Classification Techniques of PolSAR Images

by

Gadhiya Tusharkumar Damjibhai
201621009

A Thesis Submitted in Partial Fulfilment of the Requirements for the Degree of

DOCTOR OF PHILOSOPHY
to

DHIRUBHAI AMBANI INSTITUTE OF INFORMATION AND COMMUNICATION TECHNOLOGY

February, 2021

Declaration

I hereby declare that

i) the thesis comprises of my original work towards the degree of Doctor of Philosophy in Information and Communication Technology at Dhirubhai Ambani Institute of Information and Communication Technology and has not been submitted elsewhere for a degree,

ii) due acknowledgment has been made in the text to all the reference material used.

Gadhiya Tusharkumar Damjibhai

Certificate

This is to certify that the thesis work entitled CLASSIFICATION TECHNIQUES OF POLSAR IMAGES has been carried out by GADHIYA TUSHARKUMAR DAMJIBHAI for the degree of Doctor of Philosophy at *Dhirubhai Ambani Institute of Information and Communication Technology* under my supervision.

Dr. Anil K. Roy
Thesis Supervisor

Acknowledgement

Firstly, I would like to express my sincere gratitude to my advisor Dr. Anil K. Roy, for his continuous support and motivation during my Ph.D. study and research. His guidance helped me in my research and writing of this thesis.

I would like to thank my RPS and Synopsis committee members: Prof. Suman Mitra, Prof. M. V. Joshi, and Prof. Aditya Tatu, for their critical comments and insightful inputs which helped improve my thesis. I was fortunate that their broad experience in this field demanded me to widen my research problem from various perspectives

My sincere thanks go to the co-authors of some of my research papers Dr. Vinod Mall, Rutvik Shah, Akshit Soni, Sumanth Tangirala, Maitreya Patel, and Anery Patel.

I thank my fellow labmates Abhishek Jani and Nishith Kotak for the stimulating discussions and for all the fun we have had in the last four years. I am also thankful to my fellow Ph.D. scholars of DA-IICT whom i interacted with during my tenure here.

Special thanks are due to my wife for her continuous moral support, without which my Ph.D. journey would have been difficult. It would be difficult to express my gratitude for each and everyone who contributed in completing my thesis in these four years, but at the same time it would not conclude without expressing my deepest regard for my parents and family members for encouraging me throughout my Ph.D. study and my life in general.

Contents

Abstract

Over the years, optical remote sensing technology has restricted the ability to capture images during harsh weather settings and at night time. However, Synthetic Aperture Radar (SAR) is independent of solar illumination and thus allows all-weather continuous earth monitoring capability. A polarimetric synthetic aperture radar (PolSAR) is one type of SAR image which captures different attributes of the target by combining four different polarization states. Some PolSAR systems such as E-SAR, AIRSAR, F-SAR, etc., can capture abundant information of the target by employing multifrequency bands simultaneously providing rich information of the target. It makes SAR images suitable in wide range of Earth observation applications such as change detection, object detection, monitoring, classification, etc. This thesis addresses the classification problem of single frequency and multifrequency PolSAR images. PolSAR image classification is primarily a pixel based classification problem where our goal is to assign a label to each pixel of the image. Unlike optical images, PolSAR images are complex in nature which limits our ability of direct visual interpretation. Due to its active imaging nature, SAR images suffers from speckle noise which hinders the performance of pixel based classification.

To address the classification problem, five contributions related to improving classification time and accuracy are discussed. We will begin with the introduction of Optimized Wishart Network (OWN) which is an improvement over the existing Wishart Network (WN) for the classification of single frequency PolSAR images. We propose methods to improve the classification time by reducing the computation overhead in WN and improve classification accuracy by proposing a better weight initialization method. Next, we propose the extended OWN (e-OWN) for classification of multifrequency PolSAR data. We show that the proposed method is able to combine different band information effectively and produces better classification accuracy. One of the big challenge for pixel based PolSAR image classification method is the presence of speckle noise in the image. To tackle that, we propose the superpixel driven OWN which uses both pixel and superpixel information to handle the noisy pixels. Finally, we present an stacked

autoencoder based classification of multifrequency PolSAR images. All the proposed approaches are tested on variety of single frequency and multifrequency PolSAR datasets.

List of Tables

List of Figures

CHAPTER 1
Introduction

Humankind has been fascinated by Earth Observation (EO) since the first photograph of the Earth from the space [37]. The Blue Marble is still one of the most reproduced images in the history [103]. High-resolution wide-area images are required in the diverse fields of EO, ranging from urban planning and disaster management to agriculture and geology. It would be difficult to capture images of large areas from the ground. To do so, photographs are required to be taken from an altitude. It led to development of the remote sensing field. Remote sensing is the field of obtaining information about an object from a distance. It is achieved by mounting imaging sensors on a remote controlled balloon, an aircraft or a satellite. Remote sensing has given us the ability to capture the entire Earth's surface and gather tremendous amount of information in a short span of time. Since manual processing of such huge volume of data is not possible, we require computer systems that can extract meaningful information to help us better understand Earth's surface.

Typically, there are two types of imaging sensors, namely, passive sensors and active sensors. Passive sensors depend on the external illumination source, such as sunlight, to capture the image. Optical sensors like normal camera and hyperspectral sensors are examples of passive sensors. Under certain conditions such as cloud, fog, rain, or at night-time, passive sensors can not capture images due to either lack of illumination source or obstacle in the path from the sensor to the object. Thus we require active sensors which have their own source of illumination. Synthetic Aperture Radar (SAR) uses microwave signals with a long-range propagation characteristic that facilitates such imaging. Depending upon the combination of transmitters and receivers of such satellites, different types of SAR images are generated. One such type of SAR image is called full polarimetric SAR image (PolSAR). In case of a PolSAR image we receive data from four different channels, data from each channel represent different features of the same object. Some SAR systems are capable of capturing SAR images using multiple frequency bands,

Table 1.1: Notable application of SAR images

Fields	**Applications**
Agriculture	crop classification, soil moisture estimation [51], growth monitoring [85], penology estimation [98]
Urban	urban density estimation [65], change detection [92], urbanization [117], subsidence monitoring [35]
Disaster Management	disaster monitoring [125], landslide detection [112], volcano monitoring [122], earthquake damage estimation [71]
Oceanography	ocean wave monitoring [110], ship detection [84], oil spill detection [99]
Cryosphere	iceberg detection [26], snow cover mapping [101], glacier monitoring [114]
Forestry	forest height estimation [74], forest biomass estimation [76], forest fire monitoring [46], forest classification [97]

simultaneously resulting in a multitude of information compared to the optical images received from ordinary satellites. Therefore SAR images are gaining popularity for solving problems in various research domains, some of which are listed in Table 1.1. The massive size and complex nature of the SAR image limits our ability of manual interpretation. Thus, we require automated systems which can analyze these images to provide useful information. In the following sections we briefly discuss the process of PolSAR image acquisition and its representation. After that the PolSAR image classification problem is elaborated.

Table 1.2: Different frequency bands used in SAR

Band	**Frequency (GHz)**	**Wavelength (cm)**
P	0.3-1	30-100
L	1-2	15-30
S	2-4	7.5-15
C	4-8	3.8-7.5
X	8-12.5	2.4-3.8
K_u	12.5-18	1.7-2.4
K	18-26.5	1.1-1.7
K_a	26.5-40	0.75-1.1

1.1 SAR Image Acquisition

A SAR can be mounted on air-borne, say airplanes, space-borne platforms such as satellites or ground-based system. It operates at frequency band ranging from P to K_a in the microwave spectrum, as shown in Table 1.2. High wavelength bands such as P and L bands have high penetration capability which allows them to obtain even below ground information. It makes them suitable for forestry and agriculture applications. On the other hand, S and C bands have medium penetration capability which allow them to penetrate upper canopy layers of the vegetation area. They are also sensitive to ocean and ice surfaces which make them suitable for oceanography and cryosphere applications. Low wavelength bands such as X to K_u bands have low penetration capability but they provide high resolution images due to their high frequency. It makes them suitable for high resolution applications such as land target recognition and change detection. Table 1.3 shows SAR systems developed by different organizations whose data are freely available.

Table 1.3: List of SAR systems whose dataset is openly available

System	Band	Polarization	Resolution	Year	Organization
SEASAT [2]	L	HH	25m	1978	NASA
AIRSAR [1]	C,P,L	Full	1,8m-7.5m	1990-2004	NASA
ERS-1/2 [3]	C	VV	26m	1991-2011	ESA
JERS-1 [4]	L	HH	18m	1992-1998	JAXA
ALOS PALSAR [5]	L	Full	5m-10m	2002	JAXA
RADARSAT-1 [6]	C	HH	8m	1995	CSA
Sentinel-1a/1b [7]	C	Dual	5m-40m	2014-present	ESA
SMAP [8]	L	VV, HV, HV		2015-present	NASA
UAVSAR [9]	L	Full	1.0m-1.8m	2008-present	NASA

SAR transmits horizontal (H) or vertical (V) polarized electromagnetic waves towards the Earth's surface. Upon hitting the Earth's surface, these waves get scattered in all directions. Some of them are intercepted by SAR receivers. The SAR receiver measures the amplitude and phase of the backscattered wave. As shown in Figure 1.1, there are four possible combinations of the backscattered waves, namely, HH, HV, VH, VV. Here,

- HH - Transmitted H polarized wave and received H polarized wave,
- HV - Transmitted H polarized wave and received V polarized wave,

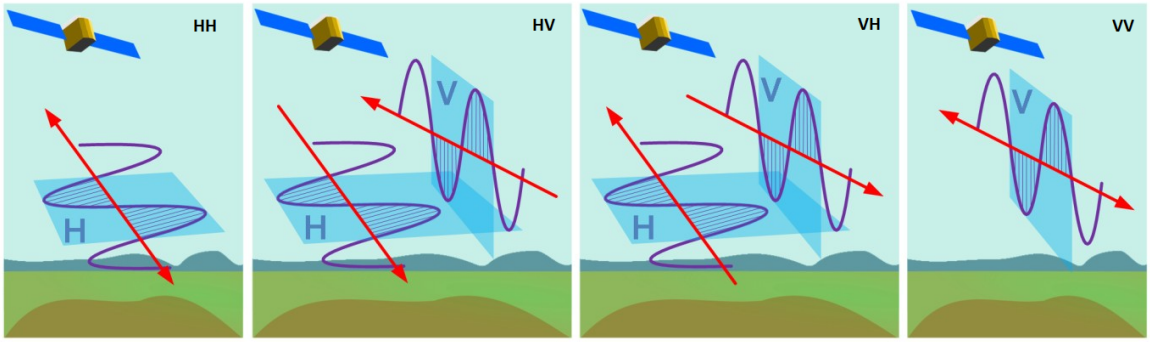


Figure 1.1: Illustration of four different polarization states of scattered waves, namely, HH, HV, VH and VV.

- VH - Transmitted V polarized wave and received H polarized wave,
- VV - Transmitted V polarized wave and received V polarized wave.

SAR typically works in three acquisition modes, i.e., single-pol, dual-pol, and full-pol. In single-pol, radar transmits only one wave (H or V) and receives only one backscattered wave (HH, HV, VH, or VV). Examples of single-pol satellites are SEASAT, ERS-1/2, RADARSAT-1, etc. In dual-pol mode, radar transmits one or two waves (H, V, or H&V) and receives two backscattered waves (HH and HV, VV and VH, or HH and VV). The examples of dual-pol satellites are ENVISAT, RISAT-1, SRTM, etc. In full-pol, also known as quad-pol or PolSAR, radar transmits two polarized waves and receives four backscattered signals. Examples of full-pol satellites are ALOS-PALSAR, RADARSAT-2, TerraSAR-X, etc. There is a trade-off between the amount of information and spatial/temporal resolution. Single-pol system has high spatial/temporal resolution but it provides less information. On the other hand, full-pol SAR provides load of information but has lower spatial/temporal resolution.

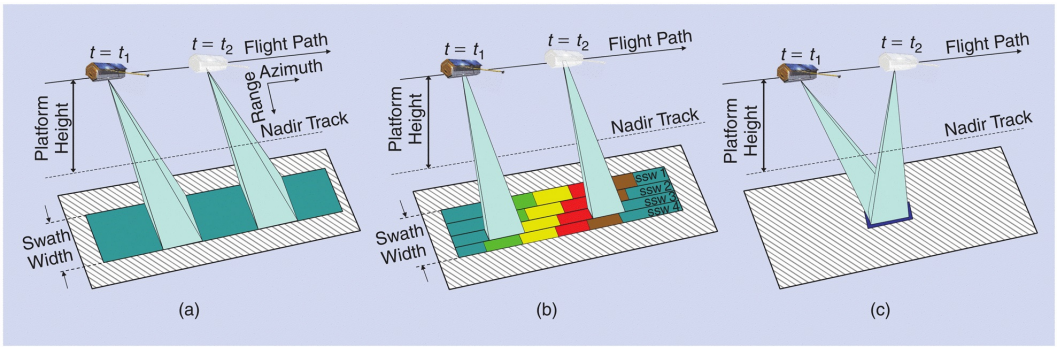


Figure 1.2: Different SAR imaging modes (a) Stripmap. (b) ScanSAR. (c) Spotlight [100].

Another critical aspect of data acquisition is imaging mode. The resolution of the SAR image also depends on the imaging mode of the radar. There are mainly three imaging modes, namely, Stripmap, Spotlight, and ScanSAR mode, as shown in Figure 1.2. In Stripmap mode, the antenna is fixed at a single position, and the area is scanned alongside the flight direction. It results in one continuous strip of the image. In ScanSAR mode, the radar beam is electronically steered at different elevations to obtain multiple swaths. It provides a larger coverage area compared to the Stripmap mode, but it has lower azimuth resolution. ScanSAR mode is typically used in applications where large area monitoring is required. For example, ice monitoring, ocean surface monitoring, flood mapping, etc. In Spotlight mode, we keep the object of interest under the radar beam by electronically steering the beam in the azimuth direction. The advantage of the Spotlight mode is that we get higher azimuth resolution. And the disadvantage is that we get less spatial coverage. Spotlight mode is typically used for high-resolution applications such as target detection and change detection.

1.2 Data Representation

In case of polarimetric synthetic aperture radar (PolSAR), horizontal (H) and vertical (V) polarized microwave signals are transmitted to be received in four combinations, i.e., HH, HV, VV, and VH. Hence a scattering matrix $\mathbf{S}$ for each pixel of the PolSAR image can be expressed as follows:

$$\mathbf{S} = \begin{bmatrix} S_{HH} & S_{HV} \\ S_{VH} & S_{VV} \end{bmatrix}. \tag{1.1}$$

Each element of this scattering matrix is a complex number representing amplitude and phase of a specific type of backscattered wave. In the case of diagonal terms of the scattering matrix, the polarization of transmitted and backscattered waves are the same. Hence, they are called co-polarized terms. The off-diagonal terms of the scattering matrix are called cross-polarized terms because they correspond to the opposite polarization of transmitted and backscattered waves. By nature scattering matrix $\mathbf{S}$ can only describe coherent targets, also known as a point or pure targets. To characterize incoherent or distributed targets, the second order descriptors can be used. Covariance and coherency matrices are examples of such second order descriptors. Under the reciprocity condition, it can be assumed that $S_{HV} = S_{VH}$. Hence $\mathbf{S}$ matrix can be represented in terms of lexicographic vector

$\mathbf{k}_l$ as:

$$\mathbf{k}_l = [S_{HH}, \sqrt{2}S_{HV}, S_{VV}]^{\mathsf{T}}. \tag{1.2}$$

Here, T is a transpose operation. Using lexicographic vector a second order descriptor, namely, covariance matrix can then be generated as:

$$\langle \mathbf{C} \rangle = \sum_{i=1}^{n} \mathbf{k}_{l_i} \mathbf{k}_{l_i}^{\dagger} = \begin{bmatrix} |S_{HH}|^2 & \sqrt{2}S_{HH}S_{HV}^{\dagger} & S_{HH}S_{VV}^{\dagger} \\ \sqrt{2}S_{HV}S_{HH}^{\dagger} & 2|S_{HV}|^2 & \sqrt{2}S_{HV}S_{VV}^{\dagger} \\ S_{VV}S_{HH}^{\dagger} & \sqrt{2}S_{VV}S_{HV}^{\dagger} & |S_{VV}|^2 \end{bmatrix}. \tag{1.3}$$

The '†' is a complex conjugate operation and $\langle\rangle$ indicates that the covariance matrix is multilook processed with a window of size n. Multilook processing is done by taking an average of covariance matrices in a window of size n centered at each pixel of a PolSAR image. n is also known as the number of looks.

1.2.1 Pauli decomposition

An alternate and more intuitive representation of scattering matrix $\mathbf{S}$ is obtained by Pauli decomposition [104]. According to Pauli decomposition, $\mathbf{S}$ can be expressed as:

$$\mathbf{S} = \alpha \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \beta \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} + \gamma \frac{1}{\sqrt{2}} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \tag{1.4}$$

where

$$\alpha = (S_{HH} + S_{VV})/\sqrt{2}, \tag{1.5}$$

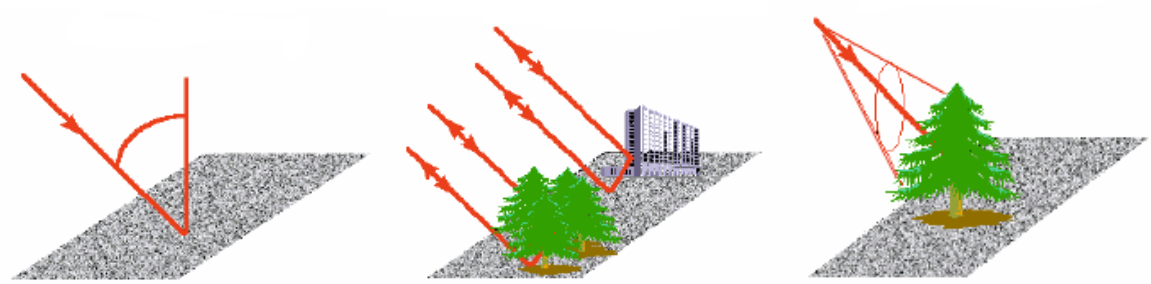
$$\beta = (S_{HH} - S_{VV})/\sqrt{2}, \tag{1.6}$$

$$\gamma = \sqrt{2}S_{HV}. \tag{1.7}$$

Using eq. (1.4), scattering matrix $\mathbf{S}$ can be represented as a complex Pauli vector $\mathbf{k}_p$:

$$\mathbf{k}_p = \frac{1}{\sqrt{2}}[S_{HH} + S_{VV}, S_{HH} - S_{VV}, 2S_{HV}]^{\mathsf{T}}. \tag{1.8}$$

Each element of this Pauli vector $\mathbf{k}_p$ is associated with one elementary scattering surface, namely, single bounce, double bounce, and volume scattering, as shown in Figure 1.3. Typically we observe single bounce from flat surfaces such as water or open field. In the case of single bounce scattering, both H and V waves observe 180° phase shift, so, $|\alpha|^2$ will be high. Double bounce is observed from dihedral structures such as buildings or the trunk of a tree. In the case of double bounce, the H wave observes a 360° phase shift while V wave observes a 180° phase shift. So,



(a) (b) (c)

Figure 1.3: Elementary scattering surfaces, namely, (a) single bounce scattering occurring due to flat surface, such as, pain field or water, (b) double bounce scattering occurring from the dihedral structure, such as, buildings or tree trunk, and (c) volume scattering occurring from complex structures, such as, tree canopy.

$|\beta|^2$ will be high. Finally, volume scattering is observed from a complex structure such as a vegetation area where the incident wave is scattered multiple times from the branches of the tree. In the case of volume scattering, both H and V waves will be depolarized due to the large number of bounces observed by both waves. So, $|\gamma|^2$ will be high. Now, the PolSAR image can be visualized as a color image by assigning blue color to $|\alpha|^2$, red color to $|\beta|^2$ and green color to $|\gamma|^2$. It is called as a Pauli decomposition image or PauliRGB image.

Pauli vector $\mathbf{k_p}$ can be used to form a complex coherency matrix $\langle \mathbf{T} \rangle$ for each pixel of the PolSAR image in the following way:

$$\langle \mathbf{T} \rangle = \sum_{i=1}^{n} \mathbf{k}_{p_i} \mathbf{k}_{p_i}^{\dagger} = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{12}^{\dagger} & T_{22} & T_{23} \\ T_{13}^{\dagger} & T_{23}^{\dagger} & T_{33} \end{bmatrix}. \tag{1.9}$$

The elements of matrix $\langle \mathbf{T} \rangle$ are related to that of $\mathbf{S}$ matrix in the following way:

$$\begin{aligned} T_{11} &= \frac{1}{2}|S_{HH} + S_{VV}|^2, \\ T_{12} &= \frac{1}{2}(S_{HH} + S_{VV})(S_{HH} - S_{VV})^{\dagger}, \\ T_{13} &= (S_{HH} + S_{VV})S_{HV}^{\dagger}, \\ T_{22} &= \frac{1}{2}|S_{HH} - S_{VV}|^2, \\ T_{23} &= (S_{HH} - S_{VV})S_{HV}^{\dagger}, \textit{ and} \\ T_{33} &= 2|S_{HV}|^2. \end{aligned} \tag{1.10}$$

The coherency matrix $\langle \mathbf{T} \rangle$ and the covariance matrix $\langle \mathbf{C} \rangle$ contain same informa-

tion. $\langle \mathbf{C} \rangle$ can be easily converted into $\langle \mathbf{T} \rangle$ as follows:

$$\langle \mathbf{T} \rangle = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & -1 \\ 0 & \sqrt{2} & 0 \end{bmatrix} \langle \mathbf{C} \rangle \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & \sqrt{2} \\ 1 & -1 & 0 \end{bmatrix}. \tag{1.11}$$

We note that the diagonal elements of a coherency matrix $\langle \mathbf{T} \rangle$ are real-valued, and off-diagonal elements are complex conjugates of each other. It is a Hermitian positive semidefinite matrix. It means that its eigenvalues and determinant are always positive and real-valued [131].

1.3 Target Decomposition (TD) Theorem

There are mainly two types of TD theorem, (i) Coherent decomposition and (ii) Incoherent decomposition. The coherent decomposition uses the scattering matrix to extract the polarimetric features and incoherent decomposition uses the covariance or coherency matrix for the feature extraction. The following two subsections describe both types of TD theorems.

1.3.1 Coherent Decomposition

Coherent decomposition decomposes scattering matrix $\mathbf{S}$ in a combination of the scattering responses from a simple target such as plane, dihedral, helix, etc.

$$\mathbf{S} = \sum_{i=1}^{k} \alpha_i \mathbf{S}_i. \tag{1.12}$$

Here $\mathbf{S}_i$ is a scattering response from a target, k is a number of elementary scatterers, and α_i is the weight coefficient of each scatterer. Pauli decomposition discussed in Chapter 1 is one type of coherent decomposition method in which the scattering matrix is decomposed into three basic scattering mechanisms, namely, single bounce, double bounce, and volume scattering. The Krogager decomposition [73] is another example of coherent decomposition. Here scattering matrix is decomposed into scattering from a sphere, oriented diplane and left or right helix. The coherent decomposition techniques are useful in characterizing coherent or point targets, but they can not describe incoherent or distributed targets. To describe the incoherent targets, we have to use second-order descriptors, namely, covariance or coherency matrix.

1.3.2 Incoherent Decomposition

To characterize the incoherent or distributed targets, we use second order descriptors such as 3×3 complex covariance matrix $\langle \mathbf{C} \rangle$ or coherency matrix $\langle \mathbf{T} \rangle$. Both of these representations contain the same information. Mathematically, incoherent decomposition can be described as:

$$\langle \mathbf{T} \rangle = \sum_{i=1}^{K} \alpha_i \langle \mathbf{T}_i \rangle. \tag{1.13}$$

Here α_i denotes the weight coefficient of a different component. The Freeman-Durden [38] decomposition method uses coherency matrix $\langle \mathbf{T} \rangle$ to model it as a three-component scattering mechanism, namely, volume scattering, double bounce scattering, and surface scattering.

$$\langle \mathbf{T} \rangle = F_{odd} * \langle \mathbf{T}_{surface} \rangle + F_{dbl} * \langle \mathbf{T}_{double} \rangle + F_{vol} * \langle \mathbf{T}_{volume} \rangle \tag{1.14}$$

Here F_{odd}, F_{dbl}, and F_{vol} correspond to weights associated with the surface, double bounce, and volume scattering respectively. The Yamaguchi decomposition [126] adds a helix scattering component too to the Freeman decomposition to model complicated human-made structures.

$$\langle \mathbf{T} \rangle = P_{odd} * \langle \mathbf{T}_{surface} \rangle + P_{dbl} * \langle \mathbf{T}_{double} \rangle + P_{vol} * \langle \mathbf{T}_{volume} \rangle + P_{hlx} * \langle \mathbf{T}_{helix} \rangle \tag{1.15}$$

Here P_{odd}, P_{dbl}, P_{vol} and P_{hlx} correspond to weights associated with surface, double bounce, volume and helix scattering respectively. The Huynen decomposition [59] aims to parameterize the coherency matrix as nine independent interpretable parameters. The Cloude-Pottier decomposition [20], also known as H/A/α decomposition, has used different approach for extracting polarimetric features from the coherency matrix. It uses the eigenvalue-eigenvector analysis of the coherency matrix to obtain seven polarimetric features. Three features are extracted from the eigenvalues of the coherency matrix, namely, Entropy (H),

Anisotropy (A) and mean magnitude (λ), as shown below:

$$\begin{aligned} H &= -\sum_{i=1}^{3} p_i log_3(p_i), \\ A &= \frac{\lambda_2 - \lambda_3}{\lambda_2 + \lambda_3}, \\ \lambda &= \sum_{i=1}^{3} p_i \lambda_i. \\ \text{where, } p_i &= \frac{\lambda_i}{\sum_{k=1}^{3} \lambda_k}, \end{aligned} \tag{1.16}$$

Here, λ_1, λ_2 and λ_3 are the eigenvalues of the coherency matrix. Entropy measures the degree of randomness in the observed scattering process. When the dominant scattering process is not present, meaning entropy is high, anisotropy gives relative importance of second and third eigenvalues. The next parameters of the Cloude-Pottier decomposition are extracted from the eigenvectors of the coherency matrix. The eigenvector of the coherency matrix is defined as:

$$\mathbf{u}_i = [cos(\alpha_i),\ sin(\alpha_i)cos(\beta_i)e^{j\delta_i},\ sin(\alpha_i)cos(\beta_i)e^{j\gamma_i}]^{\mathsf{T}}. \tag{1.17}$$

Here α, β, γ and δ are the parameters that completely describe the eigenvectors. Mean of these angles is useful in discriminating different types of the scattering processes. The mean angles are defined in the following equation:

$$\begin{aligned} \alpha &= \sum_{i=1}^{3} p_i \alpha_i, \\ \beta &= \sum_{i=1}^{3} p_i \beta_i, \\ \gamma &= \sum_{i=1}^{3} p_i \gamma_i, \ and \\ \delta &= \sum_{i=1}^{3} p_i \delta_i. \end{aligned} \tag{1.18}$$

1.4 Classification Problem

Classification is a problem where we intend to divide data into different categories or classes based on given criteria. A simple example of a classification problem is spam mail detection. Here we intend to classify the given mail into two classes,

namely, spam and not spam. To teach the machine how to do classification, we require data consisting of samples with known classes, called dataset. The dataset with N samples is described as $\mathbf{D} = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}$. Here $\mathbf{x}_i$ is i^{th} sample and $\mathbf{y}_i$ is its corresponding class, known as label. There are mainly two methods for classification: supervised and unsupervised. In the case of a supervised classification, we have a labelled dataset, which means label information of all samples are known. In the unsupervised case, we have an unlabelled dataset, meaning no information regarding the sample's label is given. The dataset is typically divided into three parts with varying proportions of samples, e.g., training set, validation set, and test set. The training set is used to train the machine for the given task. The validation set is used to validate the performance of the model during training. Once the training is done, the test set is used to measure the algorithm's performance to classify the unseen data based on the training. The performance of the given classification algorithm can be measured using metrics such as Overall Accuracy (OA) or Average Accuracy (AA) [25]. Let l_m^i be the true label of the i^{th} test sample of class m and let $\hat{l_m^i}$ be the predicted label of i^{th} test sample of class m. Then the OA and AA are calculated as below:

$$
\begin{aligned}
\textit{Classwise Accuracy } (a_m) &= \frac{1}{n_m} \sum_{i=0}^{n_m} \mathbb{I}(l_m^i = \hat{l_m^i}), \\
\textit{Average Accuracy } (AA) &= \frac{1}{M} \sum_{i=0}^{M} a_m, \\
\textit{Overall Accuracy } (OA) &= \sum_{i=0}^{M} \left(\frac{n_m}{N} a_m \right).
\end{aligned}
\tag{1.19}
$$

Here, $\mathbb{I}()$ is an indicator function with $\mathbb{I}(l_m^i = \hat{l_m^i}) = 1$ when $l_m^i = \hat{l_m^i}$, else 0. n_m is the number of test samples in class m. $N = \sum_{i=1}^{M} n_m$ is the total number of test samples. M is the number of classes. As we can see from the eq. (1.19), class-wise accuracy (a_m) calculates percentage of test samples identified correctly in class m. AA is simply an average of class-wise accuracies. On the other hand OA is the weighted average of class-wise accuracies where weight of a particular class is determined by the number of test samples in that class. OA might be misleading with highly imbalanced classes. The other important classification metrices are precision (P), recall (R) and f1 score. A precision score of a class m says that out of all samples labelled as m by the algorithm, how many indeed belong to class m. But it does not say anthing about the number of samples from class m that were not labelled correctly. On the other hand recall score of class m says that how may samples of class m were correctly labelled as m by the algorithm. But it does

not say nothing about how many samples from other classes were incorrectly also labelled as m. The F_1 score is the harmonic mean which combines precision and recall into one measure [17]. Precision, recall and F_1 score for each class can be computed as below:

$$\begin{aligned} P_m &= \frac{TP_m}{TP_m + FP_m}, \\ R_m &= \frac{TP_m}{TP_m + FN_m}, \\ F_1 &= 2\frac{P_m * R_m}{P_m + P_m}. \end{aligned} \tag{1.20}$$

. Here,

- True Positive (TP_m): Number of samples predicted as m where true label is also m.
- False Positive (TP_m): Number of samples predicted as m but true label is not m.
- False Negative (FN_m): Number of samples not predicted as m but true label is m.

