated by Colaboratory.

Original file is located at

<https://colab.research.google.com/drive/1DUO5rRgpy1cyh0SLa7SmZur7J2-9UEjR>

```
""""
```

```
from google.colab import drive
```

```
drive.mount('/content/drive')
```

```
!ls "/content/drive/My Drive/ieee sar ship subscribed"
```

```
import numpy as np
```

```
from numpy import expand_dims
```

```
import pandas as pd
```

```
import json
```

```
import matplotlib.pyplot as plt
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.model_selection import cross_val_score
```

```
from sklearn.metrics import confusion_matrix
```

```
from tensorflow.keras.utils import to_categorical
```

```
import keras
```

```
from keras import layers
```

```
from keras.wrappers.scikit_learn import KerasClassifier
```

```
from keras.models import Sequential
```

```
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D
```

```
from tensorflow.keras.optimizers import RMSprop,Adam
```

```
from keras.preprocessing.image import ImageDataGenerator
```

```
from keras.callbacks import EarlyStopping
```

```
import json
```

```

#dataset = json.load('/content/drive/My Drive/ieee sar ship subscribed/json/ship_positives')
with open('/content/drive/My Drive/ieee sar ship subscribed/json/ship_positives.json') as data_file:
    dataset = json.load(data_file)
ship= pd.DataFrame(dataset).T
ship.head()

ship.rename(columns={0:'column1'})

import pandas as pd

df2 = pd.json_normalize(ship[0])

df2.head()

dtype = type(df2["patchgt"])
print(dtype)

ship_images = df2["validais"].value_counts()[0]
no_ship_images = df2["validais"].value_counts()[1]

print("Number of the ship_images :{}".format(ship_images),"\n")
print("Number of the no ship_images :{}".format(no_ship_images))

dtype_before = type(df2["patchfu"])

data = df2["patchfu"].tolist()

dtype_after = type(data)

print("Data type before converting = {}\nData type after converting = {}".format(dtype_before, dtype_after))

print (data)

print(data[0:10])

labels = df2["validais"].tolist()

print(labels[0:10])

x = np.array(data).astype('uint8')

```

```

y = np.array(labels).astype('uint8')

newarr = x.reshape(x.shape[0], (x.shape[1]*x.shape[2]))

x= newarr

x.shape

x_reshaped = x.reshape([-1, 101, 101])

x_reshaped.shape

x_reshaped = x.reshape([-1,101, 101]).transpose([0,2,1])
x_reshaped.shape

y_reshaped = to_categorical(y, num_classes=2)

y_reshaped.shape

image_no_ship = x_reshaped[y==0]
image_ship = x_reshaped[y==1]

def plot(a,b):

    plt.figure(figsize=(15, 15))
    for i, k in enumerate(range(1,9)):
        if i < 4:
            plt.subplot(2,4,k)
            plt.title('Not A Ship')
            plt.imshow(image_no_ship[i+2])
            plt.axis("off")
        else:
            plt.subplot(2,4,k)
            plt.title('Ship')
            plt.imshow(image_ship[i+15])
            plt.axis("off")

    plt.subplots_adjust(bottom=0.3, top=0.7, hspace=0.25)

```

```

#Implementation of the function

plot(image_no_ship, image_ship)

x_reshaped = x.reshape([-1,1,101, 101]).transpose([0,2,3,1])
x_reshaped.shape

x_reshaped = x_reshaped / 255

x_reshaped[0][0][0] # Normalized RGB values of the first pixel of the first image in the dataset.

x_train_1, x_test, y_train_1, y_test = train_test_split(x_reshaped, y_reshaped,
                                                    test_size = 0.20, random_state = 42)

x_train, x_val, y_train, y_val = train_test_split(x_train_1, y_train_1,
                                                    test_size = 0.25, random_state = 42)

print("x_train shape",x_train.shape)
print("x_test shape",x_test.shape)
print("y_train shape",y_train.shape)
print("y_test shape",y_test.shape)
print("y_train shape",x_val.shape)
print("y_test shape",y_val.shape)

x_train.shape

from keras import callbacks
model = Sequential()
#
model.add(Conv2D(filters = 32, kernel_size = (4,4),padding = 'Same',
                activation = 'relu', input_shape = (101,101,1)))
model.add(MaxPool2D(pool_size=(1,1)))
model.add(Dropout(0.25))
#
model.add(Conv2D(filters = 32, kernel_size = (3,3),padding = 'Same',
                activation = 'relu'))
model.add(MaxPool2D(pool_size=(1,1), strides=(1,1)))
model.add(Dropout(0.25))