PolSAR image classification is a pixel-wise classification problem where we intend to assign a label to each pixel of the given PolSAR image. In the case of the supervised classification, we require training samples with known labels. Such training samples can be obtained by field survey, during which parameters such as type of crop, height, biomass, water content, etc., are collected at different points over the area of interest synchronous with satellite pass. A GPS location of such a sample is also collected, which allows us to register the surface observation with the image's pixel. A ground truth map can be generated from the field survey data or by combining it with visual and statistical analysis of the SAR image data or the optical image data. Such a ground truth map allows us to map the pixel of the image with the ground reality (label of a pixel). Mathemetically ground truth map is defiend as $g : S \rightarrow \Sigma$ where S is a grid and Σ is a set of classes [10].

Over the years, different methods for PolSAR image classification have been proposed. In the early days, parametric classifiers that model the probability distribution of the data were popular. Recently, due to advancements in the field of machine learning, neural network based methods have come to fore. In the following sections, we briefly describe different approaches used for the PolSAR image classification.

1.4.1 Parametric Classifiers

One of the very early approaches of PolSAR image classification was presented by Kong *et al*. [70]. It modeled the lexicographic vector $\mathbf{k_l}$ as a complex Gaussian distribution with probability density function given below:

$$p(\mathbf{k}_l) = \frac{1}{\pi^n |\langle \mathbf{C}_m \rangle|} \exp\{-\mathbf{k}_l^\dagger |\langle \mathbf{C}_m \rangle|^{-1} \mathbf{k}_l\}. \tag{1.21}$$

Using maximum likelihood ratio test the optimum classifier was derived with following distance measure:

$$d(\mathbf{k}_l, \langle \mathbf{C}_m \rangle) = \mathbf{k}_l^\dagger \langle \mathbf{C}_m \rangle^{-1} + \ln |\langle \mathbf{C}_m \rangle|. \tag{1.22}$$

Here $\langle \mathbf{C}_m \rangle$ is the mean coherency matrix of class m. Lee *et al*. extended this idea by using the second order descriptor [78], namely, coherency matrix or covariance matrix. Let $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$ be n independent $d \times 1$ multivariate gaussian random variable with zero mean and $\mathbf{\Sigma}$ covariance matrix. Let us define,

$$\mathbf{X} = \sum_{i=1}^{n} \mathbf{v}_i \mathbf{v}_i^\dagger. \tag{1.23}$$

Then, $\mathbf{X}$ has a Wishart distribution with n degrees of freedom [47]. Assuming that the lexicographic vectors $\mathbf{k}_l$ or Pauli vector $\mathbf{k}_p$ are independent and identically distributed complex Gaussian random vectors, $n\langle \mathbf{T} \rangle$ follows complex Wishart distribution. The probability density function of Wishart distribution is given as follows:

$$\begin{aligned} &p(\langle \mathbf{T} \rangle | \langle \mathbf{C}_m \rangle) = \frac{n^{nd} |\langle \mathbf{T} \rangle|^{(n-d)}}{\Gamma_p(n) |\langle \mathbf{C}_m \rangle|^n} \exp(-\text{trace}(\langle \mathbf{C}_m \rangle^{-1} \langle \mathbf{T} \rangle)), \\ &\text{where,} \\ &\Gamma_p(n) = \pi^{\frac{p(p-1)}{2}} \prod_{j=1}^{p} \Gamma(n - j + 1). \end{aligned} \tag{1.24}$$

Here $\langle \mathbf{T} \rangle$ is the coherency matrix of a PolSAR pixel. Using this probability density function, Lee *et al*. derived a distance measure using Maximum Likelihood Estimation (MLE) called Wishart distance [78]. Wishart distance between any two coherency matrices is defined as:

$$D_{Wishart}(\langle \mathbf{T} \rangle, \langle \mathbf{C}_m \rangle) = \text{trace}(\langle \mathbf{C}_m \rangle^{-1} \langle \mathbf{T} \rangle) + \ln |\langle \mathbf{C}_m \rangle|. \tag{1.25}$$

To understand how classification is performed using these distance measures, let's assume we have a PolSAR image of a vast agriculture farm. Let's consider four types of cultivated crops. Each of these crops represents one class. Each class might have thousands or millions of pixels in the PolSAR image. The mean of certified coherency matrices of a class is known as a class or cluster center. A cluster is defined as a set of samples that are closely related to each other. Now, to classify a pixel with an unknown label, we need to find out how distant it is from all those four class centers. The pixel will belong to the class for which its distance is found to be the minimum. If we use Wishart distance, then this classification is called the Wishart classification.

Wishart distance is widely used for both supervised and unsupervised classification of PolSAR images [78, 80, 124]. Wishart classifier assumes that each class of the PolSAR image is homogeneous. In that case each class is described as one cluster center only. But practically this does not happen. A class is usually heterogeneous. If the class is heterogeneous, then there can be multiple cluster centers in each class. To handle such heterogeneous classes, Gao *et al*. [43] proposed Wishart mixture model where each class is modelled as a mixture of Wishart distributions. The Wishart mixture model parameters were estimated using the Expectation-Maximization (EM) algorithm. The coherency matrix can also be modelled as multivariate K-distribution [130], G-distribution [39], Scale mixture of Gaussian (SMoG) [30], etc. Higher order statistics are also proven to be useful for analysis and interpretation of PolSAR images [24].

1.4.2 Artificial Neural Networks

Artificial Neural Network (ANN) based algorithms are inspired by the human brain. An ANN is a layered network of processing units call neurons which perform elementary mathematical operations. These neurons are connected with each other, and each connection carries a weight. Collectively, these weights represent the information learned by the ANN. Figure 1.4 shows a typical architecture of an ANN. An ANN typically consists of three types of layers, namely, input layer, hidden layer, and output layer. Through the input layer, we feed our training samples $\mathbf{x}$ to the ANN. It will be multiplied by the weights of the connections and will become the input to the next layer. The task of the hidden layer is to perform a nonlinear transformation on the input. It is achieved by using an activation function. The activation function gives the ANN the ability to learn nonlinear relationships between samples and labels. Training of ANN is done in two parts, namely, forward pass and backward pass. During forward, we feed out training

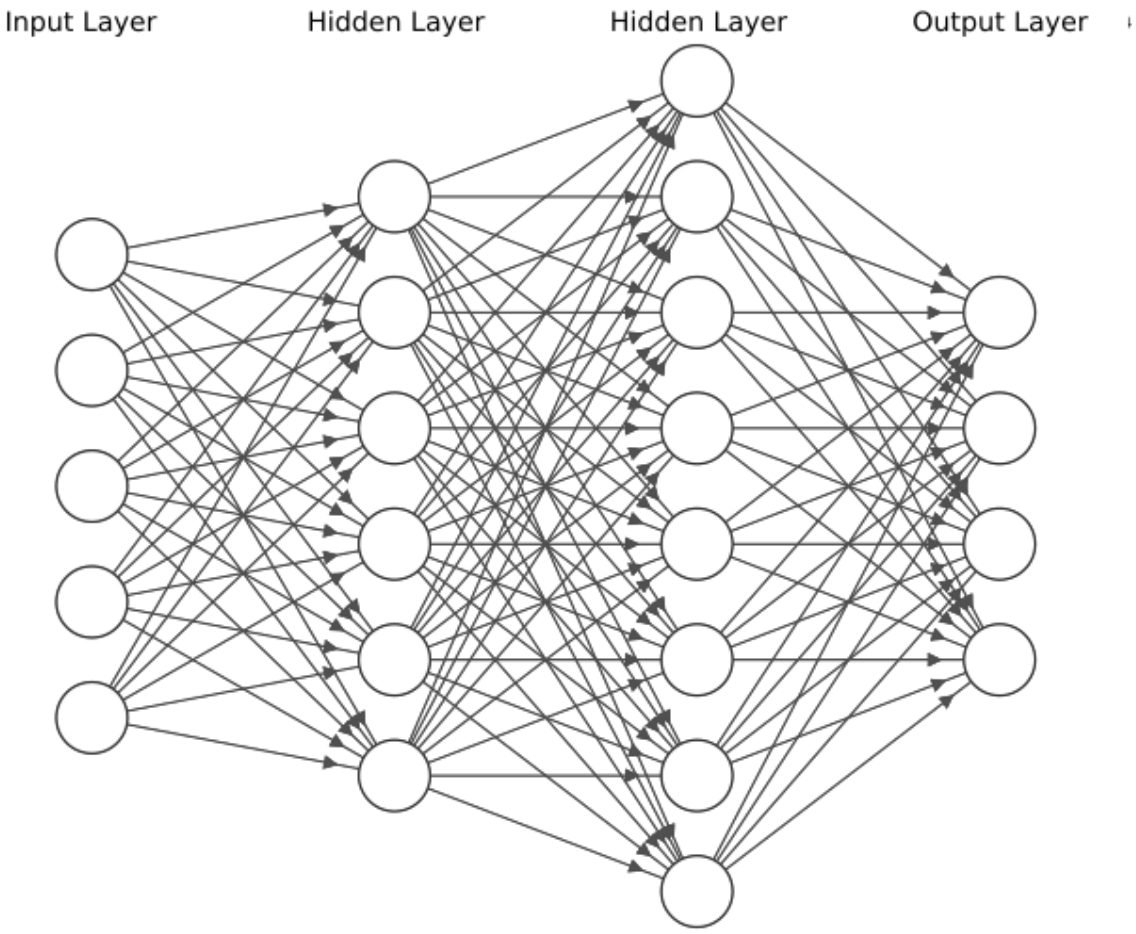


Figure 1.4: Architecture of an ANN containing 3 fully connected layers.

samples to the ANN, after passing through multiple hidden layers, the ANN produces its predicted label through the output layer. The forward pass of ANN with n layers can readily be described as eq. (1.26) mathematically.

$$\begin{aligned} \mathbf{h}_1 &= g\left(\mathbf{W}_1^{\mathsf{T}}\mathbf{x} + \mathbf{b}_1\right), \\ &\vdots \\ \mathbf{h}_k &= g\left(\mathbf{W}_k^{\mathsf{T}}\mathbf{h}_{k-1} + \mathbf{b}_k\right), \\ &\vdots \\ \bar{\mathbf{y}} &= \left(\mathbf{W}_n^{\mathsf{T}}\mathbf{h}_{n-1} + \mathbf{b}_n\right). \end{aligned} \tag{1.26}$$

Here $\mathbf{h}_1$ is the output of the first hidden layer, $\mathbf{h}_k$ is the output of the k^{th} hidden layer and $\bar{\mathbf{y}}$ is the label predicted by the ANN. $\theta = \{\mathbf{W}_1, \ldots, \mathbf{W}_n, \mathbf{b}_1, \ldots, \mathbf{b}_n\}$ are the parameters of the ANN. These parameters are randomly initialized. $g()$ is an

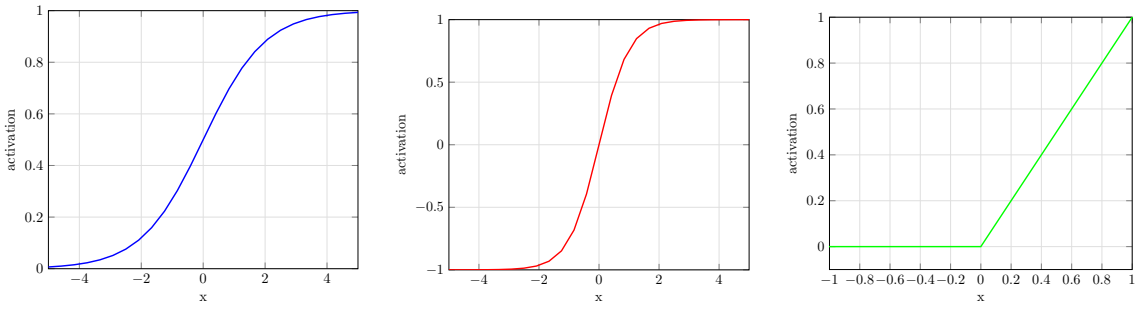


Figure 1.5: Illustrating different activation functions, namely, (a) *sigmoid*, (b) *tanh*, and (c) *ReLU*.

activation function. Some popular activation functions such as *sigmoid*, *tanh* and *ReLU* are shown below:

$$\begin{aligned} sigmoid(x) &= \frac{1}{1+e^{-x}}, \\ tanh(x) &= \frac{e^{x}+e^{-x}}{e^{x}-e^{-x}}, \\ ReLU(x) &= max\,(0,x)\,. \end{aligned} \tag{1.27}$$

Figure 1.5 depicts activation functions given in eq. (1.27). Once the forward pass is complete, we measure the difference between the predicted labels and true labels of all training samples. This difference is represented in the form of a function called loss function or cost function. The ANN's goal is to reduce the value of this cost function, such as Mean Squared Error (MSE) as shown below:

$$J(\theta) = \frac{1}{N}\sum_{i=0}^{N}\|\mathbf{y}_i - \bar{\mathbf{y}}_i\|^2, \tag{1.28}$$

where $\mathbf{y}_i$ is the true label of i^{th} training sample and $\hat{\mathbf{y}}_i$ is the predicted label of the i^{th} training sample. During the backward pass, parameters of the network are updated using gradient decent based learning algorithms. Some popular algorithms are: Stochastic Gradient Decent (SDG) [67], Adaptive Gradient (AdaGrad) [31], Root Mean Square Propagation (RMSprop) [118], Adam [68]. The flexible structure of ANN allows to model complex functions which is difficult to do by parametric classifiers.

Pottier and Saillard presented [105] one of the first neural network based supervised PolSAR image classification approach. It uses a single hidden layer ANN with a *sigmoid* activation function and MSE loss function. The elements of the

Muller matrix were used as an input to the ANN. Muller matrix is defined as:

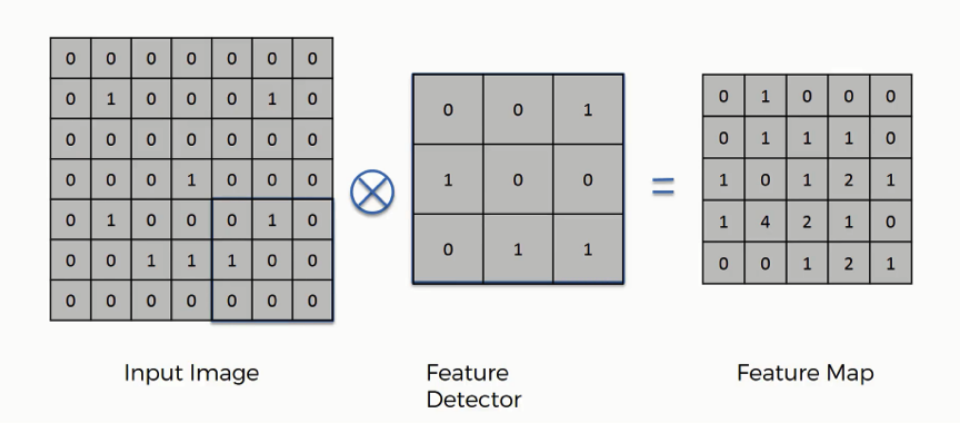
$$\mathbf{M} = \frac{1}{2}\begin{bmatrix} T_{11}+T_{12}+T_{13} & T_{12}+T_{12}^{\dagger} & T_{13}+T_{13}^{\dagger} & -i(T_{23}-T_{23}^{\dagger}) \\ T_{12}+T_{12}^{\dagger} & T_{11}+T_{22}-T_{33} & T_{23}+T_{23}^{\dagger} & -i(T_{13}-T_{13}^{\dagger}) \\ T_{13}+T_{13}^{\dagger} & T_{23}+T_{23}^{\dagger} & T_{11}-T_{22}+T_{33} & -i(T_{12}-T_{12}^{\dagger}) \\ -i(T_{23}-T_{23}^{\dagger}) & -i(T_{13}-T_{13}^{\dagger}) & -i(T_{12}-T_{12}^{\dagger}) & -T_{11}+T_{22}+T_{33} \end{bmatrix}. \tag{1.29}$$

Next, ANN was applied for unsupervised classification [53] and it was found to be better than the conventional approaches including k-means algorithm. Over the years, ANN rose in popularity [18, 54, 61, 120] but the limited computational power of early computers prohibited training of large ANNs with multiple hidden layers. Due to the recent advancements in the CPU and GPU technology, training of large neural networks has become easy. Many deep neural network based architectures containing multiple hidden layers have been proposed, such as Convolutional Neural Networks (CNN) [137], Autoencoder (AE) [55], Recurrent Neural Networks (RNN), Generative Adversarial Networks (GAN) [87, 115], etc.

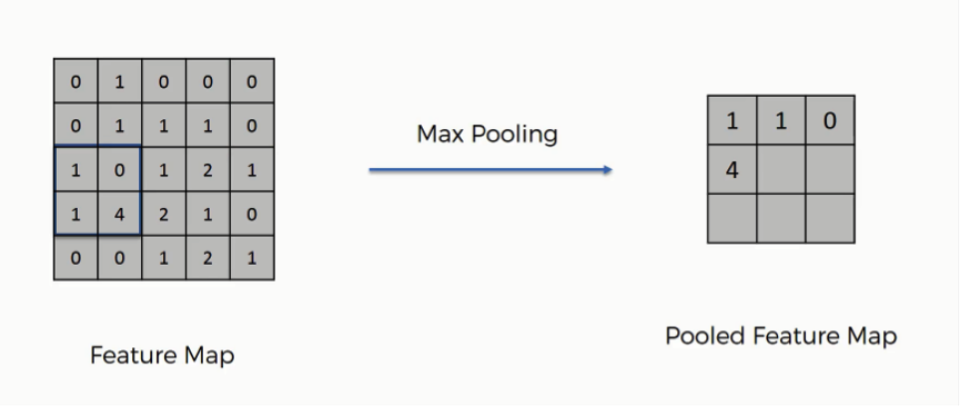
In a different approach for ANN based classification of a PolSAR image, Jiao and Liu have recently proposed Wishart Network (WN) and Wishart Deep Stacking Network (W-DSN) [63]. In these methods, the weights of the network are initialized using the statistical properties of the PolSAR image. Unlike other deep learning architectures, WN and W-DSN both have a fast convergence rate due to better initialized weights. As the name suggests, WN and W-DSN pick Wishart distance [78] in order to initialize the desired weights of the network.

1.4.3 Convolutional Neural Networks

A multilayer neural network described in the previous section can be used for computer vision applications. But it has some limitations. Considering a typical image size of 1024×1024, the input vector size will be 1048576×1, which is extremely large. Also, natural signals such as audio, image and text exhibit properties of stationarity and locality. In case of images, locality means that only pixels in a small neighborhood are related, and stationarity means that same patterns in the image are repeated. Due to fully connected layers of ANN, it is not able to capture these properties of a natural signal. Thus, it can not achieve invariance in translation, rotation, and scale of an object in the image. CNN can solve these problems by employing two critical operations, namely, convolution and pooling. Because of that, CNN based architectures have achieved remarkable results



(a)



(b)

Figure 1.6: (a) Convolution operation and (b) Pooling operation.

in the field of computer vision [60, 95, 107, 116]. Kunihiko Fukushima proposed one of the first CNN based architectures in 1980 [40]. The network was inspired by the experiments of Hubel and Wiesel on receptive fields in the visual cortex of cats [58]. In 1989, Yann LeCun successfully trained parameters of CNN using back-propagation and applied it to Zip Code recognition [77]. The advancement in the field of Graphical Processing Unit (GPU) made the training of large CNN based architecture possible. In 2012 AlexNet was proposed by Alex Krizhevsky *et al*. [72]. It won the ImageNet competition [23] by achieving 10% lower error than the second best model.

Figure 1.6(a) shows how a convolution operation works. We slide the weight matrix called kernel over the input image, and at each step, we multiply elements of the kernel with the overlapping pixels on the input image and sum the products. The output of the convolution operation is called a feature map. As we can see, compared to the fully connected layer, a convolution operation needs a very small number of weights, which are shared among the input image. The con-

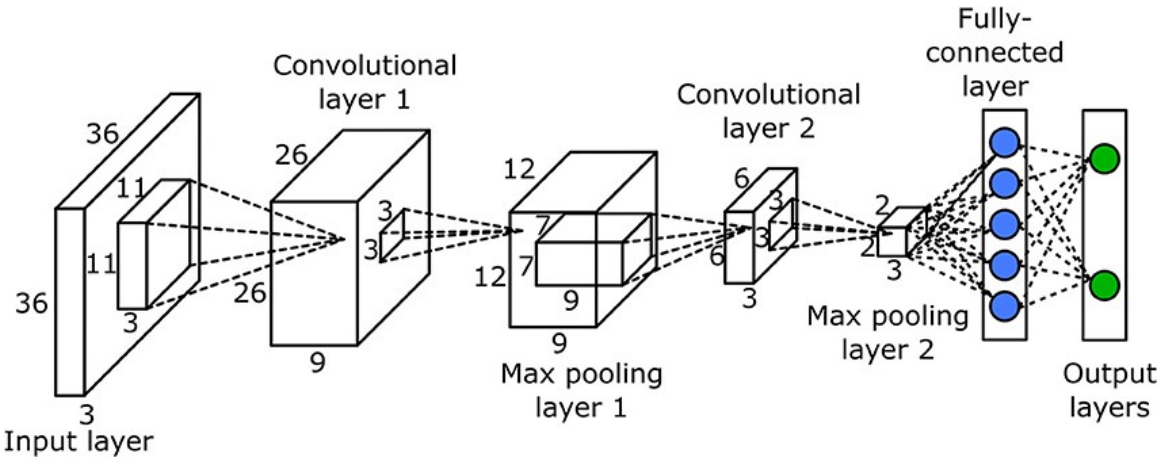


Figure 1.7: A typical CNN architecture containing two modules of convolution and pooling operations followed by two fully connected layers.

volution operation allows us to exploit the locality and stationary properties of the image. Figure 1.6(b) describes the max-pooling operation. We slide the fixed-sized window over the feature map, and at each step, we select the largest value from the overlapping elements of the feature map. To some degree, max-pooling operations make CNN translation invariant. A typical CNN, as shown in Figure 1.7, contains multiple modules of the convolution and pooling stacked together. The lower level modules typically learn the low level features from the image, and the higher level module builds on top of the low level features to learn the high level features.

To use CNN for PolSAR image classification problem, the input image is divided into small patches centered at each pixel. The patch is typically of size 12×12. These patches are then used for training the CNN. One of the first CNN based architecture for PolSAR images was proposed by Zhou *et al*. in 2016 [137]. It uses normalized elements of the coherency matrix as an input to the network. Sun *et al*. then extended it to a dual-branch CNN by adding a Pauli decomposition image as a second branch [42]. Both of these networks use real numbers as an input, but PolSAR images are represented by complex-numbers to incorporate phase information. To use phase information, a complex-valued CNN was proposed [136] where both inputs and weights of the network are represented as complex numbers. Compared to real-valued CNN, complex-valued CNN was able to achieve higher classification accuracy while using less number of parameters. In an another approach Liu *et al*. proposed a Polarimetric Convolutional Network (PCN) [94] to incorporate phase information. It uses sparse scattering coding to represent a complex number as a 2×2 real number matrix. To further improve the performance of CNN, handcrafted features that describe the physical

scattering property of the target can be used. These types of features are typically difficult to learn by the network itself. A polarimetric-feature-driven CNN was proposed which used roll-invariant and rotational features extracted from the coherency matrix [19]. It showed that CNN trained using handcrafted features could achieve better classification accuracy in comparison with CNN trained only using elements of coherency matrix. Yang *et al*. extended this idea by extracting 70 handcrafted features from the coherency matrix and used CNN to identify useful features [127].

1.4.4 Autoencoders

An Autoencoder (AE) is an unsupervised learning method for efficient data representation. Typically AE is used for dimensionality reduction. When the input vector's dimension is very large, AE can first be used to represent input into lower dimension and then to do the classification. AE consists of two modules, namely, encoder and decoder. The job of encoder is to map the input to the lower dimension representation vector. The job of a decoder is to reconstruct the original input vector from the representation vector. Many variants of AE are reported in the literature for different applications such as Denoising AE (DAE) [121] which is used to remove noise from the input, sparse AE (SAE) [57] which is used to represent input into sparse higher dimension, Contractive AE (CAE) which adds contractive term in the cost function of AE to enforce minimal variance in the representation vector [108], Variational AE (VAE) [69], Convolutional AE (ConvAE) etc. Figure 1.8 shows an architecture of a typical AE module. Input to the AE is our sample, and the output of the AE is the reconstructed sample. The goal of the AE is to minimize the error between the input sample and the reconstructed sample.

Hou *et al.* [55] presented one of the earliest multilayer AE based network for PolSAR image classification. The elements of the coherency matrix were used as an input to the two layer AE based network. To clean the prediction of the network, the image was divided into superpixels. Prediction of the network was then used to run a K Nearest Neighbors (KNN) algorithm in each superpixel to determine the class of the complete superpixels. A superpixel is an over-segmentation technique that divides an image into multiple homogeneous regions. Geng *et al*. proposed a different approach of using superpixels with AE by introducing a superpixel restrained term in the cost function of the network. It allows AE to use the spatial information from the superpixels to reduce the effect of speckle noise. The autoencoder network was also used where superpixel retained error term

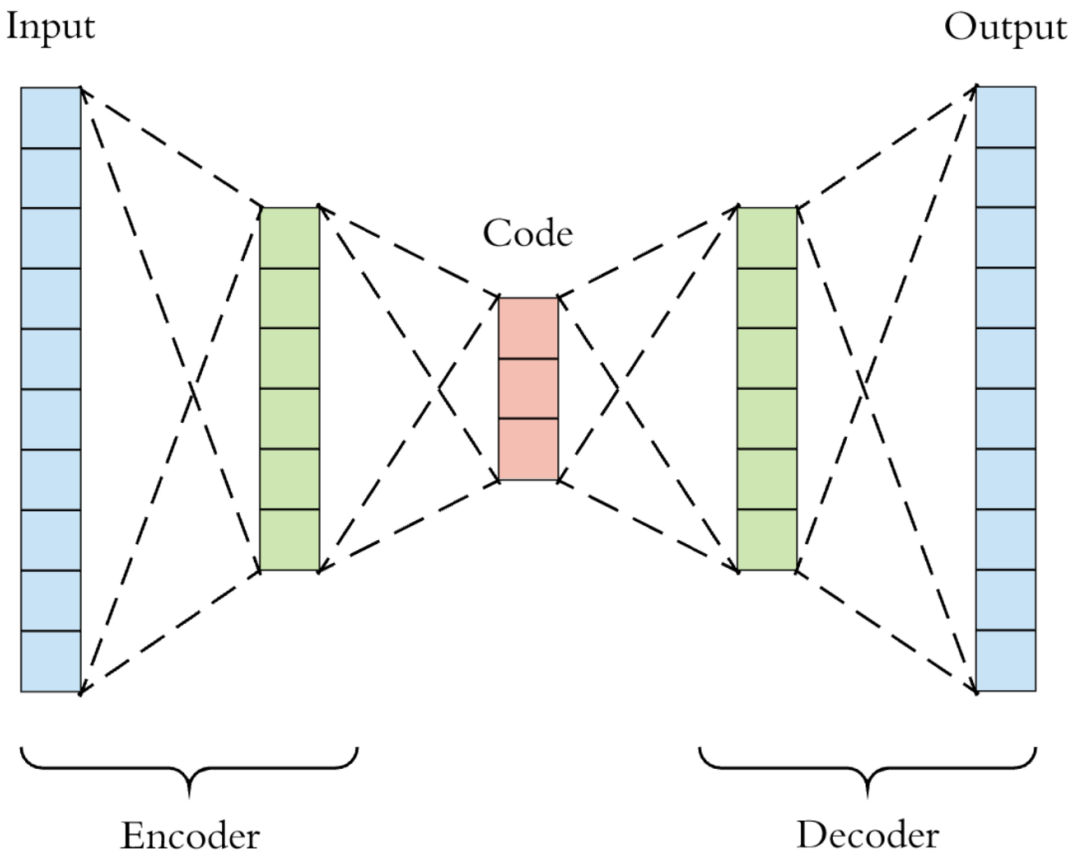


Figure 1.8: A typical AE architecture.

forced the network to consider homogeneous nature of superpixels [45]. Zhang *et al*. [132] proposed stack sparse autoencoder based network which incorporates local information to reduce the effect of speckle noise. The local information is extracted from the neighboring pixels located in the window centered at each pixel. The distance of the neighboring pixel to the center pixel of the window is used to weight the influence. If the pixel is heavily infected by the speckle noise then the local information may lead to misclassification [62]. To address this issue, Yuanyuan *et al*. [57] extended this idea by introducing adaptive nonlocal approach for extracting spatial information.

1.5 Challenges in SAR Image Classification

SAR images heavily suffer from the speckle noise. Speckle is a granular noise-like artifact that exists in the images taken by active imaging sensors such as SAR. Theoretically, speckle is not noise; it arises from the constructive and destructive interference of scattered waves in a resolution cell. Scattering matrix of a Pol-SAR pixel is observed from a resolution cell containing multiple scatterers. As

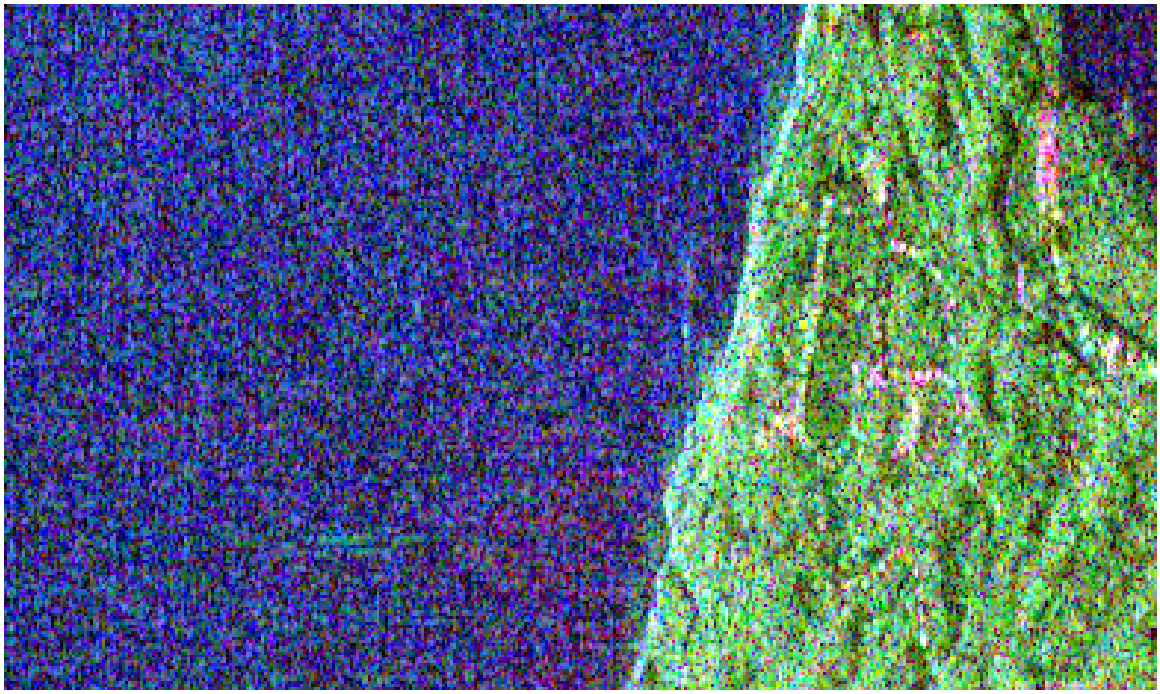


Figure 1.9: A PolSAR image suffering from the speckle noise. Constructive scattering is seen as bright spots and destructive scattering is seen as dark spots in the image.

shown in Figure 1.9, if the interference in the resolution cell is constructive, that PolSAR pixel appears bright. If the interference is destructive, then the pixel appears dark. Since it is an undesired phenomenon, speckle is considered as noise. Speckle noise can make the classification task difficult due to its multiplicative nature [96]. Since we are doing a pixel-wise classification, pixel suffering from a speckle noise may easily be misclassified. To reduce the effect of speckle on classification, it is necessary to filter the SAR image. The ideal speckle reduction filter reduces speckle with minimum loss of information and protects the image's structural features. Some notable speckle noise reduction methods can be found in the literature [29, 79, 91, 111, 129].

As discussed in section 1.2, each pixel of a PolSAR image is represented by a 3×3 complex-valued matrix. Typical operations required to classify each pixel using parametric classifier include matrix inverse, multiplication, determinant, etc. These operations are computationally expensive and considering the large size of remote sensing images, pixel-wise classification becomes a time consuming process. Some efforts to reduce the computational complexity are being made by either reducing the number of pixels that need to be classified [49, 89, 113] or by representing coherency matrix as a complex-valued vector [32, 63, 135].

Phase contains very useful information in PolSAR image [102]. The shift in

a phase of vertically polarized waves allows us to differentiate between a single bounce and double bounce scattering. How to effectively utilize the phase information is a challenging task for deep learning based architectures. A complex-valued neural network based architectures is required to use the phase information [16,50,136].

1.6 Our Proposed Approaches

- **Optimized Wishart Network (OWN):** Assuming that the lexicographic vectors $\mathbf{k}_l$ or Pauli vector $\mathbf{k}_p$ are independent and identically distributed complex Gaussian random vectors, the coherency matrix and covariance matrix follows a complex Wishart distribution. The calculation of Wishart distance, mathematically, is primarily the calculation of complex matrix inversion and multiplication. The linearization model suggested by Jiao and Liu [63] converts a matrix multiplication into a vector dot product. Hence it saves computational time. But it is to be noted that even this vector dot product calculation requires multiplication and addition of complex quantities. First, we propose a real-valued vector representation for each PolSAR pixel. Our proposal retains phase information while reducing computational time significantly as now multiplication and addition are not of complex quantities but of real numbers. Therefore it improves the response time of Wishart distance calculation and hence of classification efficiency. Next, we propose a better parameter selection for increasing the number of hidden units in the Wishart network introduced in [63]. We will show that it will result in considerable improvement in classification accuracy. Combining it with real-valued vector representation, we show that our proposed OWN is more efficient in terms of classification time and accuracy.

- **Extended OWN (e-OWN):** We propose Extended Optimized Wishart Network for classification of multifrequency PolSAR images. The proposed network has three layers: the input layer, hidden layer, and output layer. The hidden layer and output layer are fully connected, while the input layer and the hidden layer are partially connected. We choose partial connections because we observe that a fully connected input-hidden layer does not provide any additional advantage in terms of classification accuracy. After all, coherency matrices of two different frequency bands are uncorrelated [78]. On the other hand partially connected input-hidden layer will reduce the number of parameters to learn. Hence it reduces computation overhead without

sacrificing classification accuracy.

- **Superpixel Driven OWN:** We propose a fast PolSAR image classification algorithm that uses pixel-based features and spatial features around each pixel. This is achieved by introducing a superpixel driven optimized Wishart network. The first improvement suggested is to take advantage of a fast global *k*-means algorithm for obtaining optimal cluster centers within each class. It uses a real-valued vector representation of the coherency matrix along with fast matrix inverse and determinant algorithms to reduce computational overhead. Our method then exploits the information of neighboring pixels by forming a superpixel, so that even a noisy pixel may not be assigned a wrong class label. The proposed network uses dual branch architecture to combine pixel and superpixel features efficiently.

- Finally, we are proposing deep learning based classification algorithm for multifrequency PolSAR image. Using PolSAR image decomposition algorithms, 33 features are extracted from each frequency band of the given PolSAR image. Then, a two-layer autoencoder is used to reduce the dimensionality of the input feature vector while retaining useful features of the image. This reduced dimensional feature vector is then applied to generate superpixels using a simple linear iterative clustering algorithm. Next, a robust feature representation is constructed using both pixels as well as superpixel information. Finally, a softmax classifier is used to perform the classification task.

1.7 Organization of Thesis Chapters

We discuss each of our above mentioned contributory works in separate chapters. Chapter 2 discusses the real-valued vector representation of the PolSAR pixel coherency matrix followed by our proposed optimized Wishart network. We discuss extended-OWN for classification of multifrequency PolSAR data in Chapter 3. In Chapter 4 we propose faster calculation of inverse and determinant of a coherency matrix along with fast global *k*-means algorithm for the PolSAR data. Chapter 5 describes superpixel driven OWN, which uses superpixels to incorporate spatial information into the classification process. In Chapter 6 we elaborate stacked autoencoder based feature extraction and superpixel generation for multifrequency PolSAR image classification. Finally the thesis is concluded in Chapter 7.

CHAPTER 2
Optimized Wishart Network (OWN)

2.1 Background

Lee *et al*. showed that assuming the lexicographic vectors $\mathbf{k}_l$ or Pauli vector $\mathbf{k}_p$ are independent and identically distributed complex Gaussian random vectors, the coherency matrix and covariance matrix has a complex Wishart distribution. Using this we can derive a distance measure using MLE called Wishart distance [78]. The Wishart distance between any two coherency matrices is defined as:

$$D_{Wishart}(\langle \mathbf{T} \rangle, \langle \mathbf{C}_m \rangle) = \text{trace}(\langle \mathbf{C}_m \rangle^{-1} \langle \mathbf{T} \rangle) + \ln |\langle \mathbf{C}_m \rangle|. \tag{2.1}$$

Here $\langle \mathbf{C}_m \rangle$ is the cluster center of class m and $\langle \mathbf{T} \rangle$ is a coherency matrix of a test pixel. This equation says that to know the Wishart distance of a pixel from a cluster center, one is supposed to do the following:

- Calculate the inverse of the coherency matrix $\langle \mathbf{C}_m \rangle$
- Do a matrix multiplication of $\langle \mathbf{C}_m \rangle^{-1}$ with the coherency matrix $\langle \mathbf{T} \rangle$
- Calculate the determinant of $\langle \mathbf{C}_m \rangle$.

This is to note that the both matrices are complex in general. Hence the multiplication of the two may become computationally expensive. Jiao and Liu [63] observed that the calculation of Wishart distance, mathematically, is primarily the calculation of the trace of the two matrices. Using this fact, a linearization model is suggested which converts a matrix multiplication into a vector dot product. Hence it saves computational time. With help of this linear implementation, they designed the Wishart Network (WN). WN is a single hidden layer neural network. Weights of WN are initialized using the linearization of coherency matrix such that during the forward pass, WN calculates Wishart distance of all training samples with all cluster centers. In the next section, we will describe WN including some of its limitations.

2.2 Wishart Network

Let $\langle \mathbf{T_i} \rangle$ be the coherency matrix of i^{th} pixel of a PolSAR image and $\langle \mathbf{C}_m \rangle$ be the mean coherency matrix of class m. To perform Wishart classification, we have to calculate Wishart distance shown in eq. (2.1) between every PolSAR pixel $\langle \mathbf{T_i} \rangle$ and every cluster center $\langle \mathbf{C}_m \rangle$. A single Wishart distance calculation requires a 3×3 complex matrix inverse, multiplication, and determinant operation. A single 3×3 complex matrix multiplication requires 27 complex number multiplications and 20 complex number additions. Since we require trace of the $\langle \mathbf{C}_m \rangle^{-1} \langle \mathbf{T} \rangle$, the number of computations can be reduced by only calculating the diagonal entries of the resultant matrix. Let $\sigma = f(\mathbf{\Sigma})$ be a function which transforms any coherency matrix $\langle \mathbf{T} \rangle$ into a 9×1 complex vector in the following way:

$$f(\langle \mathbf{T} \rangle) = [T_{11}, T_{12}^{\dagger}, T_{13}^{\dagger}, T_{12}, T_{22}, T_{23}^{\dagger}, T_{13}, T_{23}, T_{33}]^{\mathsf{T}}. \tag{2.2}$$

$\Sigma = f^{-1}(\sigma)$ is a inverse function which transforms 9×1 complex vector back to 3×3 complex coherency matrix. The trace operation can now be calculated in terms of a vector dot product as $\text{trace}(\langle \mathbf{C}_m \rangle^{-1} \langle \mathbf{T} \rangle) = \langle f(\langle \mathbf{C}_m \rangle^{-1}), f(\langle \mathbf{T} \rangle) \rangle$. Here $\langle \mathbf{x}, \mathbf{y} \rangle$ is a dot product of two vectors $\mathbf{x}$ and $\mathbf{y}$. Now this operation only requires 9 complex number multiplications and 8 complex number additions. Let,

$$\begin{aligned} \mathbf{X} &= [\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_N],\ \forall n \in \{1, \ldots, N\}\ \mathbf{t}_n = f(\langle \mathbf{T}_n \rangle), \\ \mathbf{W} &= [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_M],\ \forall m \in \{1, \ldots, M\}\ \mathbf{c}_m = f((\langle \mathbf{C}_m \rangle^{-1})^{\mathsf{T}}), \\ \mathbf{b} &= [\ln(|\langle \mathbf{C}_1 \rangle|), \ln(|\langle \mathbf{C}_2 \rangle|), \ldots, \ln(|\langle \mathbf{C}_M \rangle|)]. \end{aligned} \tag{2.3}$$

Here $\mathbf{X}$ is a matrix whose i^{th} column contains the vector representation of coherency matrix of i^{th} training pixel $\langle \mathbf{T}_i \rangle$. N is the total number of training pixels. So, the size of the matrix $\mathbf{X}$ will be $9 \times N$. Similarly, $\mathbf{W}$ is a matrix whose i^{th} column contains the vector representation of mean coherency matrix of i^{th} class. M is the total number of class of the PolSAR image. So, the size of matrix $\mathbf{W}$ will be $9 \times M$. Finally, $\mathbf{b}$ is a vector whose i^{th} element is a log of the determinant of mean coherency matrix of the class i. Hence the size of the vector $\mathbf{b}$ will be $M \times 1$. Using the linearization technique shown in eq. (2.3), the Wishart distance between every PolSAR pixel $\langle \mathbf{T}_i \rangle$ and every cluster center $\langle \mathbf{C}_m \rangle$ can be calculated as one matrix

multiplication as shown below.

$$\mathbf{D} = \mathbf{W}^{\mathsf{T}}\mathbf{X} + \mathbf{b}$$
$$= \begin{bmatrix} D_{Wishart}(\langle \mathbf{T}_1 \rangle, \langle \mathbf{C}_1 \rangle) & D_{Wishart}(\langle \mathbf{T}_1 \rangle, \langle \mathbf{C}_1 \rangle) & \dots & D_{Wishart}(\langle \mathbf{T}_N \rangle, \langle \mathbf{C}_1 \rangle) \\ D_{Wishart}(\langle \mathbf{T}_1 \rangle, \langle \mathbf{C}_2 \rangle) & D_{Wishart}(\langle \mathbf{T}_2 \rangle, \langle \mathbf{C}_2 \rangle) & \dots & D_{Wishart}(\langle \mathbf{T}_N \rangle, \langle \mathbf{C}_2 \rangle) \\ \vdots & \vdots & \ddots & \vdots \\ D_{Wishart}(\langle \mathbf{T}_1 \rangle, \langle \mathbf{C}_M \rangle) & D_{Wishart}(\langle \mathbf{T}_2 \rangle, \langle \mathbf{C}_M \rangle) & \dots & D_{Wishart}(\langle \mathbf{T}_N \rangle, \langle \mathbf{C}_M \rangle) \end{bmatrix}. \tag{2.4}$$

The $\mathbf{D}(i, j)$ entry contains the Wishart distance between j^{th} cluster center and i^{th} pixel of the PolSAR image. We will now incorporate this linearization technique into the Wishart network.

Let $\Omega = \{(\langle \mathbf{T}_1 \rangle, \mathbf{y}_1), (\langle \mathbf{T}_2 \rangle, \mathbf{y}_2), \dots, (\langle \mathbf{T}_N \rangle, \mathbf{y}_N)\}$ be a set of N labeled training samples where $\mathbf{y}_i$ is the one-hot-encoding representation of the label of i^{th} training sample $\langle \mathbf{T}_i \rangle$. Using these training samples, matrices $\mathbf{X}$, $\mathbf{W}$ and $\mathbf{b}$ can be obtained using eq. (2.3). Figure 2.1 shows the architecture of Wishart network. It has three

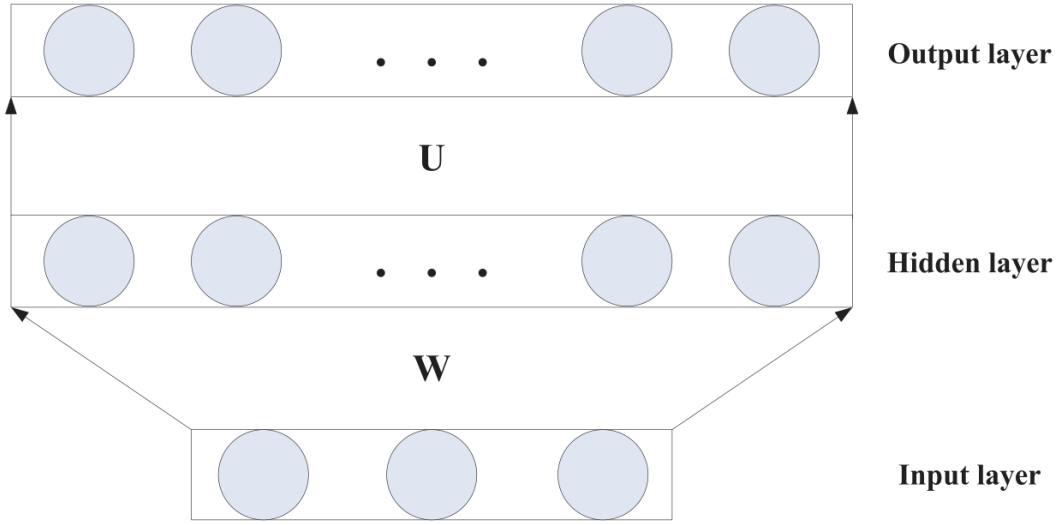


Figure 2.1: Architecture of Wishart network.

layers, weights of input-hidden layer is initialized using $\mathbf{W}$ matrix. Bias of input-hidden layer is initialized with the vector $\mathbf{b}$. We feed matrix $\mathbf{X}$ whose columns are vector representation of all training samples as an input to the network. As shown in eq. (2.4), the input-hidden layer calculates the Wishart distance of every training sample with the mean coherency matrices of all classes. Hidden-output layer then uses these distances and maps it to the correct class label. The output

of the network is given as follows:

$$\begin{aligned} \mathbf{h}_i &= sigmoid\left(\mathbf{W}^{\mathsf{T}}\mathbf{t}_i + \mathbf{b}\right), \\ \hat{\mathbf{y}}_i &= \mathbf{U}^{\mathsf{T}}\mathbf{h}_i + \mathbf{c} = \hat{\mathbf{U}}^{\mathsf{T}}\hat{\mathbf{h}}_i. \end{aligned} \tag{2.5}$$

Here, $\hat{\mathbf{U}} = [\mathbf{U}; \mathbf{c}]$ and $\hat{\mathbf{h}_i} = [\mathbf{h}_i; \mathbf{1}]$. To train the WN, MSE loss function is used. The loss function is given as follows:

$$L(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left(\|\mathbf{y}_i - \hat{\mathbf{y}}_i\|^2\right). \tag{2.6}$$

Here $\theta = \{\mathbf{W},\ \mathbf{U},\ \mathbf{b},\ \mathbf{c}\}$ is a set of parameters of WN. Since the loss function shown in eq. (2.6) is convex, the least-square estimation of $\hat{\mathbf{U}}$ can be obtained as:

$$\hat{\mathbf{U}} = \sum_{i=1}^{K}\left(\hat{\mathbf{h}_i}\hat{\mathbf{h}_i}^{\mathsf{T}}\right)^{-1}\hat{\mathbf{h}_i}\mathbf{y}_i, \tag{2.7}$$

where $\mathbf{y_i}$ is the one-hot-encoding representation of true class label of i^{th} training sample. It is understood that by increasing number of hidden units, we can improve the representation capability of a neural network. Hence it improves classification accuracy. In the case of WN, the number of hidden units depends on the number of classes. To increase the number of hidden units in a WN, weights associated with each new hidden unit have to be initialized. In a WN, weights of the first M hidden units are initialized using the inverse of class mean coherency matrix. The weights of additional hidden units are initialized using the inverse of the coherency matrix of randomly selected samples.

There are two main limitations of WN. First, WN still requires complex-valued matrix multiplication since matrices $\mathbf{X}$ and $\mathbf{W}$ are complex-valued, leading to excessive computational overhead. Second, to increase the number of hidden units, WN randomly selects pixels from the given PolSAR image, thereby ignores the heterogeneous nature of a class of the PolSAR image.

2.3 Optimized Wishart Network

We propose two significant changes in a Wishart network. First, we propose a method of representing the coherency matrix as a real-valued vector without losing the phase information. It will help reduce computational time significantly as now multiplication and addition are not of complex quantities but of real num-

bers. Therefore it improves the response time of Wishart distance calculation and hence of classification.

Next, we propose a better weight initialization method to increase the number of hidden units by considering the heterogeneous nature of a PolSAR image class. We show that it results in considerable improvement in classification accuracy. Combining the two, we show that our proposed approach is more efficient in terms of classification time and accuracy.

2.3.1 Real-valued vector representation

In section 2.2 we referred to a linear implementation [63] of Wishart distance for faster calculation, where trace($\langle \mathbf{C}_m \rangle^{-1} \langle \mathbf{T} \rangle$) is linearized by exploiting the fact that trace of any matrix is just a summation of diagonal elements of that matrix. The linearization showed that the trace could be calculated by multiplying two 9×1 complex-valued vectors, which included 9 complex numbers multiplication and 8 complex numbers addition. If we have to multiply two complex numbers a+ib and c+id, it is (ac-bd+iad+ibc) = (ac-bd + i(ad+bc)), so single complex numbers multiplication requires 4 multiplications and 2 additions of real numbers. Similarly, single complex numbers addition requires 2 real numbers addition. Hence internally multiplying two 9×1 complex-valued vectors requires 36 real number multiplications and 34 real number additions.

The first improvement in the WN is made by representing coherency matrix as a real-valued vector by exploiting Hermitian nature of a coherency matrix. We recall that diagonal elements of the coherency matrix are real-valued, and off-diagonal elements are complex conjugate to each other. So, the coherency matrix can be represented in terms of 9 unique real values, as shown in the equation below.

$$\langle \mathbf{T} \rangle = \mathbf{k}_p \mathbf{k}_p^\dagger = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{12}^\dagger & T_{22} & T_{23} \\ T_{13}^\dagger & T_{23}^\dagger & T_{33} \end{bmatrix} = \begin{bmatrix} a & b+ic & d+ie \\ b-ic & f & g+ih \\ d-ie & g-ih & j \end{bmatrix}. \tag{2.8}$$

Let $\langle \mathbf{A} \rangle$ be a coherency matrix of a pixel of a PolSAR image and $\langle \mathbf{B} \rangle$ be a coherency matrix of any one of the cluster centers of the same PolSAR image. Using the property that inverse of a Hermitian matrix is also Hermitian, it can readily be shown that if $\langle \mathbf{B} \rangle$ is a Hermitian matrix, then $\langle \mathbf{B} \rangle^{-1}$ will also be the Hermitian

matrix. Therefore, from eq. (2.8), $\langle \mathbf{A} \rangle$ and $\langle \mathbf{B} \rangle^{-1}$ have the following forms:

$$\begin{aligned} \langle \mathbf{A} \rangle &= \begin{bmatrix} a_1 & b_1 + ic_1 & d_1 + ie_1 \\ b_1 - ic_1 & f_1 & g_1 + ih_1 \\ d_1 - ie_1 & g_1 - ih_1 & j_1 \end{bmatrix} \\ \langle \mathbf{B} \rangle^{-1} &= \begin{bmatrix} a_2 & b_2 + ic_2 & d_2 + ie_2 \\ b_2 - ic_2 & f_2 & g_2 + ih_2 \\ d_2 - ie_2 & g_2 - ih_2 & j_2 \end{bmatrix} \end{aligned}. \tag{2.9}$$

Following eq. (2.1), we are only interested in the trace of matrix multiplication of two such matrices shown in eq. (2.9). It readily turns out that during matrix multiplication, imaginary parts cancel out each other, leaving with a relatively pure form of multiplication of 9 real numbers, as shown below.

$$\begin{aligned} \text{trace}\left(\langle \mathbf{B} \rangle^{-1} \langle \mathbf{A} \rangle \right) = a_1 a_2 + 2b_1 b_2 + 2c_1 c_2 + 2d_1 d_2 + 2e_1 e_2 \\ + f_1 f_2 + 2g_1 g_2 + 2h_1 h_2 + j_1 j_2. \end{aligned} \tag{2.10}$$

Recalling eq. (2.2), let σ be a function on coherency matrix which transforms 3×3 complex matrix into 9×1 real-valued vector

$$\sigma = f\Big(\langle \mathbf{T} \rangle \Big) = [a, b, c, d, e, f, g, h, j]^{\mathsf{T}}, \tag{2.11}$$

such that $\langle \mathbf{T} \rangle = f^{-1}(\sigma)$, an inverse function which transforms 9×1 real-valued vector back into 3×3 coherency matrix. Considering these two functions trace part of Wishart distance can be calculated as:

$$\begin{aligned} \text{trace}\Big(\langle \mathbf{B} \rangle^{-1} \langle \mathbf{A} \rangle \Big) = \Big\langle f\Big(\langle \mathbf{A} \rangle \Big) \odot \mathbf{p},\ f\left(\langle \mathbf{B} \rangle^{-1} \right) \Big\rangle, \\ \mathbf{p} = [1, 2, 2, 2, 2, 1, 2, 2, 1]^{\mathsf{T}}. \end{aligned} \tag{2.12}$$

Here, $\mathbf{p}$ is a constant vector arising from eq. (2.10). It has to be multiplied only ones. '$\odot$' is element-wise multiplier and $\langle \mathbf{x}, \mathbf{y} \rangle$ is a vector dot product of two 9×1 real-valued vectors $\mathbf{x}$ and $\mathbf{y}$. It is to note that in our proposed method $f(\langle \mathbf{A} \rangle)$ and $f(\langle \mathbf{B} \rangle^{-1})$ are both 9×1 real-valued vectors. Therefore, trace operation can effectively be calculated using just 9 real numbers multiplications and 8 real numbers additions. We call it Optimized Wishart Network (OWN). Here, the term *optimized* is used in the sense that it improves on classification time as well as overall accuracy of WN. Using eqs. (2.11) and (2.12), linear implementation of Wishart

distance is defined as follows:

$$\begin{aligned}
\mathbf{X} &= [\mathbf{t}_1 \odot \mathbf{p}, \mathbf{t}_2 \odot \mathbf{p}, \ldots, \mathbf{t}_N \odot \mathbf{p}]\,,\ \forall n \in \{1, \ldots, N\}\ \mathbf{t}_n = f(\langle \mathbf{T}_n \rangle), \\
\mathbf{W} &= [\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_M]\,,\ \forall m \in \{1, \ldots, M\}\ \mathbf{c}_m = f(\langle \mathbf{C}_m \rangle^{-1}), \\
\mathbf{b} &= [\ln(|\langle \mathbf{C}_1 \rangle|), \ln(|\langle \mathbf{C}_2 \rangle|), \ldots, \ln(|\langle \mathbf{C}_M \rangle|)]\,, \\
\mathbf{D} &= \mathbf{W}^{\mathsf{T}}\mathbf{X} + \mathbf{b}.
\end{aligned} \tag{2.13}$$

This process consumes at most one-fourth of the time, compared to the method of Jiao and Liu [63]. It should be noted that the use of real-valued feature vector representation does not result in loss of information because we are not discarding the complex counterpart of the coherency matrix. We are encoding complex coherency matrix in the form of the real-valued vector in such a way that we can still calculate the exact Wishart distance.

2.3.2 Expanding Network

Then we attempt on second limitation of a WN, i.e., it considers a class completely homogeneous. The second improvement in the performance of the network is attempted by increasing the number of hidden units in the proposed network. The existing WN adopted an approach where they initialized weights associated with additional hidden units with randomly selected pixels (coherency matrix) from any class. There are two issues with this approach. First is the assumption that pixels within the same class are homogeneous so that they can be modeled as a single cluster. It is understood that pixels within the same class are not necessarily homogeneous [43]. Keeping this in mind, Gao *et al*. [43] proposed to model each class as a mixture of Wishart probability distribution functions rather than a single Wishart probability distribution function. So modeling heterogeneity within a class as a single cluster may not be an accurate representation. Second, additional hidden units are only helpful in improving accuracy if they provide additional information to the already available information. Let's say we have selected a random pixel 't' from a class 'm'. If we use a random approach, then 't' may be very close to the cluster center of class 'm'. In such a case, we gain very little information from 't' because Wishart distances of any unknown sample with 't' and cluster center of 'm' are almost similar.

To solve these two issues, we adopted a different method to increase the number of hidden units. Let's assume we have M classes, we represent each class using P hidden units to get PM total hidden units. To do so, we apply the k-means algorithm with $k = P$ on training samples of each class. In other words,

we propose to find P cluster centers using the k-means algorithm instead of just one from each class. Then we use these cluster centers to initialize weights associated with all hidden units. This method assumes that pixels within the same class are in general heterogeneous, so it solves the first issue. If pixels from the same class are heterogeneous, then cluster centers found by k-means are going to be sufficiently different from each other, which solves the second issue. We are using the k-means algorithm to find approximate cluster centers with an objective of getting results fast.

2.3.3 Training Optimized Wishart Network

Real-valued vector representation developed in the section 2.3.1 can now be used for the classification problem of a PolSAR image. We have used a fully connected single hidden layer network. The input layer of the proposed network takes 9×1 real-valued vector representation of the coherency matrix as an input; input-hidden layer weights are initialized using the inverse coherency matrix of cluster centers obtained by k-means algorithm. So, during forward pass, the input-hidden layer calculates Wishart distance of training PolSAR pixel with different class means, while hidden to output layer tries to map those distances to its correct label. Once the forward pass is complete, training error is calculated using MSE loss function described in eq. (2.6). During backward pass, parameters of OWN are updated using the back-propagation algorithm. Algorithm-1 shows the training steps of the proposed network. Once the training of the OWN is complete we can measure its generalization capability by testing its performance on unseen test data. The generalization capability of any learning algorithm indicates how well the algorithm performs on the data that it has not seen previously. For all test samples we do the forward pass on OWN and obtain their prediction. Having the prediction obtained we assign label to each test samples for which its prediction value is largest. Algorithm 2 shows the steps of assigning labels to test samples.

2.4 Experiments and results

To evaluate the performance of OWN, we have used two datasets. The first dataset is the L band Flevoland dataset [11] acquired by AIRSAR aircraft. It has 15 classes, namely, Stembeans, Peas, Forest, Lucerne, Wheat, Beat, Potatoes, Bare soil, Grasses, Rapeseed, Barley, Wheat 2, Wheat 3, Water and Buildings. Its size is

Algorithm 1 Training Algorithm of OWN

M = Number of classes, P= number of cluster in each class, K = size of training set, *iter* = Number of iterations to perform, λ = Learning rate

$Y = \left[\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_K\right]$, where $\mathbf{y}_i$ is a $M \times 1$ vector whose non-zero entry indicates true class label of i^{th} training sample.

$X = \left[\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_k\right] \odot \mathbf{p}$ where $\mathbf{t}_i = f(\langle \mathbf{T}_i \rangle)$, $\langle \mathbf{T}_i \rangle$ is coherency matrix of i^{th} training sample.

for $m = 1, 2, ..., M$ **do**

Apply k-means algorithm on training sample of class m to find P cluster mean $\mathbf{w}_1^m, \mathbf{w}_2^m, ..., \mathbf{w}_P^m$.

end for

$\mathbf{W} = \left[f((\mathbf{w}_1^1)^{-1}), f((\mathbf{w}_2^1)^{-1}), \ldots, f((\mathbf{w}_P^M)^{-1})\right]$

$\mathbf{b} = \left[|\mathbf{w}_1^1|, |\mathbf{w}_2^1|, \ldots, |\mathbf{w}_P^1|, \ldots, |\mathbf{w}_1^M|, \ldots, |\mathbf{w}_P^M|\right]^{\mathsf{T}}$

for $i = 1, 2, \ldots, iter$ **do**

$\mathbf{H} = sigmoid\left(\mathbf{W^T X} + \mathbf{b}\right)$

$\mathbf{H} = \left[\mathbf{H}; \mathbf{1}\right]$

$\mathbf{U} = \left(\mathbf{H H^T}\right)^{-1}\left(\mathbf{H Y^T}\right)$

$\mathbf{W} = \mathbf{W} - \lambda \frac{\delta E}{\delta \mathbf{W}}$

for $i = 1, 2, \ldots, P * M$ **do**

$\mathbf{b}[i] = \left|[f^{-1}(\mathbf{W}[:, i])]^{-1}\right|$

end for

end for

return W, **U**, **b**

Algorithm 2 Predict Algorithm

L = Number of test samples, obtain **W**, **U** and **b** from Algorithm 1.