```



```

#
model.add(Conv2D(filters = 16, kernel_size = (2,2),padding = 'Same',
                activation = 'relu'))
model.add(MaxPool2D(pool_size=(1,1), strides=(1,1)))
model.add(Dropout(0.25))

# Fully connected
model.add(Flatten())
model.add(Dense(200, activation = "relu"))
model.add(Dropout(0.5))
model.add(Dense(100, activation = "relu"))
model.add(Dropout(0.5))
model.add(Dense(100, activation = "relu"))
model.add(Dropout(0.5))
model.add(Dense(50, activation = "relu"))
model.add(Dropout(0.5))
model.add(Dense(2, activation = "softmax"))

optimizer = Adam(lr=0.001, beta_1=0.9, beta_2=0.999)

model.compile(optimizer = optimizer , loss = "categorical_crossentropy", metrics=["accuracy"])

earlystopping = callbacks.EarlyStopping(monitor = "val_loss",
                                       mode = "min", patience = 10,
                                       restore_best_weights = True)
history = model.fit(x_train, y_train, epochs = 100, validation_data=(x_val, y_val), callbacks = [earlystopping])

model.evaluate(x_test, y_test)

pd.DataFrame(history.history).plot();

datagen = ImageDataGenerator(
    featurewise_center=False,
    samplewise_center=False,
    featurewise_std_normalization=False,
    samplewise_std_normalization=False,
    zca_whitening=False,
    rotation_range=5,
    zoom_range = 0.1,
    width_shift_range=0.1,

```

```

        height_shift_range=0.1,
        horizontal_flip=False,
        vertical_flip=False)

datagen.fit(x_train)

history = model.fit(datagen.flow(x_train, y_train), epochs = 100,
                    validation_data=(x_val, y_val), callbacks = [earlystopping])

model.evaluate(x_test, y_test)

from sklearn import metrics
import seaborn as sns
Y_pred = model.predict(x_test)
# Convert predictions classes to one hot vectors
Y_pred_classes = np.argmax(Y_pred,axis = 1)
# Convert validation observations to one hot vectors
Y_true = np.argmax(y_test,axis = 1)
# Compute the confusion matrix

print("\n""Test Accuracy Score : ",metrics.accuracy_score(Y_true, Y_pred_classes),"\n")

fig, axis = plt.subplots(1, 3, figsize=(20,6))
axis[0].plot(history.history['val_accuracy'], label='val_acc')
axis[0].set_title("Validation Accuracy")
axis[0].set_xlabel("Epochs")
axis[0].set_ylabel("Val. Acc.")
axis[1].plot(history.history['accuracy'], label='acc')
axis[1].set_title("Training Accuracy")
axis[1].set_xlabel("Epochs")
axis[0].set_ylabel("Train. Acc.")
axis[2].plot(history.history['val_loss'], label='val_loss')
axis[2].set_title("Test Loss")
axis[2].set_xlabel("Epochs")
axis[2].set_ylabel("Loss")
plt.show()

confusion_mtx = confusion_matrix(Y_true, Y_pred_classes)
# Plot the confusion matrix
f,ax = plt.subplots(figsize=(7, 7))

```

```

sns.heatmap(confusion_mtx, annot=True, linewidths=0.01, linecolor="gray", fmt= '.1f', ax=ax)
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix")
plt.tight_layout()

plt.show()

pd.DataFrame(history.history).plot();

prediction = model.predict(x_test)
pd.Series(prediction[0], index=["Not A Ship", "Ship"])

with open('modelssummary.txt', 'w') as f:

    model.summary(print_fn=lambda x: f.write(x + '\n'))

plt.rc('figure', figsize=(12, 7))
#plt.text(0.01, 0.05, str(model.summary()), {'fontsize': 12}) old approach
plt.text(0.01, 0.05, str(model.summary()), {'fontsize': 10}, fontproperties = 'monospace') # approach improved
by OP -> monospace!
plt.axis('off')
plt.tight_layout()
plt.savefig('output_model_Summary.png')

```