$\mathbf{X} = \left[\mathbf{t}_1, \mathbf{t}_2, ..., \mathbf{t}_L\right] \odot \mathbf{p}$ Where $\mathbf{t}_i$ is i^{th} test sample.

$\mathbf{X} = \left[\mathbf{X}; \mathbf{1}\right]$

$\mathbf{H} = sigmoid\left(\mathbf{W^T X}\right)$

$\mathbf{H} = \left[\mathbf{H}; \mathbf{1}\right]$

$\mathbf{prediction} = \left(\mathbf{U}^{\mathsf{T}}\mathbf{H}\right)$

return arg max$\left(\mathbf{prediction}\right)$

750×1024. Figure 2.2 shows Pauli decomposition of the Flevoland dataset along with the ground truth image. The second dataset is the C band San Francisco dataset [93] obtained by RADARSAT-2 satellite. It contains 5 classes: Water, Vegetation, Developed Area, High-Density Urban and Low-Density Urban.

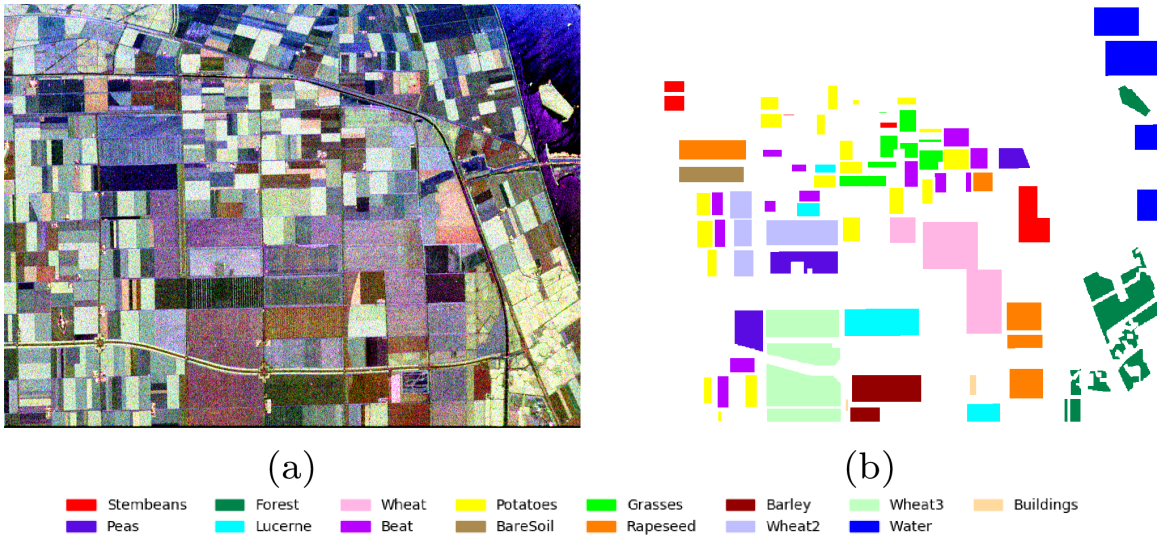


Figure 2.2: (a) Pauli Decomposition of Flevoland dataset containing 15 classes, (b) Ground Truth.

2.4.1 Comparison with Wishart network

Flevoland dataset

The Flevoland dataset contains 768000 total pixels. Of them 157296 are labeled pixels which belong to 15 classes. For fair comparison, we have followed a similar evaluation process as used by WN. From the total labeled pixels, we randomly select 5% pixels from each class as training data and the remaining 95% pixels as test data. Number of iteration was set to 100 and learning rate λ was set to 0.2. The size of the hidden layer was set as $2M$, $3M$ and $4M$. Here, M is total number of classes. For example, in case of $4M$ size, OWN selects four cluster centers from each class using k-means algorithm, resulting in a total of $4M$ cluster centers. On the other hand WN selects one cluster center from each class, resulting in a M cluster centers. Remaining $3M$ cluster centers are selected randomly from the image. Classification results of WN and OWN are shown in Table 2.1. As we can see, OWN achieves 3.31%, 2.23% and 2.06% better accuracy than WN in case of 2M, 3M and 4M hidden units respectively. It shows the superiority of the proposed weight initialization method. In the same Table 2.1, the second row of each net-

Table 2.1: Classification Accuracy and Time Comparison

Method	$2M$	$3M$	$4M$
WN [63]	**87.02**	**89.55**	**90.18**
	9.37 sec	13.53 sec	17.93 sec
OWN (Proposed)	**90.33**	**91.78**	**92.24**
	4.23 sec	6.08 sec	8.00 sec

work is showing time comparison of training plus testing for each network. One can see that OWN is also more than 2 times faster than WN. This is because of the proposed real-valued feature representation.

Table 2.2 shows class-wise accuracy comparison of WC, WN and OWN with $4M$ hidden units and Figures 2.3 show corresponding classification maps. As we can see, in comparison with WN, OWN was able to achieve better accuracy in 12 of the 15 classes. Steambeans is most improved class where the accuracy improved 11.19% as compared with WN. The other notable improvement was ob-

Table 2.2: Class-wise Classification Accuracy Comparison

Class	Samples	WC	WN (4M)	OWN (4M)
Stembeans	6103	95.18	85.19	**96.38**
Peas	9111	**96.00**	95.55	95.93
Forest	14944	90.66	89.46	**90.77**
Lucerne	9477	92.60	96.13	**96.34**
Wheat	17283	85.28	**94.48**	93.00
Beat	10050	94.64	**96.06**	95.80
Potatoes	15292	86.69	88.53	**89.62**
Bare soil	3078	98.70	98.33	**98.72**
Grasses	6269	75.29	78.17	**80.52**
Rapeseed	12690	74.34	77.23	**83.34**
Barley	7156	94.13	97.23	**98.20**
Wheat 2	10591	**84.34**	72.08	81.13
Wheat 3	21300	89.86	94.55	**95.68**
Water	13476	46.94	98.39	**98.97**
Buildings	476	82.56	**84.55**	77.88
OA	-	84.64	90.18	**92.24**
AA	-	85.81	89.72	**91.48**

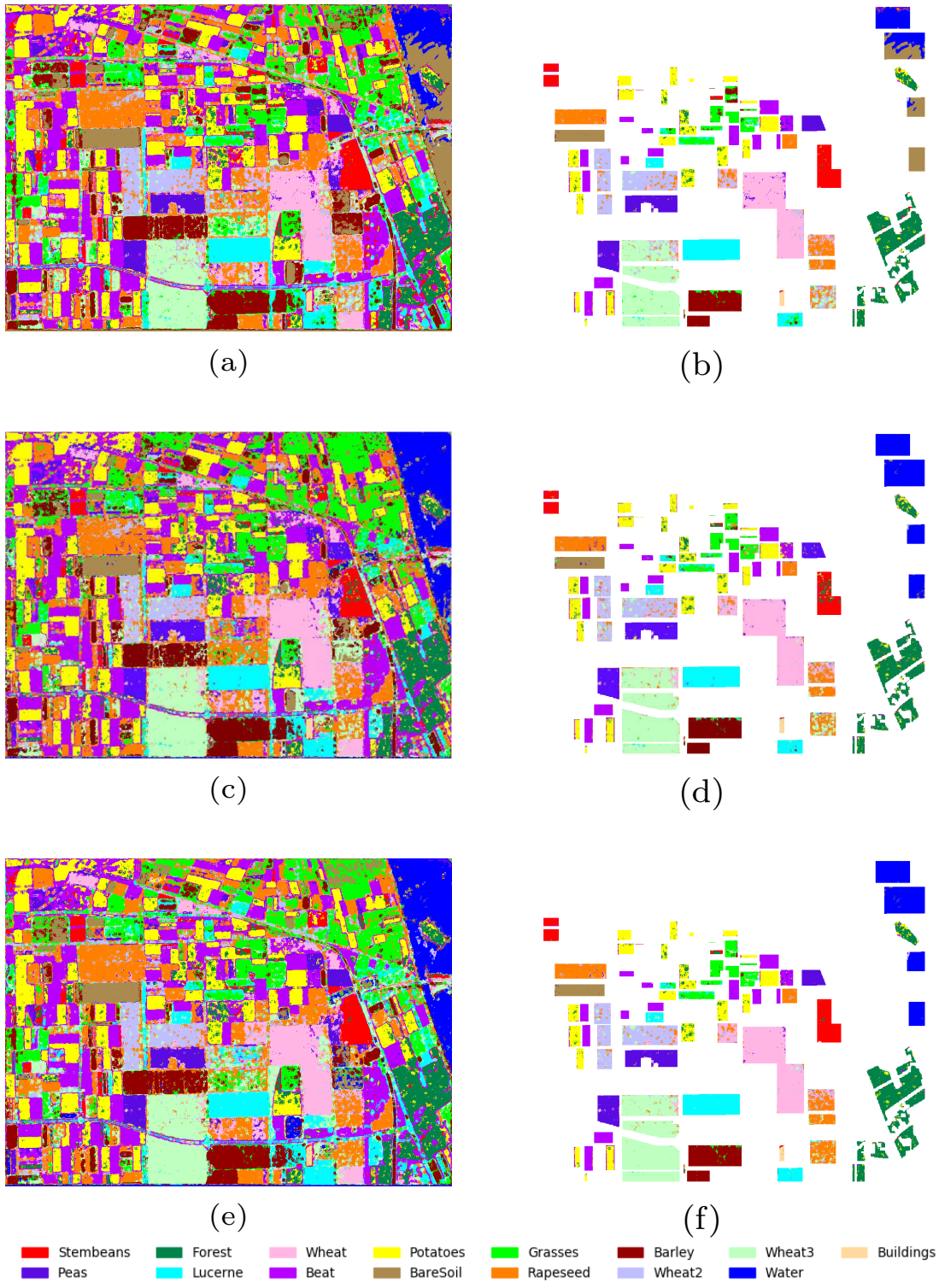


Figure 2.3: Classification maps along with its mask (overlapping with ground truth) obtained using (a)-(b) Wishart classifier [78], (c)-(d) Wishart network [63] and (e)-(f) proposed optimized Wishart network.

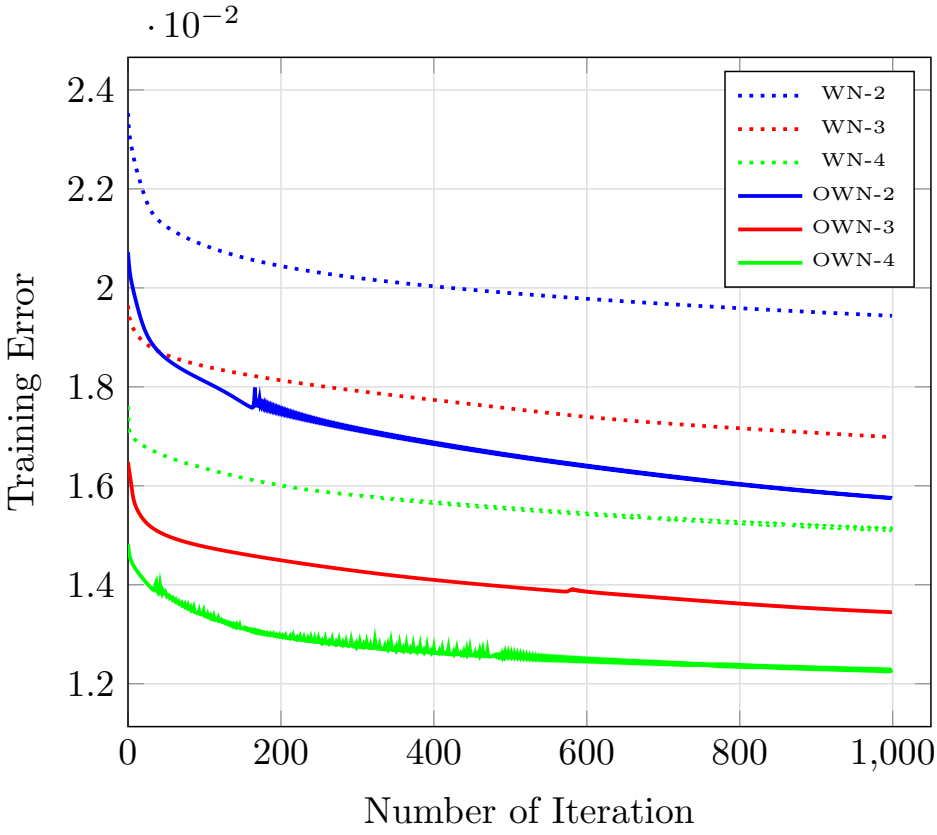


Figure 2.4: Comparison of training error between OWN and WN with different number of hidden units. Here, OWN-2 shows training error obtained by OWN with 2*M hidden units. Same is followed by other 5 methods.

served in classes of Wheat 2, Rapeseed and Grasses where the accuracy improved by 9.05%, 6.11% and 2.35% respectively. We observe decline in accuracy in case of the class Buildings. The accuracy was 6.67% lower as compared with WN. It maybe attributed to very small number of labelled samples (476 only) in that class. Therefore we had over 20 pixels for training.

Next, we compare the training error of WN and OWN. Figure 2.4 shows the training error of WN and OWN with different numbers of hidden units plotted on the x-axis. On the y-axis, we show the number of iterations. For example, OWN-4 means training error of OWN with $4M$ hidden units. As we can see, for the same number of hidden units, OWN starts with smaller training error in comparison to WN in all three cases. It shows the superiority of the proposed weight initialization method. This leads to a better convergence and improved classification accuracy of OWN.

San Francisco dataset

Similarly, we perform experiments on San Francisco dataset. Figure 2.5 shows Pauli decomposition of San Francisco dataset along with the ground truth image. It contains 2484000 total pixels out of which 1804087 are labeled. Due to large number of labeled pixels (almost 75% of total pixels) we used only 1% of them as training data and remaining as testing data. Table 2.3 shows class-wise classification accuracy obtained by WC, WN (4M) and OWN (4M) on San Francisco dataset. Figure 2.6 shows corresponding classification maps. Similar to Flevoland dataset, OWN achieves 3.13% better OA compared to WN and 5.49 % better OA than WC. The maximum improvement was observed for Low-Density Urban class with 11.68% better accuracy than WN. For Developed and High-Density Urban classes, accuracy was improved by 9.24% and 0.80% respectively. For Vegetation class, accuracy of OWN is 2.02% better than WN but 5.17% lower than Wishart classifier. In case of Wishart classifier, there is a confusion between the Low-Density Urban and High-Density Urban due to similar nature of scattering of both classes and the assumption of homogeneity in each class. Similar confusion is observed in case of WN also. Due to the consideration of heterogeneity of classes, we find less

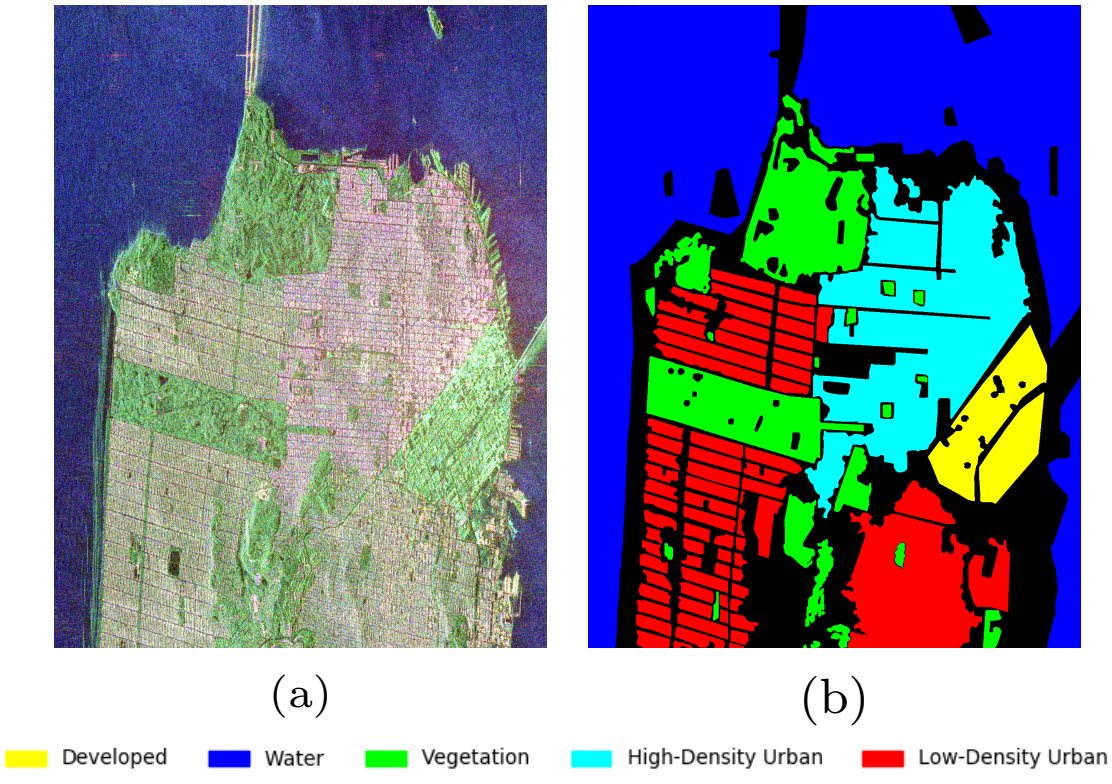


Figure 2.5: (a) Pauli Decomposition of San Francisco dataset containing 5 classes, (b) Ground Truth.

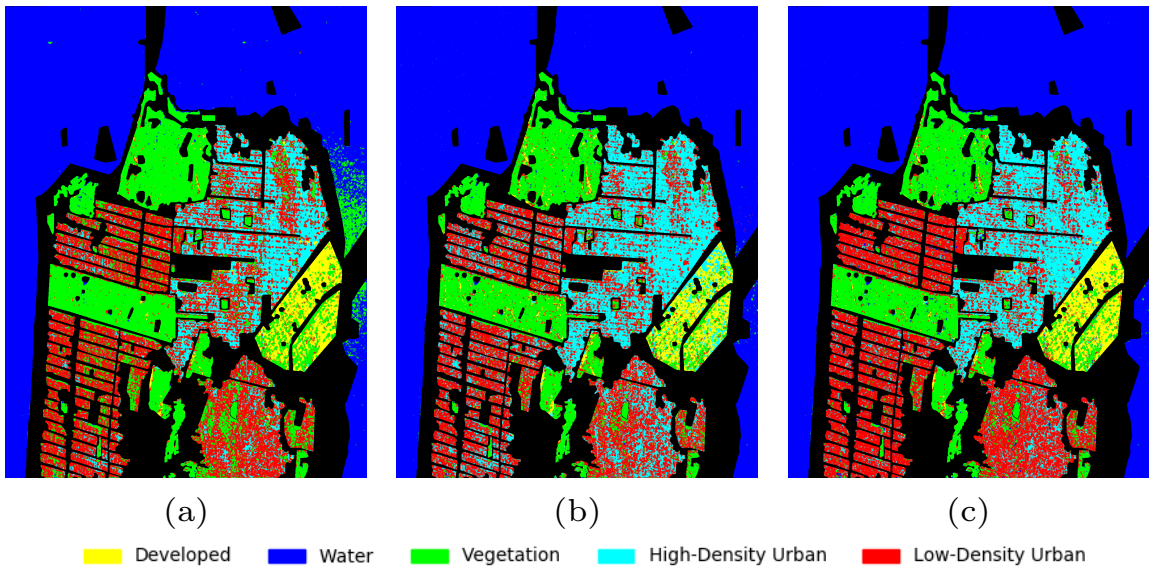


Figure 2.6: Classification map obtained using (a) Wishart classifier [78], (b) Wishart network [63] and (c) Proposed optimized Wishart network.

Table 2.3: Class-wise accuracies of San Francisco dataset

Class	Samples	WC	WN (4M)	OWN (4M)
Developed	80616	56.99	51.97	**61.21**
Water	852078	97.18	99.83	**99.95**
Vegetation	237237	**92.92**	85.73	87.75
High-Density Urban	282975	51.30	68.98	**69.78**
Low-Density Urban	351181	69.00	66.45	**78.13**
OA	-	82.14	84.50	**87.63**
AA	-	73.48	74.59	**79.36**

confusion between Low-Density and High-Density classes in case of OWN.

2.4.2 Comparison with other Wishart distribution based methods

In literature, we find multiple ways to use Wishart distribution in the learning process of the neural network. In this section, we compare our proposed method with other neural network based methods such as Wishart Autoencoder (WAE) [124] and Wishart Deep Belief Network (W-DBN) [86] that use Wishart distribution of the PolSAR pixel. For this experiment, the hidden layer's size for OWN was set to 15M.

As discussed in section 1.4.4 of Chapter 1, autoencoder based networks are

Table 2.4: Class-wise accuracy comparison of Flevoland dataset

Class	**OWN**	**WAE [124]**	**W-DBN [86]**
Stembeans	**97.48**	94.91	96.60
Peas	96.76	95.95	**96.80**
Forest	**91.93**	91.01	91.41
Lucerne	**97.22**	94.59	96.11
Wheat	**94.14**	91.19	92.32
Beat	**97.04**	95.21	95.81
Potatoes	**89.45**	89.33	88.80
Bare soil	**99.06**	92.48	96.99
Grasses	89.47	84.88	**90.93**
Rapeseed	83.60	84.37	**86.71**
Barley	96.65	95.01	**98.06**
Wheat 2	84.58	80.46	**88.17**
Wheat 3	**96.57**	94.99	93.77
Water	**99.81**	96.50	99.60
Buildings	76.89	83.79	**85.26**
OA	**93.42**	91.41	93.08
AA	92.71	90.97	**93.15**

unsupervised methods of extracting useful features by mapping inputs to lower dimension space and minimizing the reconstruction error. The WAE [124] is a pixel based classifier which uses elements of the coherency matrix as an input to a single hidden layer network. The reconstruction loss function of WAE is shown in eq. (2.14). It uses Wishart distance to measure the similarity between the input coherency matrix and the reconstructed coherency matrix.

$$\begin{aligned} J(X,Y) = & \frac{1}{2N}\sum_{i=1}^{N} D_{Wishart}\left(f^{-1}(\mathbf{x}_i), f^{-1}(\mathbf{y}_i)\right) \\ & + \frac{\lambda}{2}\sum_{j=1}^{2}\sum_{p=1}^{n}\sum_{q=1}^{m}\left\|\mathbf{W}_{pq}^{(j)}\right\|_F^2 \\ & + \frac{\beta}{2}\sum_{k=1}^{m} KL(\rho||\hat{\rho}_j). \end{aligned} \tag{2.14}$$

Here $D_{Wishart}(f^{-1}(\mathbf{x}_i), f^{-1}(\mathbf{y}_i))$ is a Wishart distance between input coherency matrix $f^{-1}(\mathbf{x}_i)$ and reconstructed coherency matrix $f^{-1}(\mathbf{y}_i)$. ρ is the sparsity

Table 2.5: Class-wise accuracy comparison of San Francisco dataset

Class	Samples	**OWN**	**WAE [124]**	**W-DBN [86]**
Developed	80616	62.68	**65.99**	62.88
Water	852078	**99.92**	99.83	**99.92**
Vegetation	237237	**88.41**	83.57	77.59
High-Density Urban	282975	**70.38**	65.44	51.01
Low-Density Urban	351181	80.93	80.27	**89.86**
OA	-	**88.57**	87.44	86.11
AA	-	**80.46**	79.02	76.25

parameter and $\hat{\rho}_j$ is the average activation of the j^{th} hidden unit. $KL(\rho||\hat{\rho}_j)$ is a Kullback-Leibler (KL) divergence. It measures the difference between two probability distributions but it is not a distance metric, because it is not symmetric. KL divergence is defined as:

$$KL(\rho||\hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1-\rho)\log \frac{(1-\rho)}{(1-\hat{\rho}_j)}. \tag{2.15}$$

Once the features are extracted, classification is performed using softmax classifier.

Deep Belief Network (DBN) [86] is a type of multilayer neural network which learns to model the probability distribution of the data in an unsupervised manner. W-DBN is the extension of DBN designed for PolSAR data whose visible layer models the Wishart distribution. Once the W-DBN is trained, classification is performed using softmax classifier.

Table 2.4 compares classification accuracy of the proposed OWN to WAE [124] and W-DBN [86] on Flevoland dataset. As we can see OWN achieves better accuracy in 9 out of 15 classes. Similar experiment was also conducted on San Francisco dataset and shown in Table 2.5. Here, OWN was able to achieve better accuracy in three out of five classes. From experiment on both dataset, it is evident that the proposed shallow neural network with small number of weights achieves comparable results to deep neural networks with large number of weights.

2.5 Conclusion

Classification of PolSAR images has been a major area of research in the field of remote sensing. Wishart Network (WN) is the way of incorporating the statisti-

cal property of the PolSAR image into the learning process of a neural network. During the training of WN, forward pass leads to the calculation of Wishart distance and then mapping of these distance values to the predicted class label is done. Inherently, WN assumes that each class of a PolSAR data is homogeneous. Therefore it models each class as a single Wishart distribution. It is shown [43] that the homogeneity assumption is not always valid. So, considering each class as a single Wishart distribution may lead to reduced performance of the classifier.

We proposed an Optimized Wishart Network (OWN), which incorporates two novel steps to improve the performance of the WN. First, we proposed a novel linearization method of calculating Wishart distance, which uses only real numbers. It leads to a much faster training and classification time. We then also proposed a better technique of initializing weights of hidden units by considering heterogeneous nature of PolSAR pixels of same class. It led to improved classification accuracy. To test the performance of OWN, we performed experiments on the Flevoland dataset containing 15 classes and San Francisco dataset with 5 classes. We showed that proposed OWN was faster by at least 2 times and achieved better classification accuracy compared to the WN.

In OWN, we used the k-means algorithm to find the cluster centers, which is a very fast algorithm. But, its disadvantage is that cluster centers are not guaranteed to be optimal, and Euclidean distance used in k-means is not an appropriate approach for PolSAR data. In Chapter 4, we present a fast global k-means based algorithm that uses revised Wishart distance to get optimal cluster centers. The above is done for classification of single frequency or single band PolSAR image. In the next chapter, we will extend the proposed optimized Wishart network for the classification of a multifrequency dataset. It utilizes the fact that the pixels of different frequencies are statistically independent.

CHAPTER 3
Extended OWN (e-OWN) for Multifrequency PolSAR Data

3.1 Introduction

In the previous Chapter, we presented the Optimized Wishart Network (OWN) for the classification of single frequency PolSAR data. OWN was able to achieve better classification accuracy not only than WN but also in comparison with deep learning based architectures such as WAE and W-DBN. In this chapter, we explore the potential of OWN for multifrequency PolSAR data.

Some PolSAR systems, such as AIRSAR, EMISAR, F-SAR, etc., are capable of capturing PolSAR images in multifrequency bands simultaneously. The advantage of this multifrequency bands data is that it results in a multitude of information in comparison to the single frequency PolSAR data. Since the penetration capability of each frequency band is different, each band provides a different set of information of the same terrain target. Due to this, combining different bands information can improve classification accuracy [34, 36, 64]. In chapter 2 we saw how a novel linear implementation of the Wishart distance just by using the real numbers and by using k-means algorithm to take care of the heterogeneity in the pixels distribution in the image improved the classification efficiency for a single frequency PolSAR image. In this Chapter we would explore if the same could be *extended* for multifrequency PolSAR images too. We call it the "extended Optimized Wishart Network (e-OWN)".

The proposed network has multiple input branches, one for each frequency band. Each input branch has its respective weight matrix. It is initialized by a real-valued vector representation of the inverse of cluster center matrices. Then we stack the output of each input branch which allows us to compare the features of each pixel at different frequency bands. This stacked feature vector is then used to perform classification.

This Chapter is organized as follows. In section 3.2, we first present the linear implementation of Wishart distance for multifrequency PolSAR data thereby we introduce the formulation of the proposed e-OWN. In section 3.3, we discuss experiments performed on the Flevoland and Landes dataset for C, P, and L bands. Then we compare our results with the ensemble of OWN trained on different frequency bands and with stacked autoencoder based deep learning architecture [22] to evaluate the classification efficiency and accuracy.

3.2 Extended Optimized Wishart Network (e-OWN)

The basis of OWN was a Wishart classifier that used a Wishart distance measure to compare two coherency matrices. We know that the scattered signal collected by the PolSAR satellite's antenna are coming from different parts of the same terrain, such as from the leaf canopy, from the trunk of the tree and from the soil (in case of an agriculture land). Due to different penetration depths of different frequency bands, the data from each frequency band is assumed to be statistically independent [78]. Using this fact, Wishart classifier can be extended to accommodate the classification of multifrequency PolSAR data by expanding the dimension of cluster center $\langle \mathbf{C_m} \rangle$ and PolSAR pixel $\langle \mathbf{T} \rangle$. Assuming that the data of different frequency bands are independent, the joint probability density function of a pixel over all frequency bands can be given as follows [78]:

$$p\Big(\langle \mathbf{T} \rangle | \langle \mathbf{C}_m \rangle\Big) = \prod_q \frac{n^{nd} |\langle \mathbf{T} \rangle_{(q)}|^{(n-d)}}{\Gamma_p(n) |\langle \mathbf{C}_m \rangle_{(q)}|^n} \exp\Big(- \text{trace}\Big(\langle \mathbf{C}_m \rangle_{(q)}^{-1} \langle \mathbf{T} \rangle_{(q)}\Big)\Big)$$

where, (3.1)

$$\Gamma_p(n) = \pi^{\frac{d(d-1)}{2}} \prod_{j=1}^{d} \Gamma\Big(n - j + 1\Big).$$

Here $\langle \mathbf{C}_m \rangle_{(q)}$ is the mean coherency matrix of the m^{th} class of the q-band PolSAR image. $\langle \mathbf{T} \rangle_{(q)}$ is a single pixel's coherency matrix from the q-band PolSAR image. From the joint probability distribution function, the Wishart distance for multifrequency data can be obtained by using maximum likelihood estimator in the following way:

$$D_{Wishart}\Big(\langle \mathbf{T} \rangle | \langle \mathbf{C}_m \rangle\Big) = \sum_q \ln |\langle \mathbf{C}_m \rangle_{(q)}| + \sum_q \text{trace}\Big(\langle \mathbf{C}_m \rangle_{(q)}^{-1} \langle \mathbf{T} \rangle_{(q)}\Big). \tag{3.2}$$

Using eq. (3.2) the linearization model of the Wishart classifier developed for single frequency PolSAR data can be extended for multifrequency data. Let $\langle \mathbf{T} \rangle^{i}_{(q)}$ be a coherency matrix of the i^{th} pixel of the q-band PolSAR image where $i = 1, 2, ..., N$ and $q \in \{q_1, q_2, \ldots q_n\}$. N is the total number of pixels in an image and $q_1, q_2, \ldots, q_n$ are the microwave frequency bands.

Let $\mathbf{t}^{i}_{q}$ be a 9×1 real-valued vector representation of coherency matrix of i^{th} pixel of q-band image such that $\mathbf{t}^{i}_{q} = f(\langle \mathbf{T} \rangle^{i}_{(q)})$. Here $\mathbf{t}^{i}_{q}$ is the extension of $\mathbf{t}_i$ vector discussed in Chapter 2 for the multifrequency data. Let, $\mathbf{c}^{m}_{q} = f(\langle \mathbf{C}_m \rangle_{(q)})$. Referring to eq. (2.13) of Chapter 2, in order to define a linearization model of Wishart classifier for multifrequency PolSAR data, the mathematical steps are shown in eq. (3.3). For each frequency band q, we first construct $\mathbf{X}_q$ matrix. Its columns contain real-valued vector representation of coherency matrices of training samples multiplied by constant vector $\mathbf{p}$. Similarly, we construct $\mathbf{W}_q$ matrix for each frequency band q. Each column of matrix $\mathbf{W}_q$ contains a real-valued vector representation of inverse of mean coherency matrix of some classes. Finally, vector $\mathbf{b}_q$ is constructed for each frequency band q. Its elements contain determinants of mean coherency matrices of all classes.

$$
\begin{aligned}
\mathbf{X}_q &= \left[\mathbf{t}^{1}_{q} \odot \mathbf{p}, \mathbf{t}^{2}_{q} \odot \mathbf{p}, \ldots, \mathbf{t}^{N}_{q} \odot \mathbf{p} \right], \forall q \in \{q_1, \ldots q_n\}, \\
\mathbf{W}_q &= \left[\mathbf{c}^{1}_{q}, \mathbf{c}^{2}_{q}, \ldots, \mathbf{c}^{M}_{q} \right], \ \forall q \in \{q_1, \ldots q_n\}, \\
\mathbf{b}_q &= \left[\ln(|\langle \mathbf{C}_1 \rangle_{(q)}|), \ln(|\langle \mathbf{C}_2 \rangle_{(q)}|), \ldots, \ln(|\langle \mathbf{C}_M \rangle_{(q)}|) \right], \ \forall q \in \{q_1, \ldots q_n\}.
\end{aligned}
\tag{3.3}
$$

Here $\odot$ is an element-wise multiplication operation. Using eq. (3.3), the Wishart distance between every PolSAR pixel $\langle \mathbf{T} \rangle^{i}_{(q)}$ of all bands in q and every cluster center $\langle \mathbf{C}_m \rangle_{(q)}$ of all bands in q can be calculated as n matrix multiplication as

shown below, where n is the total number of bands:

$$
\begin{aligned}
\mathbf{D} &= \sum_{q} \mathbf{W}_q^{\top} \mathbf{X}_q + \mathbf{b}_q \\
&= \begin{bmatrix} D_{Wishart}(\langle \mathbf{T}_1 \rangle_{(q_1)}, \langle \mathbf{C}_1 \rangle_{(q_1)}) & \dots & D_{Wishart}(\langle \mathbf{T}_N \rangle_{(q_1)}, \langle \mathbf{C}_1 \rangle_{(q_1)}) \\ D_{Wishart}(\langle \mathbf{T}_1 \rangle_{(q_1)}, \langle \mathbf{C}_2 \rangle_{(q_1)}) & \dots & D_{Wishart}(\langle \mathbf{T}_\mathbf{N} \rangle_{(q_1)}, \langle \mathbf{C}_2 \rangle_{(q_1)}) \\ \vdots & \ddots & \vdots \\ D_{Wishart}(\langle \mathbf{T}_1 \rangle_{(q_1)}, \langle \mathbf{C}_M \rangle_{(q_1)}) & \dots & D_{Wishart}(\langle \mathbf{T}_N \rangle_{(q_1)}, \langle \mathbf{C}_M \rangle_{(q_1)}) \end{bmatrix} \\
&+ \dots \\
&+ \begin{bmatrix} D_{Wishart}(\langle \mathbf{T}_1 \rangle_{(q_n)}, \langle \mathbf{C}_1 \rangle_{(q_n)}) & \dots & D_{Wishart}(\langle \mathbf{T}_N \rangle_{(q_n)}, \langle \mathbf{C}_1 \rangle_{(q_n)}) \\ D_{Wishart}(\langle \mathbf{T}_1 \rangle_{(q_n)}, \langle \mathbf{C}_2 \rangle_{(q_n)}) & \dots & D_{Wishart}(\langle \mathbf{T}_\mathbf{N} \rangle_{(q_n)}, \langle \mathbf{C}_2 \rangle_{(q_n)}) \\ \vdots & \ddots & \vdots \\ D_{Wishart}(\langle \mathbf{T}_1 \rangle_{(q_n)}, \langle \mathbf{C}_M \rangle_{(q_n)}) & \dots & D_{Wishart}(\langle \mathbf{T}_N \rangle_{(q_n)}, \langle \mathbf{C}_M \rangle_{(q_n)}) \end{bmatrix}.
\end{aligned} \tag{3.4}
$$

3.2.1 Network Architecture of e-OWN

Figure 3.1 shows proposed network architecture. It has three layers: the input layer, the hidden layer, and the output layer. The hidden layer and the output layer are fully connected, while the input layer and the hidden layer are partially connected. As shown in Figure 1.4 of Chapter 1, in a fully connected layer, each neuron of the current layer is connected with the next layer. It means that there is a weight associated with each neuron of the current layer to each neuron of the next layer. In case of e-OWN, the weights of input-hidden layer represent cluster centers of all classes of all frequency bands. Since different frequency bands are statistically independent [78], the Wishart distance between a pixel from band q_1 to a cluster center of band q_2 does not provide any advantage in terms of classification accuracy. Hence, we chose partially connected input-hidden layer where neurons related to input of some band q_1 is only connected with neurons of hidden layers that are related with cluster centers of band q_1. Therefore, it reduces number of weights and in turn reduces computational overhead without compromising on classification accuracy.

3.2.2 Training of e-OWN

For training and testing of the proposed e-OWN, we divide our dataset into two parts, namely, training set and test set. Let $I_m (m = 1, 2, ..., M)$ be the set of randomly selected indices of PolSAR image pixels from class m, then $I = I_1 \cup I_2 \cup ... \cup I_M$ will be the set of randomly selected indices of all training pixels. Let's say

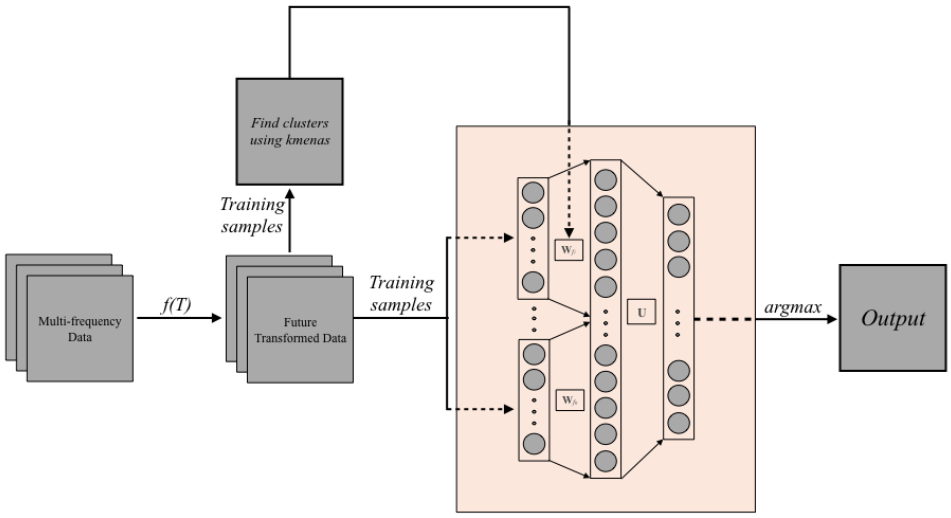


Figure 3.1: Network architecture of proposed e-OWN for multifrequency data.

we have 3 bands, L, P and C in our dataset. To train the proposed network we have used Mean Squared Error (MSE). The loss function of the proposed network is as follows:

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{y}_i - \hat{\mathbf{y}}_i\right\|^2 \tag{3.5}$$

Here $\mathbf{y}_i$ is one-hot-encoding of true label of i^{th} training sample and $\hat{\mathbf{y}}_i$ is the prediction of the network for i^{th} training sample. θ is a set of parameters of the proposed network. During forward pass, e-OWN will calculate Wishart distance between all training samples of q-band with all cluster centers of q-band found by k-means algorithm. We then apply non-linearity activation function *sigmoid* on these distances. The outputs of the non-linearity function are then stacked vertically as shown in eq. (3.6).

$$\begin{cases} \mathbf{H}_L = sigmoid\left(\mathbf{W}_L^T\mathbf{X}_L + \mathbf{b}_L\right) \\ \mathbf{H}_P = sigmoid\left(\mathbf{W}_P^T\mathbf{X}_P + \mathbf{b}_P\right) \\ \mathbf{H}_C = sigmoid\left(\mathbf{W}_C^T\mathbf{X}_C + \mathbf{b}_C\right) \\ \mathbf{H} = \begin{bmatrix} \mathbf{H}_L \\ \mathbf{H}_P \\ \mathbf{H}_C \\ \mathbf{1} \end{bmatrix} \end{cases} \tag{3.6}$$

During the backward pass, we update the weights of the network. Given the weights of the input-hidden layer, the loss function of the proposed network is convex. So the optimal solution of matrix $\mathbf{U}$ can be obtained by the least square

estimation. After that, gradient at the output and hidden layers can be computed as under.

$$\begin{cases} \begin{bmatrix} \mathbf{U} \\ \mathbf{c} \end{bmatrix} = \left(\mathbf{Y}\mathbf{H}^{\mathbf{T}}\right)\left(\mathbf{H}\mathbf{H}^{\mathbf{T}}\right)^{-\mathbf{1}} \\ \delta_{out} = \mathbf{U}\begin{bmatrix} \mathbf{H}_L \\ \mathbf{H}_P \\ \mathbf{H}_C \end{bmatrix} + \mathbf{c} - \mathbf{Y} \\ \delta_{mid} = \begin{bmatrix} \mathbf{H}_L \\ \mathbf{H}_P \\ \mathbf{H}_C \end{bmatrix} \odot \left(1 - \begin{bmatrix} \mathbf{H}_L \\ \mathbf{H}_P \\ \mathbf{H}_C \end{bmatrix}\right) \odot \left(\mathbf{U}^{\mathbf{T}}\delta_{out}\right) \end{cases} . \tag{3.7}$$

Here $\odot$ is an element-wise matrix multiplication operation, and matrix $\mathbf{Y}$ contains correct class labels of all training samples in the form of one-hot-encoding representation. Input-hidden layer is not fully connected and contains multiple branches. So weights of the input-hidden layer can be updated by spiting gradients obtained at the hidden layer in the following manner:

$$\begin{cases} \mathbf{W}_L^{new} = \mathbf{W}_L^{old} - \lambda \mathbf{X}_L \delta_{mid}[0 : KM, :]^{\mathsf{T}}, \\ \mathbf{W}_P^{new} = \mathbf{W}_P^{old} - \lambda \mathbf{X}_P \delta_{mid}[KM : 2KM, :]^{\mathsf{T}}, \\ \mathbf{W}_C^{new} = \mathbf{W}_C^{old} - \lambda \mathbf{X}_C \delta_{mid}[2KM : 3KM, :]^{\mathsf{T}}, \\ \mathbf{b}_L^{new} = det\Big(inv\left(\mathbf{W}_L^{new}\right)\Big), \\ \mathbf{b}_P^{new} = det\Big(inv\left(\mathbf{W}_P^{new}\right)\Big), \\ \mathbf{b}_C^{new} = det\Big(inv\left(\mathbf{W}_C^{new}\right)\Big). \end{cases} \tag{3.8}$$

Here λ is a , *det* and *inv* compute respectively the determinant and the inverse of coherency matrix represented as real-valued vector in each column of the input matrix.

Algorithm 3 and 4 show training and testing steps of the proposed e-OWN for the classification of multifrequency data. Once the training of the e-OWN is complete we can measure its generalization capability by testing its performance on unseen test data. The generalization capability of any learning based algorithm indicates how well the algorithm performs on the data that it has not seen previously. For all test samples we do the forward pass on e-OWN and obtain their prediction. Once the prediction is obtained we assign labels to each test sample for which its prediction value is largest. Algorithm 4 shows the steps of assigning labels to the test samples.

Algorithm 3 Training Algorithm for e-OWN

N = Number of training samples, M = Number of classes, PM = Size of hidden layer, $iter$ = Number of iterations, λ = Learning rate

$\mathbf{Y} = \left[\mathbf{y}_1^l, \mathbf{y}_2^l, ..., \mathbf{y}_N^l\right]$, where non-zero entry in $\mathbf{y}_i^l$ indicates true class label of $\mathbf{t}_q^{I(i)}$.

for each frequancy band $q \in \left\{q_1, q_2, \ldots, q_n\right\}$ **do**

$\mathbf{T} = \left[\mathbf{t}_q^{I(1)}, \mathbf{t}_q^{I(2)}, \ldots, \mathbf{t}_q^{I(N)}\right] \odot \mathbf{p}$ Where $I(i)$ is i^{th} value of set I which represents the index of i^{th} training sample.

$\mathbf{T}_q = \left[\mathbf{T}; \mathbf{1}\right]$

for $m = 1, 2, ..., M$ **do**

Let $\Omega_t^m = \{\langle \mathbf{T} \rangle_{(q)}^i | \forall i \in I_m\}$ be a set of training samples of class m.

From training set Ω_t^m find P cluster center $\mathbf{w}_1^m, \mathbf{w}_2^m, \ldots, \mathbf{w}_P^m$ using k-means algorithm.

end for

$\mathbf{W}_q = \left[f((\mathbf{w}_1^1)^{-1}), f((\mathbf{w}_2^1)^{-1}), \ldots, f((\mathbf{w}_P^M)^{-1})\right]$

$\mathbf{b}_q = \left[\ln|\mathbf{w}_1^1|, \ln|\mathbf{w}_2^1|, \ldots, \ln|\mathbf{w}_P^1|, \ln|\mathbf{w}_1^M|, \ldots, \ln|\mathbf{w}_P^M|\right]$

end for

for $i = 1, 2, ..., iter$ **do**

for each frequancy band $q \in \left\{q_1, q_2, \ldots, q_n\right\}$ **do**

$\mathbf{H}_q = sigmoid\left(\mathbf{W}_q^T \mathbf{T}_q + \mathbf{b}_q\right)$

end for

$\mathbf{H} = \left[\mathbf{H}_{q_1}; \mathbf{H}_{q_2}; \ldots; \mathbf{H}_{q_n}; \mathbf{1}\right]$

$[\mathbf{U}; \mathbf{c}] = \left(\mathbf{H}\mathbf{H}^{\mathbf{T}}\right)^{-\mathbf{1}}\left(\mathbf{H}\mathbf{Y}^{\mathbf{T}}\right)$

for each frequancy band $q \in \left\{q_1, q_2, \ldots, q_n\right\}$ **do**

$\mathbf{W}_q = \mathbf{W}_q - \lambda \frac{\delta E}{\delta \mathbf{W}_q}$

for $i = 1, 2, ..., PM$ **do**

$\mathbf{b}_q[i] = \ln\left(\left|(f^{-1}(\mathbf{W}_q[i]))^{-1}\right|\right)$

end for

end for

end for

3.3 Experiments and results

To evaluate the performance of e-OWN for multifrequency data classification, we have used Flevoland [11] and Landes datasets [1] obtained with L, P, and C frequency bands on July 3, 1991 and June 20, 1991 respectively. Figure 3.2 shows Pauli decomposition of all bands of the Flevoland dataset. It has 16 classes, namely, Oats, Grass, Lucerne, Flax, Barley, Peas, Fruit, Potato, Beans, Beet, Build-

Algorithm 4 Predict Algorithm

L = Number of test samples
Obtain matrices $\mathbf{W}_{q_1}, \mathbf{W}_{q_2}, \ldots \mathbf{W}_{q_n}, \mathbf{b}_{q_1}, \mathbf{b}_{q_2}, \ldots \mathbf{b}_{q_n}, \mathbf{U}$ and $\mathbf{c}$ from Algorithm 3.
for each frequancy band $q \in \{q_1, q_2, \ldots, q_n\}$ **do**
$\mathbf{T}_q = \left[\mathbf{t}_q^1, \mathbf{t}_q^2, \ldots, \mathbf{t}_q^L\right] \odot \mathbf{p}$ Where $\mathbf{t}_q^i$ is i^{th} test sample of band q.
$\mathbf{H}_q = sigmoid\left(\mathbf{W}_q^T \mathbf{T}_q + \mathbf{b}_q\right)$
end for
$\mathbf{H} = \left[\mathbf{H}_{q_1}; \mathbf{H}_{q_2}; \ldots; \mathbf{H}_{q_n}; \mathbf{1}\right]$
predition = $\left(\mathbf{U}^\mathsf{T}\mathbf{H} + \mathbf{c}\right)$
return arg max$\left(\mathbf{prediction}\right)$

ing, Onions, Wheat, Road, Rapeseed and Maize. Let's call it Flevoland16 dataset. Figure 3.7 shows Pauli decomposition of all bands of Landes dataset. It is of a forest area with trees of different ages, whose ground truth map was also obtained [22].

3.3.1 Performance evaluation of e-OWN

Figure 3.3 shows classification maps of individual bands and different band fusions. Table 3.1 shows corresponding classwise accuracies. Among the three bands, the P band has the longest wavelength and has very good penetration capability. But because of its long wavelength, it faces difficulty in differentiating similar crops. C band has a short wavelength with limited penetration capability, so the volume scattering mechanism is not fully exploited [43]. Wavelength of L band is higher than the C band but lower than the P band, so it has an advantage of reasonable penetration and discrimination. The same is evident from Table 3.1.

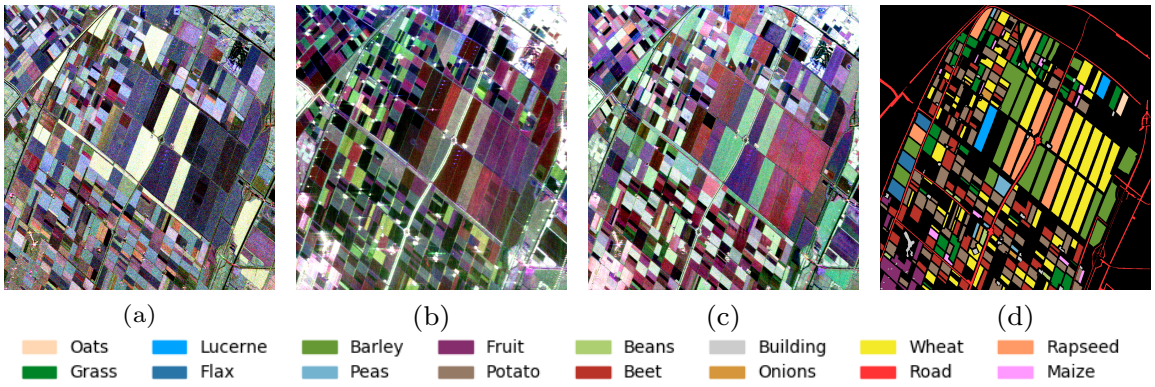


Figure 3.2: Pauli Decomposition of (a) C band, (b) P band, (c) L band and (d) ground truth of Flevoland16 dataset.

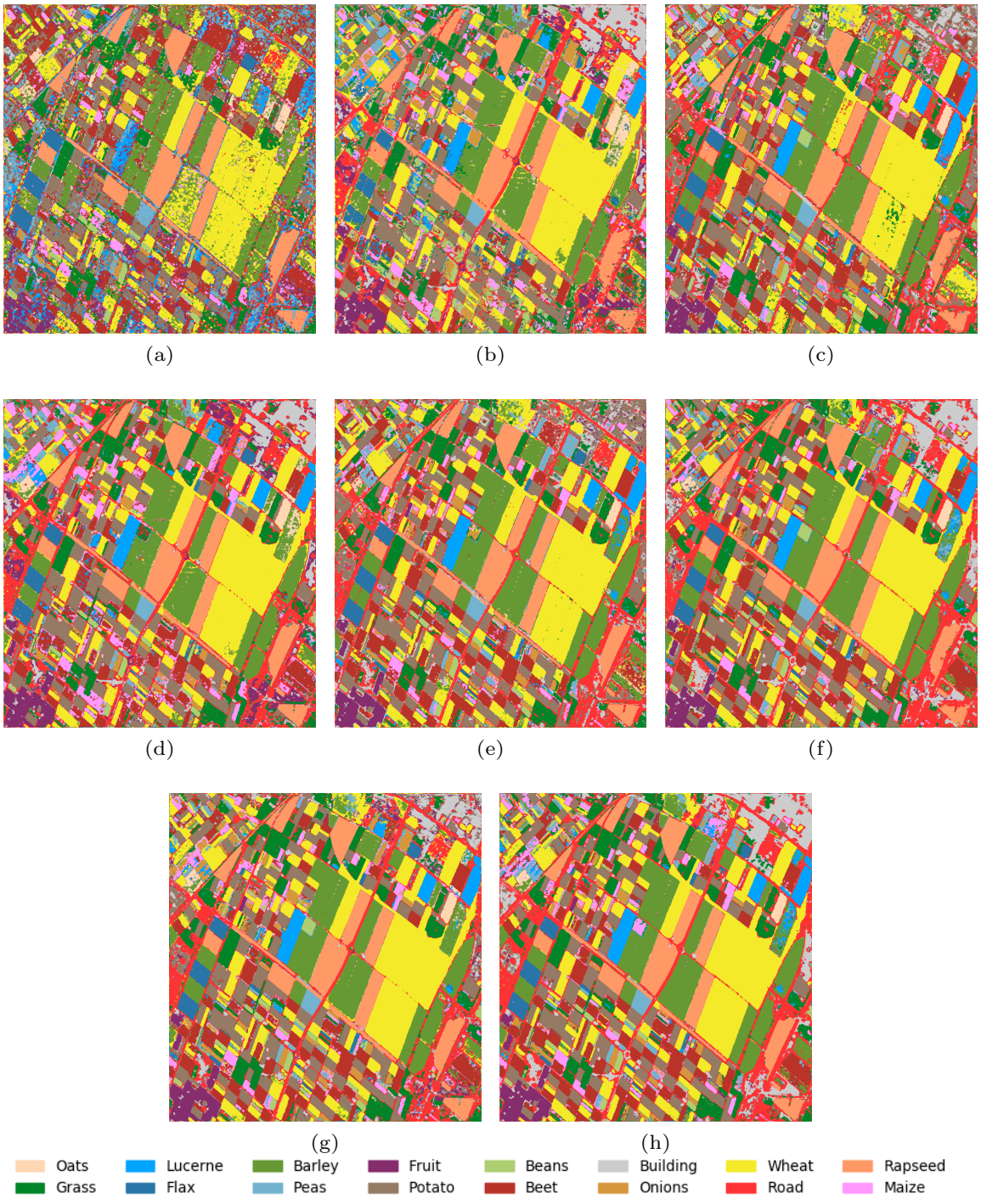


Figure 3.3: (a)-(c) are classification maps of C, P and L band individually. (d)-(f) classification maps of different band fusions obtained using proposed e-OWN, namely, CP, CL and PL, (g) classification map of CPL bands obtained using majority voting (CPL^{+}). (h) classification map of CPL bands obtained using e-OWN

Table 3.1: Class-wise accuracies for different band fusion

Class/Band	C	P	L	$CPL^{\dagger}$	CP	CL	PL	CPL
Oats	99.35	98.37	99.40	**100**	**100**	99.84	**100**	**100**
Grass	72.20	66.19	80.33	88.08	86.55	83.55	85.27	**90.85**
Lucerne	63.83	0.945	99.68	99.89	99.46	99.93	99.88	**99.98**
Flax	99.29	89.76	98.57	**99.55**	99.14	**99.55**	98.85	99.40
Barley	80.69	94.82	96.78	**99.28**	97.97	99.14	98.85	99.00
Peas	83.78	75.62	93.86	96.94	91.29	**98.07**	96.92	98.00
Fruit	73.94	90.24	96.33	**98.79**	95.37	94.03	95.38	96.16
Potato	90.24	93.43	95.43	**99.90**	98.55	98.33	98.48	98.96
Beans	99.60	85.97	98.79	96.77	99.75	99.75	98.79	**99.80**
Beet	44.95	54.44	76.39	78.77	83.93	91.36	91.37	**95.70**
Building	32.33	74.75	54.88	86.83	73.71	72.79	86.22	**89.65**
Onions	95.24	77.61	98.32	98.77	97.82	98.55	**98.87**	98.69
Wheat	85.20	94.45	94.42	**99.09**	97.27	97.06	98.02	98.94
Road	39.56	55.37	60.99	75.96	75.68	76.52	81.63	**83.47**
Rapseed	99.74	98.56	99.64	**99.94**	99.76	99.80	99.76	99.84
Maize	63.22	69.14	91.52	95.92	92.58	96.46	95.60	**98.70**
OA	76.43	84.17	89.62	94.79	93.53	94.40	95.26	**96.71**
AA	76.45	82.08	89.71	94.65	93.05	94.05	95.24	**96.70**

The first three columns of Table 3.1 compare class-wise accuracies obtained using individual bands. For that, we have used OWN proposed in the previous Chapter. As one can see, the L band achieved the highest overall accuracy of 89.62% in individual bands. The worst overall accuracy of 76.43% was obtained by the C band due to its low penetration capability. Performance on P band was in between C and L bands with 84.17%.

Combining multiple bands information can further improve classification accuracy since data of different bands are statistically independent [78]. Meaning, the Wishart distance of some pixel t of class m to cluster center of m is going to be different and unrelated for different bands. So different bands information is going to be complementary to each other. This leads to improved classification accuracy. One way to utilize different bands information is to do ensemble learning of OWN trained on different frequency data.

In the context of machine learning ensemble learning is a supervise learning method of combining finite set of classifiers to produce one strong classifier whose performance is better than all classifiers in the set [27]. Majority voting classifier [14] is one type of ensemble learning method where we train each classifier in a set individually. After that, to obtain the label of any test sample, we obtain

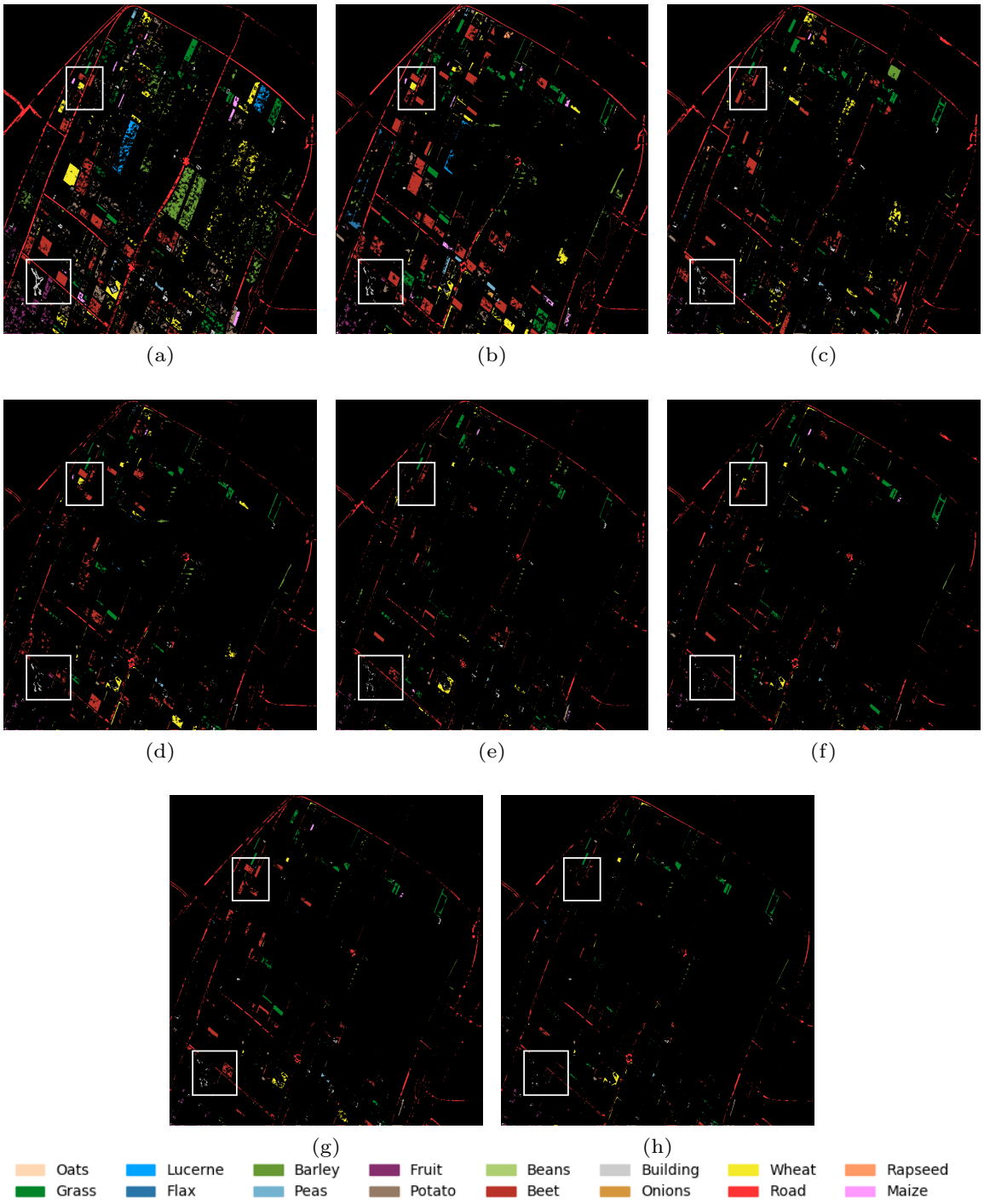


Figure 3.4: (a)-(c) are error maps of C, P and L band individually. (d)-(f) error maps of different band fusions obtained using proposed e-OWN, namely, CP, CL and PL, (g) error map of CPL bands obtained using majority voting ($CPL^{\dagger}$). (h) error map of CPL bands obtained using e-OWN. Here black pixel indicates correctly classified pixel and colored pixel indicated incorrectly classified pixel. Color of the pixel indicates true class label.

the prediction for that sample from all classifiers in the set. Then we take the average of all predictions to produce the final label for the test sample. In our experiment we tried to see what ensemble learning method would result. Let $CPL^{\dagger}$ be the ensemble of three OWN trained on C, P, and L bands respectively. Result is shown in the fourth column of the Table 3.1. It was found that the ensemble learning improved the overall accuracy from 89.62% to 94.79%. But still we found many pixels misclassified as shown in the Figure 3.4(g). The last four columns of Table 3.1 show class-wise accuracies obtained by combining different bands information using the proposed e-OWN. As one can see, e-OWN is able to efficiently combine various bands information to achieve better accuracy compared to that of the individual bands as well as ensemble learning. Figure 3.3 shows classification maps of individual bands and different band fusions.

To further investigate the efficiency of e-OWN, we look at the error maps of individual bands and different bands fusion as shown in Figure 3.4. Here a black pixel indicates that the pixel was classified correctly and a different colored pixel indicates that the pixel was incorrectly classified, the color indicates true class label of that pixel. From Figures 3.4(a)-(c), we can observe some overlap between misclassified pixels in individual bands highlighted within white box. We can also observe areas where pixels misclassified in one band are correctly classified in another. Figures 3.4(g) and 3.4(h) show error maps of combining all three bands using ensemble learning and by proposed e-OWN respectively. As we can see, ensemble learning is able to improve the classification result by combining different bands, but it fails in some region when a pixel is misclassified in all three bands (highlighted by the white box), in Figure 3.4(g). In case of ensemble learning, network was trained independently on each band, so it does not have the ability of combining different bands features. On the other hand e-OWN gives much improved classification result shown in Figure 3.4(h).

3.3.2 Generalization capability of e-OWN

Generalization capability of an algorithm indicates how the algorithm performs on an unseen data. To test the generalization capability of e-OWN with various size of training data, we experiment with the number of samples used for training, namely, sampling rate versus classification accuracy. The results for the same are shown in Figure 3.5. We vary the sampling rate from 0.05% to 10% on the x-axis. For each sampling rate, we train e-OWN using all three frequency bands (CPL) data and report the classification accuracy on the y-axis. As we can see, even with using only 0.05% samples, we achieve 95.8% accuracy. It shows that the proposed

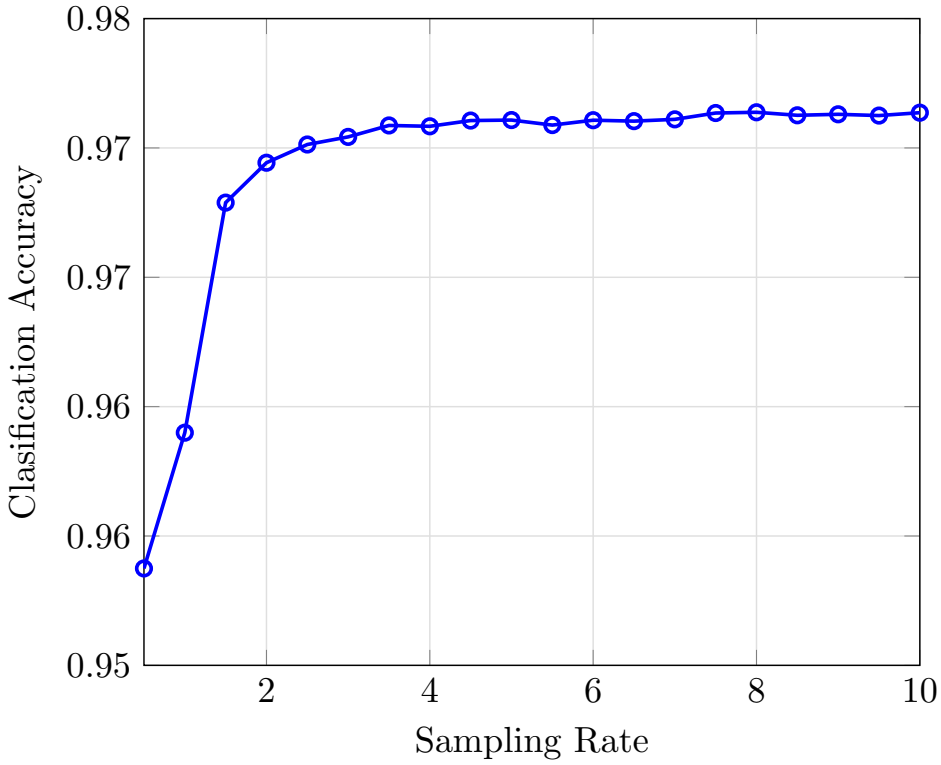


Figure 3.5: Classification accuracy on Flevoland16 dataset at different sampling rate (from 0.05% to 10% of total labeled pixels).

e-OWN requires very few training samples. As we increase the sampling rate, classification accuracy increases until around 3%. At 3% sampling rate the overall classification accuracy reaches to 97.1%. After 3%, the increase in classification accuracy does not remain that sharp. Figure 3.5 shows that at 10% sampling rate the classification accuracy reaches to 97.3%.

3.3.3 Comparison with ANN [22]

In this section, we compare the proposed e-OWN with autoencoder based ANN architecture proposed in [22]. For comparison, we have used two dataset. First dataset is a subset of the Flevoland16 dataset shown in Figure 3.6, let us call this subset Flevoland7. Second is Landes dataset shown in Figure 3.7. Table 3.2 compares the performance of e-OWN and ANN [22] on single as well as multifrequency data. It should be noted that for the single frequency data, e-OWN will have only one input branch. So, its architecture will be same as that of the OWN discussed in Chapter 2.

For the Flevoland7 dataset, in a single-band case, e-OWN achieves higher accuracy for the L band but fails to show better results in C and P bands. In case

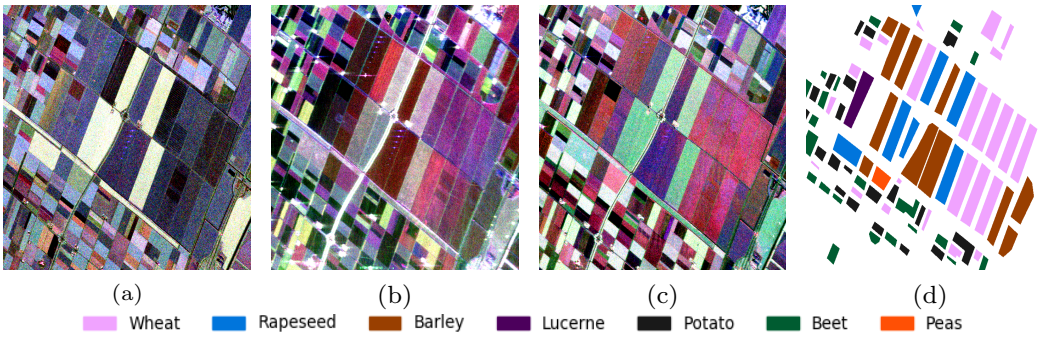


Figure 3.6: Pauli Decomposition of (a) C band, (b) P band, (c) L band and (d) Ground Truth of Flevoland7 dataset.

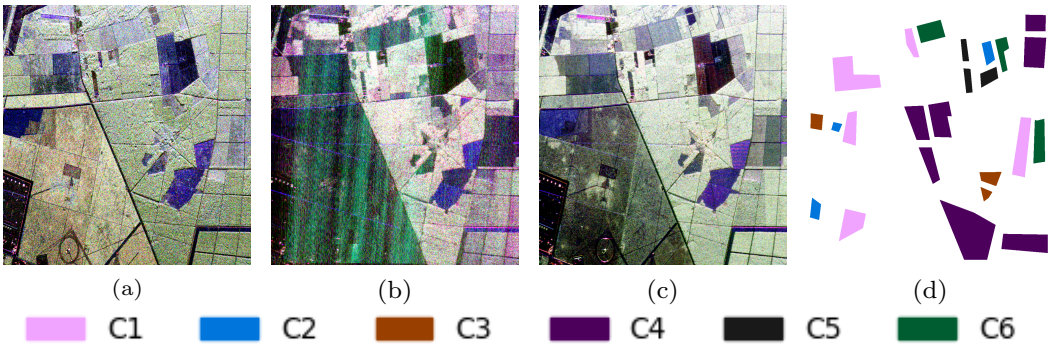


Figure 3.7: Pauli Decomposition of (a) C band, (b) P band, (c) L band and (d) Ground Truth of Landes dataset.

of P band it achieves above 90% accuracy for Wheat and Rapeseed while poor performance is observed for Lucerne which is misclassified as Beat. In case of C band it fails to differentiate between Beat, Peas and Lucerne classes but achieves good result in case of Rapeseed where we observe 95.68% accuracy. For C band worst performance is observed in case of Peas which is misclassified as Potato. In case of L band, we observe over 90% accuracy for Wheat, Rapeseed, Potato and Barley. Worst performance is observed in case of Beat with 80.87% accuracy. It is misclassified as Wheat. One can see that e-OWN achieves higher accuracy in most of the band fusions (CL, PL, and CPL) cases. For CP band fusion case accuracy of e-OWN is lower, only 93.72%. It may be attributed to even lower accuracies of C (73.62%) and P (87.79%) bands.

In the case of Landes dataset, ANN [22] results are available from the C and L band. The Landes dataset contains trees of different ages. In case of individual bands we observed similar pattern as that of Flevoland7. L band achieved best accuracy of 93.26% followed by P band with 87.33% accuracy. Performance of C band was worst with 78.52% accuracy due to its lower penetration capability. Combining all three bands, e-OWN was able to achieve 97.28% overall classifica-

Table 3.2: Classification Accuracy Comparison for e-OWN

Band	Flevoland7		Landes	
	ANN [22]	OWN	ANN [22]	OWN
C	**89.78**	73.62	77.42	**78.52**
L	90.86	**92.17**	89.73	**93.26**
P	**90.75**	87.79	-	**87.33**
Bands	ANN [22]	e-OWN	ANN [22]	e-OWN
CP	**96.59**	93.72	-	**93.24**
CL	95.58	**96.40**	**96.23**	96.18
PL	96.40	**97.70**	-	**96.58**
CPL	98.23	**98.65**	-	**97.28**

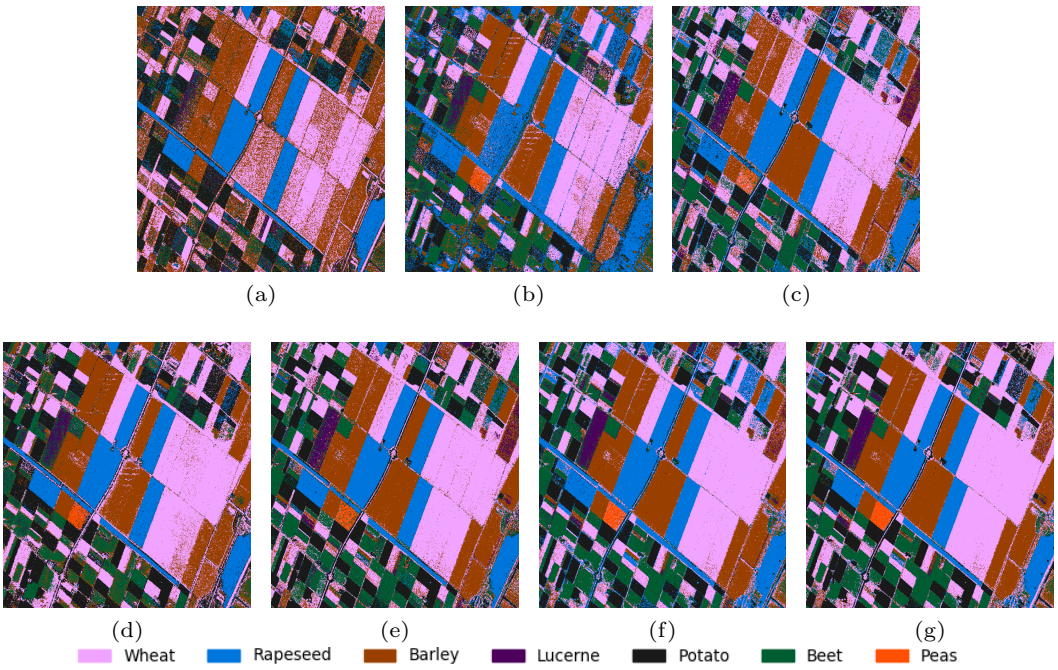


Figure 3.8: Classification results of (a) C band, (b) P band, (c) L band, (d) Combining C and P band, (e) Combining C and L band, (f) Combining L and P band and (g) Combining C, L and P band of Flevoland7 dataset.

tion accuracy.

From the experiments on both datasets, it is evident that even with only a single hidden layer, e-OWN can combine multiple band information more efficiently than ANN [22] with multiple hidden layers. The explanation for this is the fact that e-OWN considers polarization information of PolSAR data [63], while ANN [22] does not. Also, e-OWN starts with relatively high classification accuracy, because the first iteration of e-OWN achieves classification accuracy as high

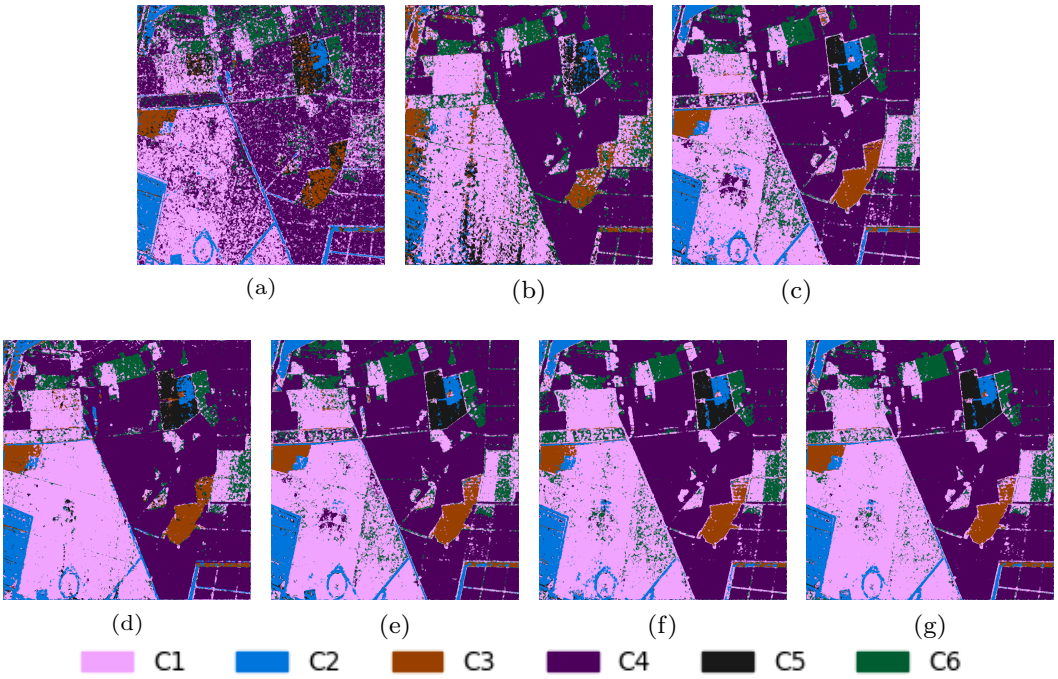


Figure 3.9: Classification results of (a) C band, (b) P band, (c) L band, (d) Combining C and P band, (e) Combining C and L band, (f) Combining L and P band and (g) Combining C, L and P band of Landes dataset.

as Wishart classifier. This helps e-OWN to converge quickly with high accuracy. Figures 3.8 and 3.9 show classification maps of C, L, P, CP, CL, PL, and CPL bands combination for Flevoland7 and Landes datasets. For individual bands, in both datasets L band performs best which shows the superiority of L band PolSAR image for crop classification. Out of all possible band fusions, the highest classification accuracy was observed in the case of CPL.

3.4 Conclusion

In this chapter, we defined and used a new network named e-OWN (Extended Optimized Wishart Network) for the classification of multifrequency PolSAR data. We started with proposing linearization model of Wishart classifier for multifrequency PolSAR data. Using this model, we developed single hidden layer, multi input branch neural network called e-OWN. Input layer of e-OWN has multiple branches, one for each frequency band of a PolSAR image. During forward pass, the input-hidden layer calculates Wishart distance of each training pixel of each frequency band with all cluster centers of its respective frequency band. At the hidden layer, we stack these distance values to obtain a combined feature vector for each pixel. Hidden-output layer uses this feature vector to perform

classification. We performed experiments on Flevoland and Landes dataset and demonstrated that proposed e-OWN could efficiently combine multiple bands information in comparison with ensemble training and ANN [22].

Currently, both OWN and e-OWN use a k-means algorithm to find multiple cluster centers from each class. However, the k-means algorithm has a limitation: since it starts with randomly initialized cluster centers, its convergence to optimal clusters is not guaranteed. Global k-means can solve this problem of convergence [82]. Given the value of K, the global k-means always finds optimal clusters for every class. The second advantage of the global k-means is that to find a solution for K clusters, it iteratively solves the problem for $k = 1, 2, \ldots, K$. This characteristic may help in estimating the correct value of K. On this logic, in the next chapter, we will discuss the global k-means algorithm for PolSAR data.

CHAPTER 4
Global k-means for a PolSAR Image

4.1 Introduction

In Chapter 2, we discussed OWN for single frequency data, and in Chapter 3, we introduced e-OWN for multifrequency data. Both OWN and e-OWN used the k-means algorithm to find multiple cluster centers from each class of each frequency band. To increase the number of hidden units, we showed that coherency matrices found by k-means outperformed randomly selected coherency matrices. We showed that both OWN and e-OWN resulted in better classification efficiency. However it has some limitations too. The first disadvantage of this approach is that the cluster centers located by k-means are not guaranteed to be optimal. The second drawback is that Euclidean distance is used in k-means to compare two PolSAR pixels, which is not appropriate for PolSAR data since the PolSAR pixel follows Wishart distribution. To solve these problems, we propose to use global k-means, which guarantees optimal convergence of cluster centers. And instead of Euclidean distance, we recommend using Revised Wishart Distance (RWD), which considers the probability distribution of PolSAR data [66, 106, 135]. One thing here to note is that the computation of RWD is more expensive than Wishart distance. RWD requires to compute determinants of all training samples as well as test samples. So we have to find out way to calculate determinant fast. In section 4.2, we present fast coherency matrix inversion and determinant calculation using real-valued vector representation proposed in Chapter 2. In section 4.3, we present a fast variant of the global k-means algorithm designed for PolSAR data. It utilizes the proposed real-valued vector representation to perform all coherency matrix related computations.

4.2 Fast Matrix Inversion and Determinant Computation

In Chapters 2 and 3, the fast calculation of Wishart distance demonstrated the effectiveness of the real-valued vector representation, which resulted in outperforming the existing methods. In this section, we will show that real-valued vector representation is also useful for fast inverse and determinant computation of a coherency matrix. Coelho *et al*. [21] proposed an algorithm for fast matrix inversion and determinant computation. It used 6 elements of complex coherency matrix for this calculation. The authors proved that it was two times faster than the usual Cholesky factorization. The real-valued representation shown in eq. (2.11) of Chapter 2 can directly be used to calculate the real-valued representation of the inverse of a matrix and determinant of a matrix. We have modified the algorithm proposed by Coelho *et al*. [21] to directly work on the real-valued vector representation shown in eq. (2.11). Algorithm 5 calculates the real-valued representation of the inverse of the input matrix and the determinant of the input matrix directly from the real-valued vector representation of any PolSAR pixel.

Algorithm 5 Fast Matrix Inversion and Determinant

Input $\mathbf{v} = f(\mathbf{A}) = [a, b, c, d, e, f, g, h, j]$ real-valued vector representation of matrix A.

$$
\begin{aligned}
a_i &= f * j - g^2 - h^2 & b_i &= e * h + d * g - b * j \\
c_i &= e * g - h * d - c * j & d_i &= b * g - c * h - f * d \\
e_i &= b * h + c * g - f * e & f_i &= a * j - d^2 - e^2 \\
g_i &= e * c + d * b - a * g & h_i &= e * b - d * c - a * h \\
j_i &= a * f - b^2 - c^2 & & \\
det(\mathbf{v}) &= a * a_i + b * b_i + c * c_i + d * d_i + e * e_i & & \\
inv(\mathbf{v}) &= [a_i, b_i, c_i, d_i, e_i, f_i, g_i, h_i, j_i]/det(\mathbf{v}) & &
\end{aligned}
$$

Output $inv(\mathbf{v})$ real-valued vector representation of $\mathbf{A}^{-1}$ and $det(\mathbf{v})$ determinant of $\mathbf{A}$.

4.3 Fast Global k-means for a PolSAR Image

PolSAR imaging may not necessarily have homogeneity in the distribution of all the pixels within even the same class [43]. Keeping this in mind OWN [41] was proposed which considered k-means algorithm to quickly find multiple cluster centers within each class. But, k-means algorithm has a limitation: since it starts

with randomly initialized cluster centers, its convergence to optimal clusters is not guaranteed. This problem of convergence may be solved by global k-means [82]. Given the value of K, global k-means always finds optimal cluster centers. Second advantage of global k-means is that in order to find a solution for K clusters, it iteratively solves the problem for $k = 1, 2, \ldots, K$. This characteristic may help in estimating the correct value of K.

Let $\mathbf{X} = \{\langle \mathbf{T}_1 \rangle, \langle \mathbf{T}_2 \rangle, \ldots, \langle \mathbf{T}_N \rangle\}$ be N coherency matrices of a class m. We want to find K clusters such that it partitions these coherency matrices into K disjoint subsets whose centroids are represented as $\langle \mathbf{C}_1 \rangle, \langle \mathbf{C}_2 \rangle, \ldots, \langle \mathbf{C}_K \rangle$. In this case, clustering error function E is expressed as follows:

$$E = \sum_{i=1}^{N} \sum_{k=1}^{K} I(\langle \mathbf{T}_i \rangle \in \langle \mathbf{C}_k \rangle) * d(\langle \mathbf{T}_i \rangle, \langle \mathbf{C}_k \rangle). \tag{4.1}$$

Here $I(\langle \mathbf{T}_i \rangle \in \langle \mathbf{C}_k \rangle) = 1$ if $d(\langle \mathbf{T}_i \rangle, \langle \mathbf{C}_k \rangle)$ is minimum $\forall k = 1, 2, \ldots, K$ and 0 otherwise. Clustering error (E) measures on an average how distant every sample is from their respective closest cluster center. Lower value of E indicates better cluster centers. Global k-means algorithm [82] starts from single cluster and iteratively solves the problem for $k = 2, 3, \ldots, K$. Solving for $k = 1$ is easy, it can be done by taking mean of all points, i.e., centroid of all the coherency matrices. Next, to solve problem with more than one cluster, i.e., $k > 1$, we retain $k - 1$ cluster centers from previous stage and perform N executions of the k-means algorithm. During each execution we consider one of the N pixels along with previous $k - 1$ solutions as initial cluster centers and run the k-means until it converges. After the N executions we select the solution which achieves minimum clustering error shown in eq. (4.1). We repeat this process until K cluster centers are obtained. As we can see, to find each cluster center we have to perform N executions of k-means algorithm, the complete operation becomes computationally costly. To reduce the computational cost, a fast variant of global k-means has also been suggested [82]. Instead of running k-means algorithm for each N pixels, the upper bound $E_n \leq E - b_n$ of the clustering error function shown in eq. (4.1) for all samples is calculated with

$$b_n = \sum_{j=1}^{N} max(d_{k-1}^{j} - d(\langle \mathbf{T}_n \rangle, \langle \mathbf{T}_j \rangle), 0). \tag{4.2}$$

Here d_{k-1}^{j} denotes the distance between $\langle \mathbf{T}_j \rangle$ and the closest mean coherency matrix among all the $k - 1$ cluster centers obtained so far. The value of b_n represents the assured reduction in the error measure obtained if we consider $\langle \mathbf{T}_n \rangle$ as a new

initial cluster center.

It has been reported that instead of Euclidean distance, a revised Wishart distance (RWD) is more suitable to a PolSAR data as a distance measure [52, 135]. Unlike Wishart distance, RWD is always positive and it has a minimum constant distance value of 0. RWD between two PolSAR coherency matrices $\langle \mathbf{T}_j \rangle$ and $\langle \mathbf{T}_n \rangle$ is given as

$$D_{RWD}(\langle \mathbf{T}_n \rangle, \langle \mathbf{T}_j \rangle) = \text{trace}(\langle \mathbf{T}_j \rangle^{-1} \langle \mathbf{T}_n \rangle) + \ln \left(\frac{|\langle \mathbf{T}_j \rangle|}{|\langle \mathbf{T}_n \rangle|} \right) - n. \tag{4.3}$$

For each value of $k = 1, 2, \ldots, K$, we have to calculate b_n for all N samples which is still a computationally costly operation. To reduce the computational overhead we use real-valued vector representation for a coherency matrix along with fast determinant and inverse computation algorithm discussed in section 4.2.

Algorithm 6 Fast Global k-means for PolSAR data

K = Number of cluster, N = Number of PolSAR pixels, $\{\langle \mathbf{T}_1 \rangle, \langle \mathbf{T}_2 \rangle, \ldots, \langle \mathbf{T}_N \rangle\}$ coherency matrices of N PolSAR pixels, $\mathbf{t}_i = f(\langle \mathbf{T}_i \rangle)$.
$\mathbf{X} = [\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_N]$
$\mathbf{X}_{inv} = [inv(\mathbf{t}_1), inv(\mathbf{t}_2), \ldots, inv(\mathbf{t}_N)]$
$\mathbf{b} = [det(\mathbf{t}_1), det(\mathbf{t}_2), \ldots, det(\mathbf{t}_N)]^{\mathsf{T}}$
$\mathbf{B} = ln(\frac{\mathbf{b}}{\mathbf{b}^{\mathsf{T}}})$
$\mathbf{D} = (\mathbf{X} \odot \mathbf{p})^{\mathsf{T}} \mathbf{X}_{inv} + \mathbf{B}$
$\mathbf{C}_1 = mean(\mathbf{X})$
Let **CloseClusterDistance** be a vector where i^{th} entry corresponds to RWD of T_i pixel to its closest cluster center.
for $k = 2, \ldots, K$ **do**
  ErrorReduction $= add($**CloseClusterDistance**$, -\mathbf{D})$
  $\mathbf{b}_n = sum(max($**ErrorReduction**$, \mathbf{0}))$
  $\mathbf{C}_k = \mathbf{X}[arg\ max(\mathbf{b}_n)]$
  for $j = 1, \ldots, number\ of\ iterartion$ **do**
    $\mathbf{C}_{inv} = [inv(\mathbf{C}_1), inv(\mathbf{C}_2), \ldots, inv(\mathbf{C}_k)]$
    $\mathbf{C}_d = [det(\mathbf{C}_1), det(\mathbf{C}_2), \ldots, det(\mathbf{C}_k)]$
    $\mathbf{B} = ln(\frac{\mathbf{d}}{\mathbf{b}^{\mathsf{T}}})$
    distance $= (\mathbf{C}_{inv} \odot \mathbf{p})^{\mathsf{T}} \mathbf{X} + \mathbf{B}$
    closecluster $= arg\ min($**distance**$)$
    for $i = 2, \ldots, k$ **do**
      $\mathbf{C}_i = \sum_{l=1}^{l=N} \mathbb{I}($**closecluster**$[l] = i)\mathbf{t}_l$
    end for
  end for
  Update **CloseClusterDistance**
end for

For this, we first calculate RWD between each sample to every other sample. Let $\mathbf{t}_i = f(\langle \mathbf{T}_i \rangle)$ and $\mathbf{X} = [\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_N]$ be a matrix containing vector repre-

sentation of all samples, i^{th} column of $\mathbf{X}$ is a vector representation of $\langle \mathbf{T}_i \rangle$. Let $\mathbf{X}_{inv} = [inv(\mathbf{t}_1), inv(\mathbf{t}_2), \ldots, inv(\mathbf{t}_N)]$, then the i^{th} column of $\mathbf{X}_{inv}$ is a vector representation of $\langle \mathbf{T}_i \rangle^{-1}$. Let $\mathbf{b} = [det(\mathbf{t}_1), det(\mathbf{t}_2), \ldots, det(\mathbf{t}_N)]^\mathsf{T}$. Now, RWD between each sample and every other sample can simultaneously be calculated using eq. (4.4), which is

$$\mathbf{D} = (\mathbf{X} \odot \mathbf{p})^\mathsf{T} \mathbf{X}_{inv} + \ln(\mathbf{b}/\mathbf{b}^\mathsf{T}). \tag{4.4}$$

Here i^{th} row of $\mathbf{D}$ contains RWD between $\langle \mathbf{T}_i \rangle$ and $\langle \mathbf{T}_j \rangle$ $\forall j \in 1, 2, \ldots, N$. Algorithm 6 lists steps involved in fast global k-means algorithm designed for the PolSAR data. Here, *mean* is the row-wise mean operation, $add(\mathbf{x}, \mathbf{y})$ adds vector $\mathbf{x}$ to each row of matrix $\mathbf{y}$, *sum* is row-wise summation operation and $arg\ max(\mathbf{x})$ returns the index of largest value in vector $\mathbf{x}$.

4.4 Experiments and results

Experiments are conducted on the following four different real PolSAR image datasets:

- An L-band PolSAR image of the Flevoland Region in The Netherlands obtained using AIRSAR satellite on 16 Aug, 1989. The ground truth [136] of this image is used for discussion of classification accuracy. It contains 15 different classes, such as, Baresoil, Barley, Beet, Buildings, Forest, Grasses, Lucerne, Peas, Potatoes, Rapeseed, Steambeans, Water, Wheat 1, Wheat 2 and Wheat 3. This image is of 750×1024 size. Let us call it Flevoland15 dataset.

- An L-band PolSAR image of the Landes Region in France obtained by AIRSAR satellite on 19 June, 1991. The ground truth [22] of this image is used for discussion of classification accuracy. It comprises of 6 classes of trees of differing ages. It is of 1050×1000 size.

- An L-band PolSAR image of Flevoland region in The Netherlands obtained by AIRSAR satellite on some other date i.e., 16 June, 1991. Its ground truth [22] is used for discussion of classification accuracy. It has 7 different classes, such as, Barley, Beet, Lucerne, Peas, Potato, Rapeseed and Wheat. Its size is 750×700. Let us call it Flevoland7 dataset.

- An C-band PolSAR image of the San Francisco obtained using RADARSAT on 2008. The ground truth for this image is obtained from [93]. It contains

5 classes, namely, Water, Vegetation, Developed, High-density Urban and Low-density Urban. It is of size 1800 $\times$ 1380.

4.4.1 Effectiveness of global *k*-means

To measure the effectiveness of global *k*-means we have compared our method with OWN [41] based on *k*-means. OWN is a single hidden layer network that uses *k*-means algorithm to initialize weights of the input-hidden layer. During forward pass OWN calculates Wishart distance of all training samples with respect to the cluster centers found by *k*-means.

To establish effectiveness of global *k*-means we kept all parameters of OWN same except we replaced k-means algorithm with global k-means algorithm. Let us call this modified method Gk-OWN. During forward pass, Gk-OWN will calculate RWD of all training samples with respect to all cluster centers obtained by the proposed fast global *k*-means algorithm as explained in section 4.3. Gk-OWN has thus two advantages over OWN. First, it uses global *k*-means algorithm which gives optimal cluster centers. Second, it finds RWD in fast global *k*-means which is more appropriate for a PolSAR data than an Euclidean distance used by *k*-means in OWN.

Parameters of both the networks are kept same for true comparison. We kept value of K=4, number of iterations as 100 and learning rate λ as 0.2. Experiments were repeated 50 times. Each time we took randomly selected 5% pixels as training samples for both the methods. The test accuracies are reported on remaining 95%. Figure 4.1 shows comparison of overall accuracies obtained using both methods. As one can infer from Figure 4.1, the overall accuracy (OA) for the Flevoland15 dataset has improved from 92.24% to 93.20%, for Landes dataset OA improved from 91.09% to 93.19% and for Flevoland7 dataset OA improved from

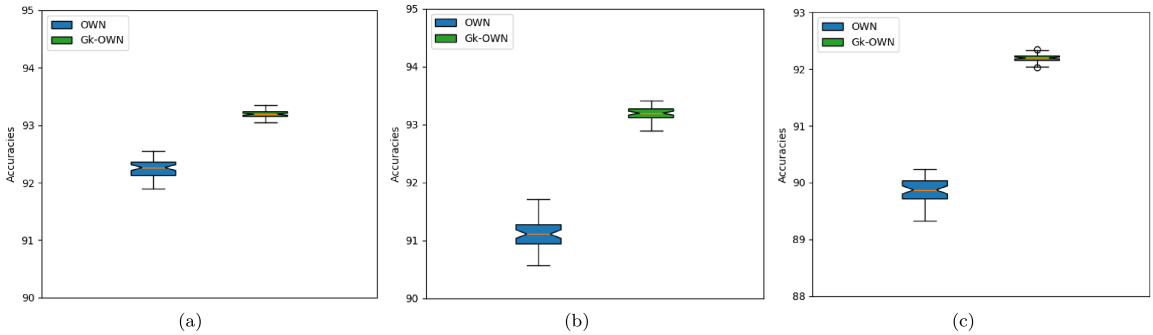


Figure 4.1: Classification accuracies obtained over repeated execution of OWN using *k*-means and Gk-OWN using global *k*-means for (a) Flevoland15 dataset, (b) Landes dataset and (c) Flevoland7 dataset.

89.85% to 92.17% on use of proposed fast global k-means algorithm.

4.4.2 Experiments on Flevoland15 dataset

We compare the training error of WN, OWN and proposed Gk-OWN with 4M hidden units. Figure 4.2 shows the training error of WN, OWN and Gk-OWN on the x-axis. On the y-axis, we show the number of iterations. As we can see, Gk-OWN starts with smaller training error in comparison to both WN and OWN due to a better initialization method. These lead to a faster convergence and improved classification accuracy of Gk-OWN. Next, we compare class-wise classification accuracies of WC [78], WN [63], OWN and Gk-OWN in Table 4.1. Figure 4.3 shows the corresponding classification maps. As we can see, Gk-OWN achieves better classification accuracy in 11 out of 15 classes compared to the remaining three methods. In comparison to OWN, Gk-OWN achieves better accuracy in

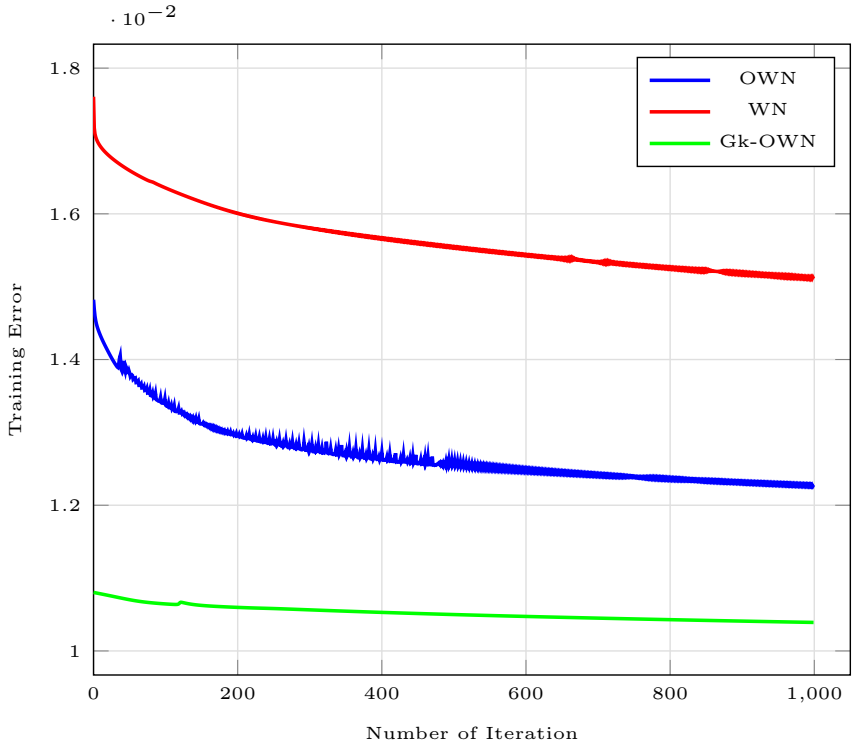


Figure 4.2: Comparison of training error between WN, OWN and Gk-OWN with 4M hidden units.

12 out of 15 classes. The three most improved classes in comparison to OWN were Grasses, Wheat 2 and Buildings. In case of Grasses, accuracy was improved from 80.52% to 88.05%. In case of Wheat 2, accuracy was increased from 81.13% to 84.91% and in case of Buildings, accuracy improved from 77.88% to 80.25%. The three classes where accuracy of Gk-OWN was lower compared to OWN were

Table 4.1: Class-wise Classification Accuracy Comparison

Class	Samples	WC [78]	WN [63]	OWN	Gk-OWN
Stembeans	6103	95.08	85.19	96.38	**97.98**
Peas	9111	96.28	95.55	95.93	**96.67**
Forest	14944	87.91	89.46	90.77	**91.97**
Lucerne	9477	92.93	96.13	96.34	**97.67**
Wheat 1	17283	86.22	**94.48**	93.00	92.67
Beat	10050	95.13	96.06	95.80	**97.64**
Potatoes	15292	87.75	88.53	**89.62**	88.66
Bare soil	3078	99.20	98.33	98.72	**99.19**
Grasses	6269	72.46	78.17	80.52	**88.05**
Rapeseed	12690	74.84	77.23	83.34	**85.00**
Barley	7156	95.26	97.23	**98.20**	96.52
Wheat 2	10591	82.72	72.08	81.13	**84.91**
Wheat 3	21300	88.64	94.55	95.68	**95.71**
Water	13476	51.75	98.39	98.97	**99.93**
Buildings	476	83.40	**84.55**	77.88	80.25
OA		85.04	90.18	92.24	**93.32**

Wheat, Potatoes and Barley. In these classes the accuracy was reduced by 0.33%, 0.96% and 1.68% respectively, which was not so noticeable.

4.4.3 Experiments on San Francisco dataset

We perform experiments on San Francisco dataset. It is of size 1800×1380. It contains 2484000 total pixels out of which 1804087 are labeled. Due to large number of labeled pixels we only used 1% of them as a training data and remaining as testing data. Table 2.3 shows class-wise classification accuracy obtained by WN (4M), OWN (4M) and Gk-OWN (4M) on San Francisco dataset. Figure 2.6 shows corresponding classification maps. Similar to Flevoland15 dataset, OWN achieves 3.68% better OA compare to WN and 0.55% better OA than OWN. In comparison with OWN, the highest improvement was observed for High-Density Urban class with 2.73% better accuracy. For Developed and Vegetation class accuracy was improved 2.04% and 0.29% respectively. For Low-Density Urban class, accuracy of Gk-OWN was 11.61% better than WN but only 0.07% lower than OWN.

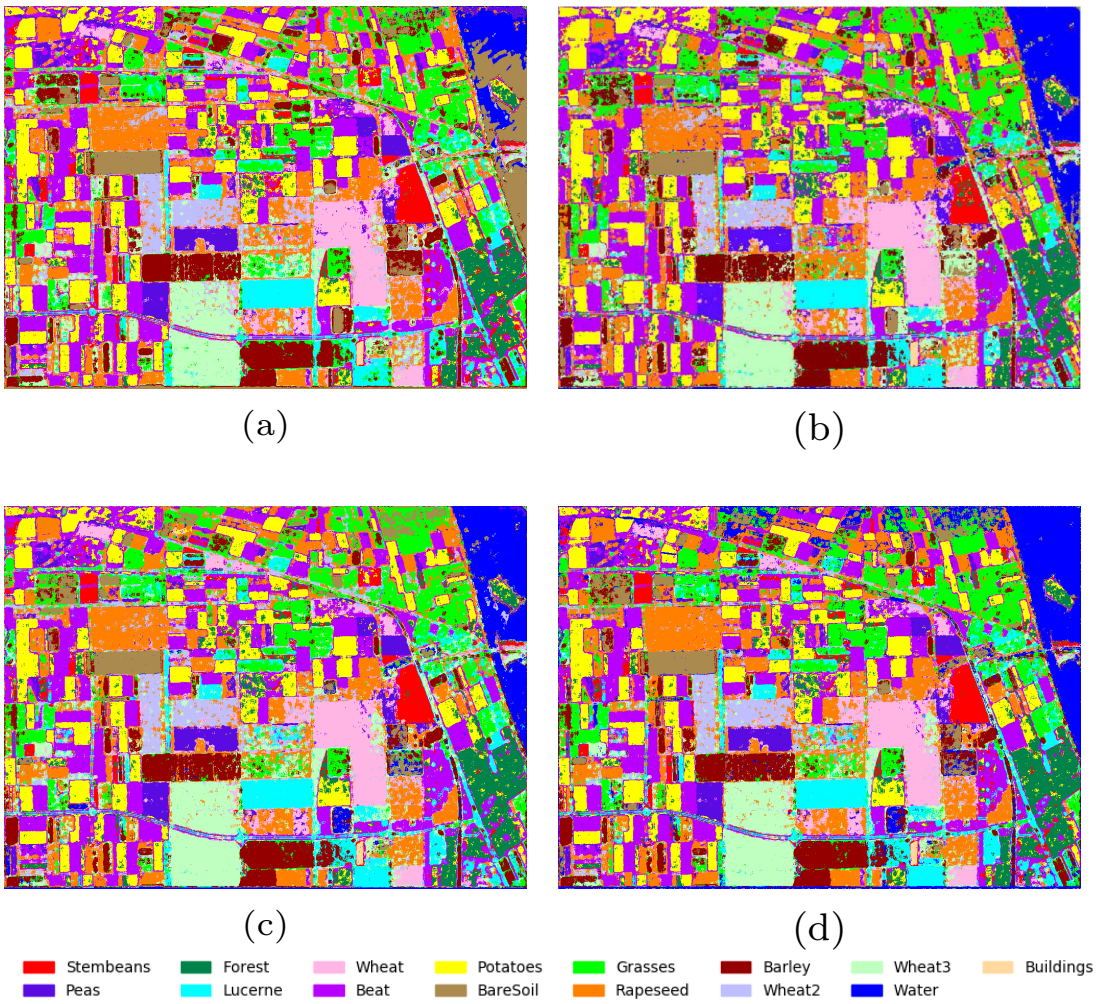


Figure 4.3: Classification map obtained using (a) WN [63], (b) OWN and (c) Proposed Gk-OWN.

Table 4.2: Class-wise accuracies of San Francisco dataset

Class	Samples	WN	OWN	Gk-OWN
Developed	80616	51.97	61.21	**63.25**
Water	852078	99.83	**99.95**	**99.95**
Vegetation	237237	85.73	87.75	**88.04**
High-Density Urban	282975	68.98	69.78	**72.51**
Low-Density Urban	351181	66.45	**78.13**	78.06
OA	-	84.50	87.63	**88.18**
AA	-	74.59	79.36	**80.36**

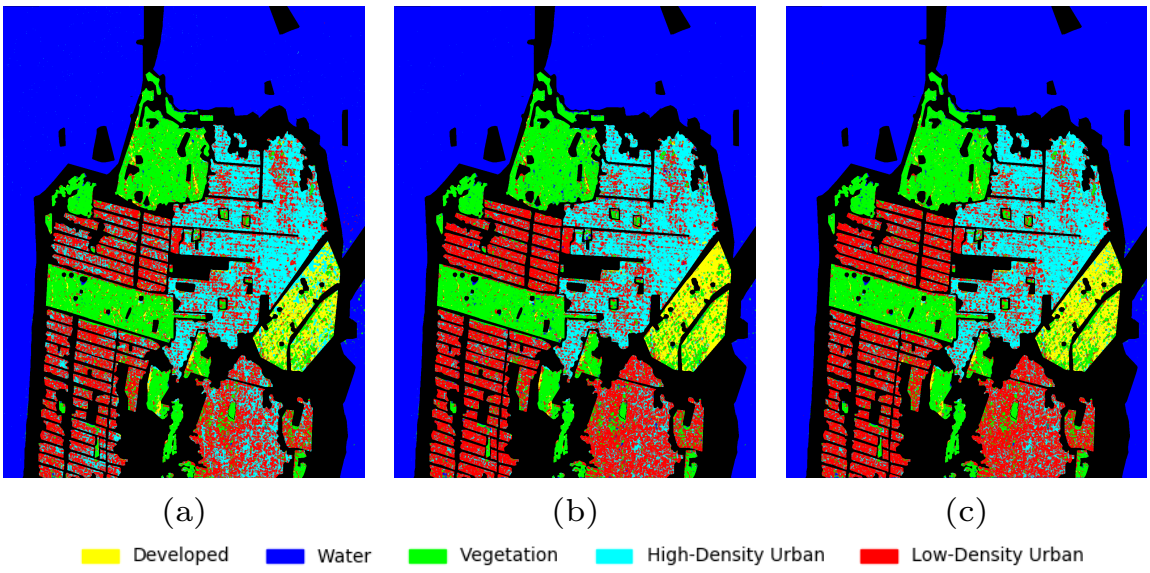


Figure 4.4: Classification map obtained using (a) Wishart network [63], (b) OWN and (c) Gk-OWN.

4.5 Conclusion

In this chapter, we presented the fast global k-means algorithm for PolSAR images. Our proposal addressed two issues associated with k-means algorithm used in OWN and e-OWN discussed in chapter 2 and 3 respectively. First, instead of Euclidean distance, we used the revised Wishart distance in global k-means which considered the probability distribution of the PolSAR data. Second, unlike k-means algorithm, global k-means guarantees the optimal cluster centers. Since the revised Wishart distance calculation is computationally costly, to compensate that we also proposed fast determinant and inverse calculation of coherency matrix using its real-valued vector representation discussed in chapter 2. We showed that global k-means was able to achieve higher classification accuracy as compared to k-means algorithm.

Till now we presented three different algorithms, namely, OWN, e-OWN and Gk-OWN, for classification of single frequency and multifrequency PolSAR images. All three methods are pixel-based classifiers, meaning we classify each pixel independently. Also, PolSAR images suffer from speckle noise. In this case if a pixel is noisy, then it may result into misclassification. To reduce the effect of speckle noise on classification we can use spatial information present in the image. In the next chapter we propose a superpixel driven OWN which uses superpixels to capture the spatial information in the PolSAR images.

CHAPTER 5
Superpixel Driven Optimized Wishart Network

5.1 Introduction

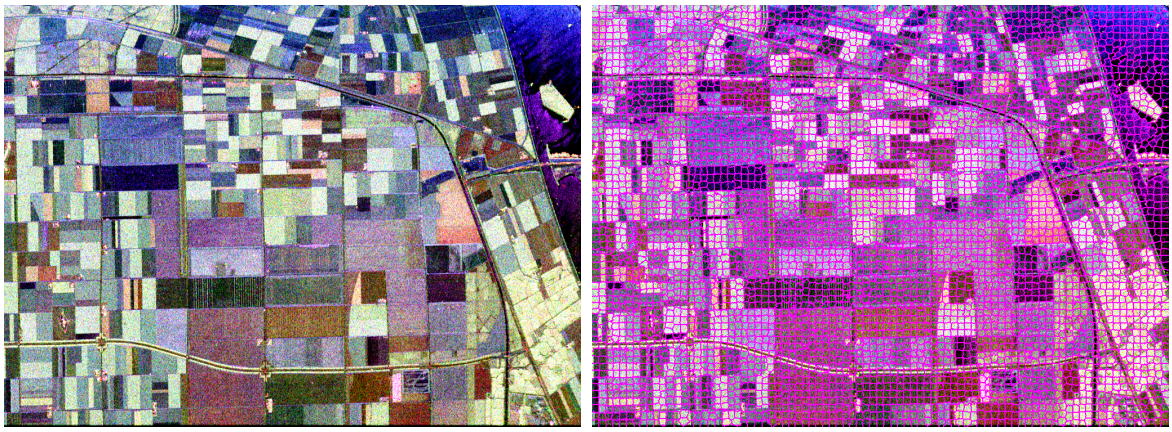
Generally, we find two types of algorithms to address PolSAR image classification problem. One exploits spatial information to perform classification and other uses pixel based classification technique. It has been observed that spatial information based classification algorithm results in better accuracy in comparison to pixel based techniques [55, 86, 88, 124, 133, 134]. Different techniques for incorporating spatial information have been proposed in the literature. One of the approaches is to use deep learning based neural networks such as convolutional neural network (CNN) [19, 28, 94, 109, 123, 136, 137] or convolutional autoencoder (CAE) [44, 81, 124]. A complex-valued CNN [136] uses a 12×12 window centered at each pixel for extracting its spatial information. In another work a polarimetric convolutional network was proposed [94] that needed a polarimetric scattering coding to extract features from scattering matrix followed by a CNN to use spatial information. These methods are able to produce better results than pixel based classifiers. But the problem with convolutional neural network based method is that it uses a fixed square shape filter to capture the spatial information. Since the shape of the filter is fixed, it is likely to fail to effectively capture spatial information at boundaries of the image.

Recently, superpixel algorithm in conjunction with deep neural networks has gained popularity for capturing spatial information of a PolSAR image [15, 33, 48, 49, 55, 83, 88, 90]. Superpixel is an over segmentation technique which divides an image into multiple homogeneous regions. One of the methods [55] proposed two layer autoencoder network followed by softmax layer for classification. The result obtained by the autoencoder network was then cleaned using superpixels. In another work a fuzzy superpixel algorithm was proposed [49] for a PolSAR image to clean the classification results. The autoencoder network was also used where superpixel retained error term forced the network to consider homogeneous na-

ture of superpixels [45]. Although these methods are able to achieve good results, training of such networks is found to be a time consuming process. In this chapter, we have proposed superpixel driven optimized Wishart network which combines fast convergence rate of Optimized Wishart Network (OWN) presented in Chapter 2 and spatial information of superpixels to achieve high classification accuracy.

5.2 Superpixel Driven OWN

In literature we find Wishart classifier being widely used for multilook PolSAR image classification which is a pixel based classifier, wherein each pixel is classified independently. It might be a possibility that a pixel is noisy and hence the direct application of this classifier may relate that pixel to some incorrect class. In such cases the classification becomes inaccurate. To solve this problem we suggest that each pixel not only be classified independently but also be seen with respect to a superpixel located around it. To implement this we compute two things: i) RWD of an individual pixel with respect to all cluster centers and ii) RWD of corresponding superpixel mean of each pixel to all cluster centers. Hence, ultimately that pixel is assigned a class label by considering both of these distance values. We have used Pauli decomposition of a PolSAR image as an input to simple linear iterative clustering (SLIC) algorithm [13] to generate superpixels. Figure 5.1 shows an example of superpixel generation on Flevoland15 dataset using SLIC algorithm.



(a) (b)

Figure 5.1: (a) PauliRGB image of Flevoland dataset, (b) superpixel generated using SLIC [13] algorithm. Here, red lines indicates boundaries of different superpixels.

Let $\mathbf{SP}_x$ be the set which contains all PolSAR pixels belonging to the x^{th} super-

pixel. Let $\mathbf{S}_x$ be the mean of the set $\mathbf{SP}_x$. Then

$$\langle \mathbf{S}_x \rangle = \frac{1}{|\mathbf{SP}_x|} \sum_{i=1}^{i=N} I(\langle \mathbf{T}_i \rangle \in \mathbf{SP}_x) \langle \mathbf{T}_i \rangle. \tag{5.1}$$

Here $I(\langle \mathbf{T}_i \rangle \in \mathbf{SP}_x) = 1$ if $\langle \mathbf{T}_i \rangle \in \mathbf{SP}_x$ and 0 if $\langle \mathbf{T}_i \rangle \notin \mathbf{SP}_x$. $|\mathbf{SP}_x|$ is the size of the set $\mathbf{SP}_x$. We explored two primary ways of extracting spatial information. First, we assigned a class label to each pixel based on a cluster center which minimized the summation of these two distances. Let us denote this method as $S{+}P_{(Gk-OWN)}$. Second, we proposed dual branch OWN which would learn to combine pixel and superpixel information with help of these two distances. We denote this method by $SP_{(Gk-OWN)}$. Both methods are explained in the following sections.

5.3 $S{+}P_{(Gk-OWN)}$

To incorporate spatial information for i^{th} pixel of the given PolSAR image we consider an additive distance measure. It uses coherency matrix of i^{th} pixel of the given PolSAR image along with mean coherency matrix $\langle \mathbf{S}_x \rangle$ of x^{th} superpixel. Here it is to be noted that $\langle \mathbf{T}_i \rangle \in \mathbf{SP}_x$. Therefore the additive distance measure corresponding to i^{th} pixel with respect to j^{th} cluster center of class m has two parts as shown in eq. (5.2).

$$\begin{aligned} d(\langle \mathbf{T}_i \rangle, \langle \mathbf{C}_m^j \rangle) = \frac{1}{2} \left[\text{trace}(\langle \mathbf{C}_m^j \rangle^{-1} \langle \mathbf{T}_i \rangle) + \ln \left(\frac{|\langle \mathbf{C}_m^j \rangle|}{|\langle \mathbf{T}_i \rangle|} \right) - n \right] \\ + \frac{1}{2} \left[\text{trace}(\langle \mathbf{C}_m^j \rangle^{-1} \langle \mathbf{S}_x \rangle) + \ln \left(\frac{|\langle \mathbf{C}_m^j \rangle|}{|\langle \mathbf{S}_x \rangle|} \right) - n \right]. \end{aligned} \tag{5.2}$$

Eq. (5.2) can readily be modified as follows:

$$\begin{aligned} d(\langle \mathbf{T}_i \rangle, \langle \mathbf{C}_m^j \rangle) = \frac{1}{2} \left[\text{trace}(\langle \mathbf{C}_m^j \rangle^{-1} \langle \mathbf{T}_i \rangle) + \text{trace}(\langle \mathbf{C}_m^j \rangle^{-1} \langle \mathbf{S}_x \rangle) \right] \\ + \frac{1}{2} \left[\ln \left(\frac{|\langle \mathbf{C}_m^j \rangle|^2}{|\langle \mathbf{T}_i \rangle||\langle \mathbf{S}_x \rangle|} \right) \right] - n. \end{aligned} \tag{5.3}$$

Since we are only interested in trace, first part of eq. (5.3) can be executed efficiently using the function f proposed in Chapter 2. Let $\mathbf{c}_m^j = f(\langle \mathbf{C}_m^j \rangle)$, $\mathbf{t}_i = f(\langle \mathbf{T}_i \rangle)$ and $\mathbf{s}_x = f(\langle \mathbf{S}_x \rangle)$. Using the function f and Algorithm 5 as proposed in

Chapter 4, eq. (5.3) can now mathematically be simplified as follows:

$$\begin{aligned} d(\langle \mathbf{T}_i \rangle, \langle \mathbf{C}_m^j \rangle) = \frac{\mathbf{p}}{2} \odot \left[inv(\mathbf{c}_m^j)^{\mathsf{T}} \mathbf{t}_i + inv(\mathbf{c}_m^j)^{\mathsf{T}} \mathbf{s}_x \right] \\ + \frac{1}{2} \left[\ln \left(\frac{det(\mathbf{c}_m^j)^2}{det(\mathbf{t}_i) det(\mathbf{s}_x)} \right) \right] - n. \end{aligned} \tag{5.4}$$

Here $\mathbf{p} = [1, 2, 2, 2, 2, 1, 2, 2, 1]$ is a constant vector which has to be multiplied to get the correct value of RWD. Eq. (5.4) is required to be solved for all pairs of training samples and cluster means. It can be calculated simultaneously, hence assuring quite low computational overhead.

Let $\mathbf{X}$ and $\mathbf{Z}$ be two matrices in which i^{th} columns of these matrices contain vector representation of i^{th} PolSAR pixel coherency matrix and corresponding superpixel mean coherency matrix. Let $\mathbf{W}$ be a matrix whose columns contain vector representation of inverse of cluster centers. Let $\mathbf{b}$ be a vector where j^{th} element is determinant of j^{th} cluster of class m. Let $\mathbf{d}_t$ and $\mathbf{d}_s$ be two vectors such that i^{th} values in vector $\mathbf{d}_t$ and $\mathbf{d}_s$ are determinants of i^{th} PolSAR pixel and corresponding superpixel mean coherency matrix. Now we can calculate these distances simultaneously by constructing a matrix $\mathbf{D}$, which is elaborated as follows:

$$\begin{aligned} \mathbf{X} &= [\mathbf{t}_1 \odot \mathbf{p}, \mathbf{t}_2 \odot \mathbf{p}, \ldots, \mathbf{t}_K \odot \mathbf{p}], \\ \mathbf{Z} &= [\mathbf{s}_1 \odot \mathbf{p}, \mathbf{s}_2 \odot \mathbf{p}, \ldots, \mathbf{s}_K \odot \mathbf{p}], \\ \mathbf{W} &= [inv(\mathbf{c}_1^1), inv(\mathbf{c}_1^2), \ldots, inv(\mathbf{c}_M^K)], \\ \mathbf{b} &= [det(\mathbf{c}_1^1), det(\mathbf{c}_1^2), \ldots, det(\mathbf{c}_K^M)]^{\mathsf{T}}, \\ \mathbf{d}_t &= [det(\mathbf{t}_1), det(\mathbf{t}_2), \ldots, det(\mathbf{t}_N)]^{\mathsf{T}}, \\ \mathbf{d}_s &= [det(\mathbf{s}_1), det(\mathbf{s}_2), \ldots, det(\mathbf{s}_N)]^{\mathsf{T}}, \\ \mathbf{B} &= \ln(\mathbf{b}^2 / (\mathbf{d}_t \odot \mathbf{d}_s)^{\mathsf{T}}), \\ \mathbf{D} &= \mathbf{W}^{\mathsf{T}} \mathbf{X} + \mathbf{W}^{\mathsf{T}} \mathbf{Z} + \mathbf{B}. \end{aligned} \tag{5.5}$$

All matrices shown on right hand side are known to us. To train $S{+}P_{(Gk-OWN)}$ we have used single hidden layer network which is similar to OWN. During forward pass, input-hidden layer computes the $\mathbf{D}$ matrix shown in eq. (5.5) to get the RWD values at the hidden layer. Then hidden-output layer maps it to the correct class label.

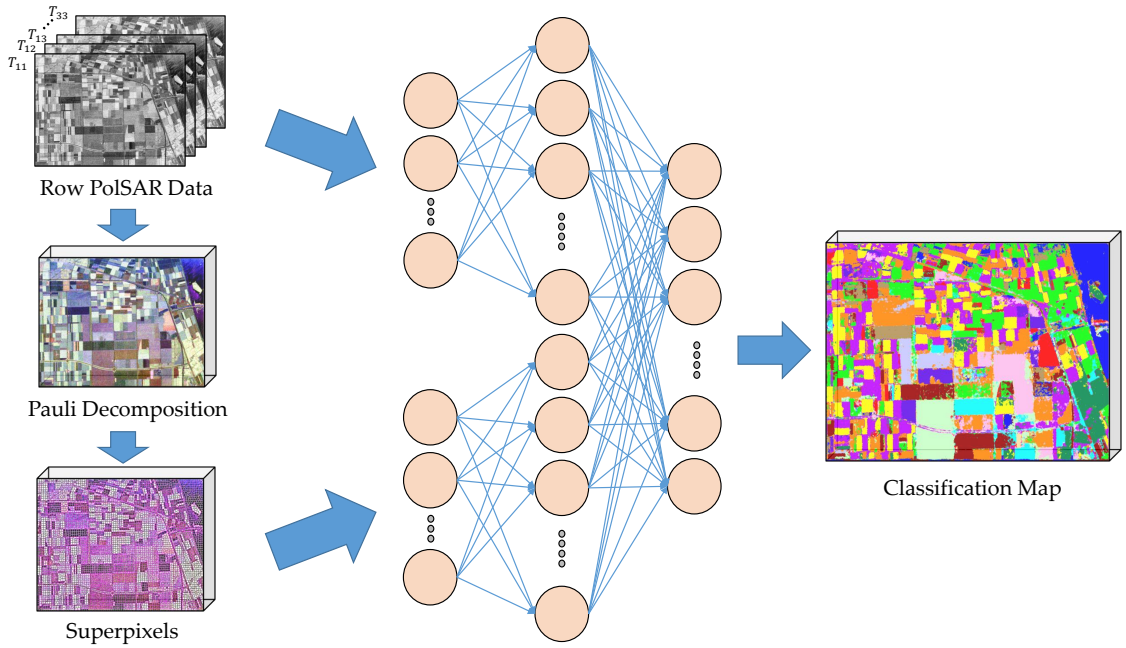


Figure 5.2: Network architecture of proposed $SP_{(Gk-OWN)}$

5.4 $SP_{(Gk-OWN)}$

In the above section we suggested an additive distance measure which used equal influence of both pixel and superpixel information. In this section we discuss dual branch OWN which learns to combine pixel and superpixel information. $SP_{(Gk-OWN)}$ is a single hidden layer dual branch feed forward neural network. It has total of three layers. The input layer contains two branches, one for the PolSAR pixel and other for its corresponding superpixel mean. The weights of this input-hidden layer are initialized using cluster centers obtained by fast global k-means algorithm. Let $\mathbf{W_t}$ and $\mathbf{W_s}$ be the weight matrices associated with pixel and superpixel branch. Hidden-output layer is fully connected layer and its weight is calculated using pseudo-inverse method [63]. Loss function of a proposed network is stated in eq. (5.6).

$$L = \frac{1}{N}\sum_{i=1}^{i=N}\left\|\mathbf{U}^{\mathsf{T}}\begin{bmatrix} tanh(\mathbf{W}_t^{\mathsf{T}}(\mathbf{t}_i \odot \mathbf{p}) + \ln(\mathbf{b}_t/\mathbf{d}_{t_i})) \\ tanh(\mathbf{W}_s^{\mathsf{T}}(\mathbf{s}_i \odot \mathbf{p}) + \ln(\mathbf{b}_s/\mathbf{d}_{s_i})) \end{bmatrix} + \mathbf{c} - \mathbf{y}_i\right\|^2 \tag{5.6}$$

where $\mathbf{W}_t$, $\mathbf{W}_s$, $\mathbf{b}_t$ and $\mathbf{b}_s$ are the parameters of input-hidden layer, $\mathbf{U}$ and $\mathbf{c}$ are the parameters of hidden-output layer, $\mathbf{y}_i$ is a one-hot-encoding representation of true class label of i^{th} training sample and $tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ is an activation function. During forward pass we feed all training samples along with its corresponding superpixel mean to the input layer. Initially we set $\mathbf{W}_t = \mathbf{W}_s = \mathbf{W}$. Since weights

of input-hidden layer are initialized as inverse cluster centers, network calculates two things, i) RWD between every training pixel and every cluster center, ii) RWD between superpixel mean of every training pixel and every cluster center. These two distance vectors are then stacked to get the final hidden layer values. Steps shown in eq. (5.7) illustrate this forward pass mathematically.

$$\begin{cases} \mathbf{H}_t = tanh(\mathbf{W}_t^{\mathsf{T}}\mathbf{X} + \mathbf{B}_t), \\ \mathbf{H}_s = tanh(\mathbf{W}_s^{\mathsf{T}}\mathbf{Z} + \mathbf{B}_s), \\ \mathbf{H} = \begin{bmatrix} \mathbf{H}_t \\ \mathbf{H}_s \\ \mathbf{1} \end{bmatrix}. \end{cases} \tag{5.7}$$

Once the forward pass is completed, total error is calculated by eq. (5.6). We have used gradient decent algorithm to learn parameters of the network. Let δ_{out} and δ_{mid} be gradients for output and hidden layers respectively. It can be computed as follows:

$$\begin{cases} \begin{bmatrix} \mathbf{U} \\ \mathbf{c} \end{bmatrix} = (\mathbf{H}\mathbf{H}^{\mathsf{T}} + \alpha\mathbf{I})^{-\mathbf{1}}(\mathbf{Y}\mathbf{H}^{\mathsf{T}}), \\ \delta_{out} = \mathbf{U}\begin{bmatrix} \mathbf{H}_t \\ \mathbf{H}_s \end{bmatrix} + \mathbf{c} - \mathbf{Y}, \\ \delta_{mid} = \left(1 - \begin{bmatrix} \mathbf{H}_t \\ \mathbf{H}_s \end{bmatrix}^2\right) \odot (\mathbf{U}^{\mathsf{T}}\delta_{out}). \end{cases} \tag{5.8}$$

Here $\odot$ is an element-wise matrix multiplication operation and matrix $\mathbf{Y}$ contains true class labels of training samples in the form of one hot encoding. Input-hidden layer is not fully connected, so the gradients obtained at hidden layer have to be distributed accordingly. Weights of the input-hidden layer can be updated as shown in eq. (5.9).

$$\begin{cases} \mathbf{W}_t^{new} = \mathbf{W}_t^{old} - \lambda\mathbf{X}\delta_{mid}[0 : KM, :]^{\mathsf{T}}, \\ \mathbf{W}_s^{new} = \mathbf{W}_s^{old} - \lambda\mathbf{Z}\delta_{mid}[KM : 2KM, :]^{\mathsf{T}}, \\ \mathbf{b}_t^{new} = det(inv(\mathbf{W}_t^{new})), \\ \mathbf{b}_s^{new} = det(inv(\mathbf{W}_s^{new})), \\ \mathbf{B}_t = \ln(\mathbf{b}_t^{new} \odot \mathbf{d}_t^{\mathsf{T}}), \\ \mathbf{B}_s = \ln(\mathbf{b}_s^{new} \odot \mathbf{d}_s^{\mathsf{T}}). \end{cases} \tag{5.9}$$

Here λ is a learning rate whose value is set between 0 and 1.

5.5 Experiment

This section starts with the details of five datasets that are used for our experiments. Based on the observations of the experiments discussed in Chapter 4, we showed that the proposed fast global k-means algorithm improved effectiveness of the classification. Now we will discuss the classification results using the proposed methods in comparison to other deep learning based methods available in the literature. We will also examine how number of cluster centers belonging to one class, may impact the classification accuracy.

5.5.1 Datasets

Experiments are conducted on the following five different real PolSAR image datasets:

- An L-band PolSAR image of the Flevoland Region in The Netherlands obtained using AIRSAR satellite on 16 Aug, 1989. The ground truth [136] of this image is used for discussion of classification accuracy. It contains 15 different classes, that are, Baresoil, Barley, Beet, Buildings, Forest, Grasses, Lucerne, Peas, Potatoes, Rapeseed, Steambeans, Water, Wheat 1, Wheat 2 and Wheat 3. This image is of 750×1024 size. Let us call it Flevoland15 dataset.

- An L-band PolSAR image of the Landes Region in France obtained by AIRSAR satellite on 19 June, 1991. The ground truth [22] of this image is used for discussion of classification accuracy. It comprises of 6 classes of trees of differing ages. It is of 1050×1000 size.

- An L-band PolSAR image of Flevoland region in The Netherlands obtained by AIRSAR satellite on 16 June, 1991. Its ground truth [22] is used for discussion of classification accuracy. It has 7 different classes, such as, Barley, Beet, Lucerne, Peas, Potato, Rapeseed and Wheat. Its size is 750×700. Let us call it Flevoland7 dataset.

- An L-band PolSAR image of Oberpfaffenhofen Region in Germany obtained by E-SAR satellite. The corresponding ground truth [136] of this image is used for discussion of classification accuracy. It contains 3 different terrain classes, Open Area, Built-up Area and Wood Land. It has a size 1300×1200.

Table 5.1: List of datasets used for the experiments

Name	Radar	Band	Year	Resolution	Size	Classes
Flevoland15	AIRSAR	L	1989	10×10m	750×1024	15
Landes	AIRSAR	L	1991	10×10m	1050×1000	6
Flevoland7	AIRSAR	L	1991	10×10m	750×700	7
Oberpfaffenhofen	E-SAR	L	1991	3×2.2m	1300×1200	3
San Francisco	RADARSAT-2	C	2008	10×5m	1800×1380	5

- An C-band PolSAR image of the San Francisco Bay Area in USA obtained using RADARSAT-2 satellite in 2008. It has 5 different classes, namely, Vegetation, Developed, Ocean, Low-density Urban and High-density Urban. Its size is 1800×1380.

Table 5.1 summarizes the details of dataset used for the experiments. It is to note that the first three datasets involve classes of almost similar pattern, mostly dealing with different types of crops. One may say that the surface scattering properties of these classes are more or less alike. So an algorithm working effectively on such classes may behave poorly on a complex terrain. Therefore, dataset (iv) and (v) are selected as examples of complex terrain. Results of the experiments corresponding to each dataset are reported in the following sections.

5.5.2 Results on Flevoland15 Dataset

Flevoland15 dataset is considered to be the benchmark dataset in the field of PolSAR image classification. To evaluate performance of the proposed algorithms we compared the classification accuracy obtained by our method with 9 other algorithms, namely, Wishart classifier (WC) [78], optimized Wishart network (OWN) [41], support vector machine (SVM) [75], deep sparse filtering network (DSFN) [88], Wishart deep stacking network (W-DSN) [63], superpixel retained deep neural network (SRDNN) [45], polarimetric convolutional network [94], Wishart deep belief network [86] and complex-valued CNN [136].

Comparison of classification map obtained by each method is shown in Figures 5.3(a)-(l). The class-wise accuracies are listed in Table 5.2. Results of our proposed algorithms are shown in the last three rows of the Table 5.2. The best results are shown in bold. One can readily see that except for one class, i.e., Wheat 2, our proposed algorithms are showing best results. This leads to a significant improvement in overall accuracy for this dataset.

WC, OWN, SVM and W-DSN are pixel-wise classification methods which do

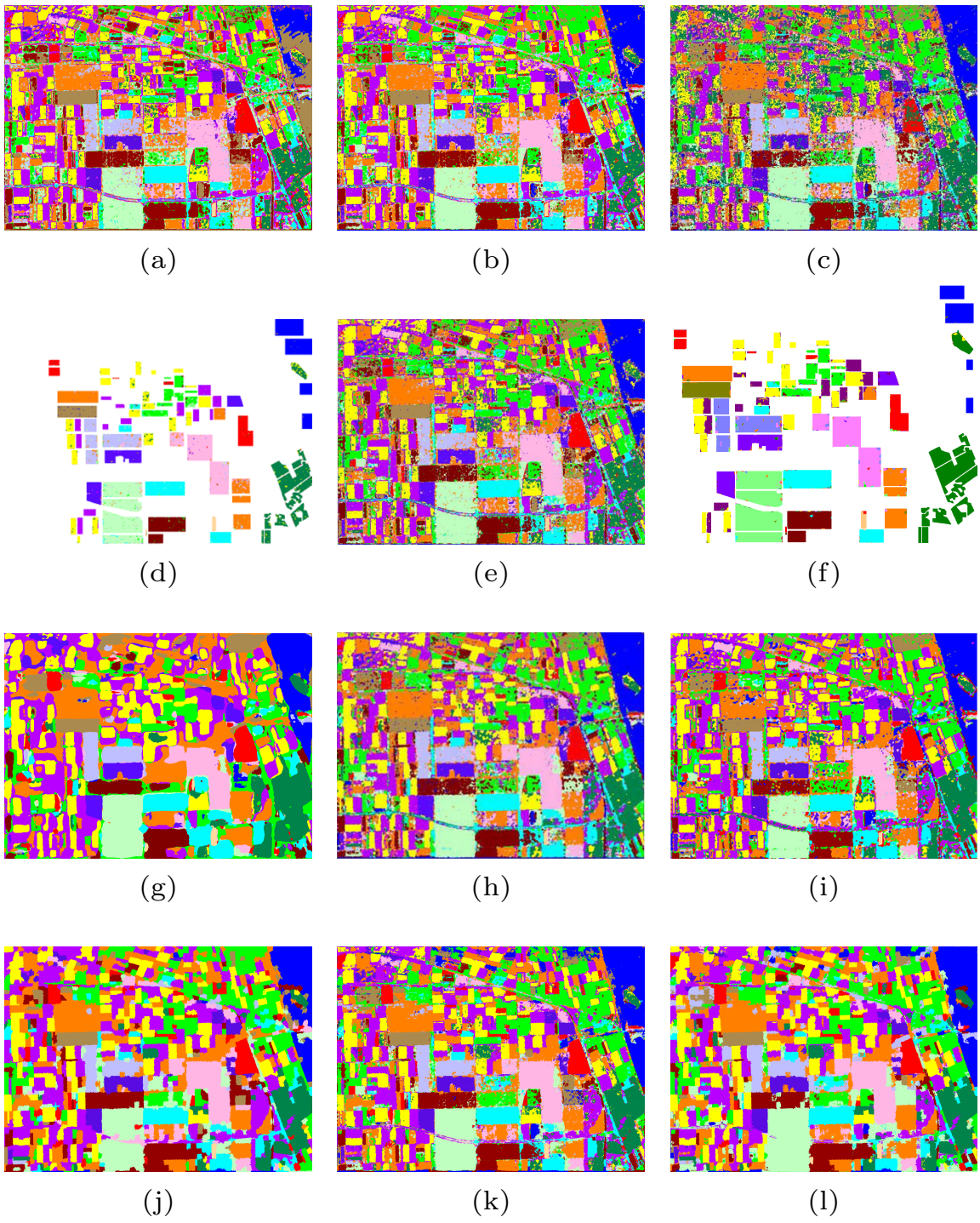


Figure 5.3: Classification map obtained using different methods. (a) Wishart classifier [78], (b) Optimized Wishart classifier [41], (c) SVM classifier [75], (d) Deep sparse filtering network [88], (e) Wishart deep stacking network [63], (f) Superpixel restrained DNN [45], (g) Polarimetric CNN [94], (h) Deep belief network [86], (i) complex-valued CNN [136], (j) $S_{(GK-OWN)}$ where superpixels were used during post-processing to clean the results, (k) $S{+}P_{(Gk-OWN)}$ which uses additive distance measure to combine pixel and superpixel information, (l) $SP_{(Gk-OWN)}$ which uses dual branch OWN to efficiently combine pixel and superpixel information. note that (j), (k) and (l) are results of our proposed methods of this chapter.

Table 5.2: Class-wise Classification Accuracy Comparison on Flevoland15 dataset

Method/Class	Stembeans	Peas	Forest	Lucerne	Wheat 1	Beet	Potatoes	Baresoil
WC [78]	96.15	96.51	90.14	91.74	85.53	94.73	86.89	98.77
OWN [41]	96.38	95.93	90.77	96.34	93.00	95.80	89.62	98.72
SVM [75]	63.55	78.94	93.82	96.04	90.79	90.91	53.97	96.58
DSFN [88]	97.19	92.15	97.37	94.05	97.98	90.05	96.78	94.68
W-DSN [63]	96.00	96.81	90.85	96.05	93.22	96.22	87.63	99.26
SRDNN [45]	97.08	94.52	97.31	95.55	95.52	92.38	94.19	93.92
PCN [94]	96.44	95.29	95.00	93.51	95.20	97.59	95.89	96.45
W-DBN [86]	96.71	98.68	96.45	98.47	97.67	98.14	98.08	97.34
CV-CNN [136]	98.80	98.70	96.80	98.10	95.00	97.60	96.70	98.80
$S_{(Gk-OWN)}$	96.74	95.98	98.29	97.83	98.70	97.55	99.12	**99.71**
$S{+}P_{(Gk-OWN)}$	98.93	99.31	98.00	99.45	98.02	**99.49**	97.67	99.59
$SP_{(Gk-OWN)}$	**99.71**	**99.32**	**99.73**	**99.58**	**99.09**	98.77	**99.38**	99.69
Method/Class	Grasses	Rapeseed	Barley	Wheat 2	Wheat 3	Water	Buildings	OA
WC [78]	74.37	73.14	94.10	85.31	90.23	47.98	82.35	84.71
OWN [41]	80.52	83.34	98.20	81.13	95.68	98.97	77.88	92.24
SVM [75]	87.59	81.28	97.17	81.15	95.48	97.82	43.00	85.87
DSFN [88]	96.53	97.06	93.06	89.11	92.40	97.53	95.29	94.86
W-DSN [63]	90.13	86.18	97.05	81.20	95.79	98.88	84.12	92.68
SRDNN [45]	87.26	91.81	97.54	89.65	97.60	99.53	81.74	94.66
PCN [94]	94.33	93.98	94.51	95.03	95.43	99.50	95.58	96.94
W-DBN [86]	95.39	95.90	99.49	94.79	98.55	99.90	88.56	97.57
CV-CNN [136]	90.00	92.00	94.50	**97.20**	95.60	98.50	80.00	97.70
$S_{(Gk-OWN)}$	98.17	87.95	**100**	92.33	97.05	96.94	82.35	96.68
$S{+}P_{(Gk-OWN)}$	97.77	93.62	99.60	94.33	99.65	99.98	92.05	98.13
$SP_{(Gk-OWN)}$	**99.78**	**97.13**	**100**	96.28	**99.92**	**99.97**	**99.34**	**99.15**

Table 5.3: Time Analysis

Methods	Preprocessing	Training	Testing	Total
SRDNN	-	721.15	6.89	728.04
W-DBN	-	103.16	92.21	195.37
PCN	-	341	35	376
W-DSN	-	41.48	8.28	49.77
$S_{(Gk-OWN)}$	1.43s	12.47s	2.68s	16.58s
$S+P_{(Gk-OWN)}$	1.43s	14.18s	3.21s	18.82s
$SP_{(Gk-OWN)}$	1.43s	18.83s	4.17s	24.43s

not use spatial information. Therefore the overall accuracies of these methods remains generally low. Overall accuracies of WC, OWN, SVM and W-DSN are 84.71%, 92.24%, 85,87% and 92.68% respectively. DSFN, SRDNN, PCN, W-DBN and CV-CNN use spatial information due to which their accuracies are better in comparison to pixel-based methods. DSFN and SRDNN use superpixels to include spatial information. Their overall classification accuracies are 94.86% and 94.66% respectively. On the other hand, PCN and CV-CNN are based on CNN architecture to exploit spatial information. Their overall classification accuracies are 96.94% and 97.70%. W-DBN does not use spatial information while training, but it exploits cleaning algorithm in post-processing stage to correct poorly labeled pixels. Its overall accuracy is 97.57%.

As we can see, in the case of WC (a pixel-based method) an inaccurate classification between Water and Baresoil is reported due to almost similar nature of scattering of both the terrains. In case of OWN and W-DSN there is a trace of misclassification between Forest and Potato. In case of SVM there is a high misclassification in almost all classes. In case of DSFN (a spatial information based method) there is a confusion between Forest and Potato similar to OWN and W-DSN. Also there is a minor confusion between Wheat 1 and Rapeseed. From the classification map of PCN shown in the Figure 5.3(g), we can observe that it over smooths the boundary region which may lead to misclassification of boundary pixels.

One may also use superpixel information simply as a post-processing step to clean the result of classification like it is done in W-DBN. To achieve this we cleaned the result obtained by Gk-OWN using superpixels. Let us denote this method as $S_{(Gk-OWN)}$. Figure 5.3(j) shows classification map obtained by $S_{(Gk-OWN)}$. It uses majority voting to clean the result of the classification, so the pixels belonging to the same superpixel acquire the same class label. A disadvantage of

this approach is that misclassification may occur if the superpixel itself is poorly formed.

Figure 5.3(k) shows classification map obtained by $S{+}P_{(Gk-OWN)}$. In case of $S{+}P_{(Gk-OWN)}$, we first combine pixel and superpixel information using additive distance measure and then present it to the network. Since it uses both pixel and superpixel information, performance of $S{+}P_{(Gk-OWN)}$ is better in comparison to $S_{(Gk-OWN)}$.

Figure 5.3(l) shows the classification map obtained by $SP_{(Gk-OWN)}$ which by definition takes dual branch OWN to combine pixel and superpixel information. Since the information of pixels and superpixels are presented separately in dual branch architecture, network learns to combine them efficiently. This can readily seen in the performance of $SP_{(Gk-OWN)}$ in comparison to both $S_{(Gk-OWN)}$ and $S{+}P_{(Gk-OWN)}$. It gives best results in 12 out of the 15 classes. $SP_{(Gk-OWN)}$ achieves 99.15% overall accuracy and outperforms all 9 methods by 14.44%, 6.91%, 13.28%, 4.29%, 6.47%, 4.49%, 2.21%, 1.58% and 1.45% respectively.

For computational overhead calculation we next compared the time taken by SRDNN, W-DBN, PCN, W-DSN, $S_{(Gk-OWN)}$, $S{+}P_{(Gk-OWN)}$ and $SP_{(Gk-OWN)}$ methods to classify Flevoland15 dataset. We measured three main time requirements, e.g., preprocessing time, training time and testing time. SRDNN, W-DBN, PCN and W-DSN do not employ any preprocessing. In case of our proposed methods, preprocessing time includes time required to generate the superpixels. The training time includes both fast global k-means run time as well as the time required to train the network using 5% training samples. The testing time includes the time required to test remaining 95% samples. Table 5.3 shows time required by all methods. $SP_{(Gk-OWN)}$ requires highest training and testing time among all three proposed methods due to its dual branch architecture. Due to their deep neural network based architectures SRDNN, W-DBN, PCN and W-DSN require higher training and testing time in comparison with proposed approaches.

5.5.3 Results on Oberpfaffenhofen dataset

Oberpfaffenhofen dataset [12] is a complex urban terrain containing three classes, i.e., Open Area, Built-up Area and Wood Land. The ground truth for this dataset was obtained from [136]. Figure 5.4(a) and (b) show Pauli decomposition of this dataset and its ground truth map. One can see that classes of this dataset are more heterogeneous than the previous datasets.

Figure 5.5(a)-(f) show classification map obtained by CV-CNN [136] and SRDNN [45] besides our proposed algorithms of Gk-OWN, $S_{(Gk-OWN)}$, $S{+}P_{(Gk-OWN)}$ and

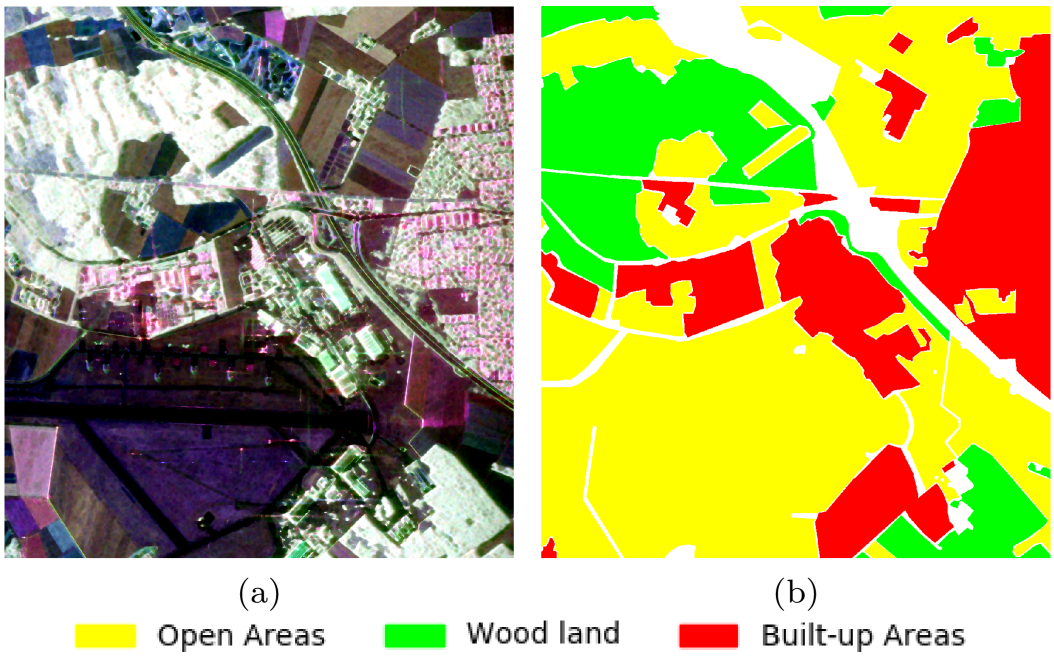


Figure 5.4: (a) PauliRGB image of Oberpfaffenhofen dataset. (b) Ground truth of Oberpfaffenhofen dataset [136]. The three classes of this dataset are represented by three colours.

$SP_{(Gk-OWN)}$. Table 5.4 shows class-wise accuracies obtained by all six methods for the Oberpfaffenhofen dataset. Gk-OWN achieves the lowest classification accuracy because it is a pixel-wise classifier and does not use spatial information. As we can see, the performance of Gk-OWN improves after incorporating spatial information. In case of $S_{(Gk-OWN)}$ we observe 3.28% improvement in comparison to Gk-OWN. Majority of this improvement came from Build-up Area class. In case of $S + P_{(Gk-OWN)}$ and $SP_{(Gk-OWN)}$, the accuracy of Built-up Areas and Wood land improved significantly. In the Open Areas class, the added spatial information did not lead to much improvement, because it was well classified by Gk-OWN itself due to less complexity in the terrain type. It is evident from Table 5.4 that the proposed SP_{Gk-OWN} achieves highest overall accuracy of 95.09% among all five methods.

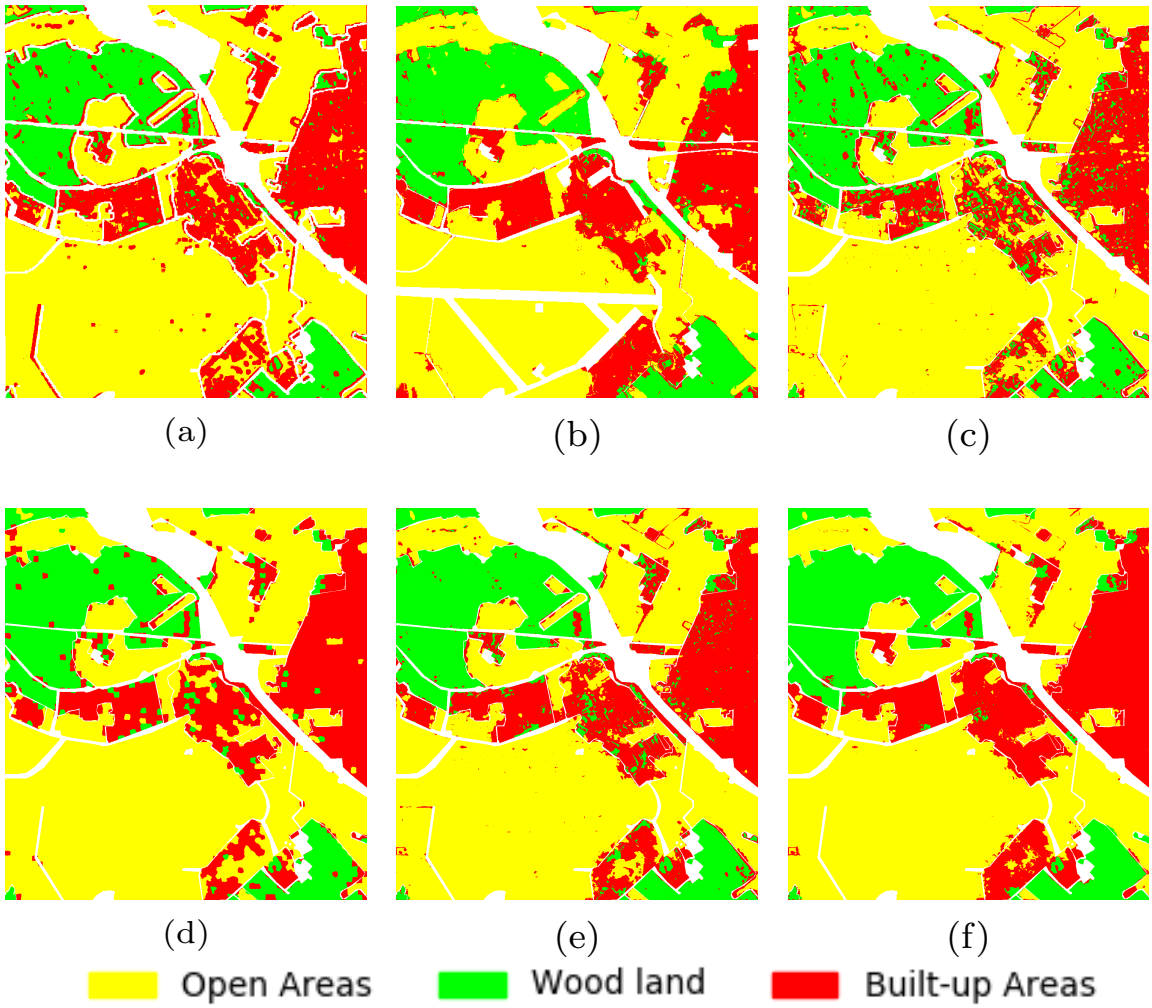


Figure 5.5: Classification map of Oberpfaffenhofen dataset obtained using (a) CV-CNN [136] (b) SRDNN [45], (c) Gk-OWN, (d) $S_{(Gk-OWN)}$, (e) $S{+}P_{(Gk-OWN)}$ and (f) $SP_{(Gk-OWN)}$.

Table 5.4: Class-wise Classification Accuracy Comparison on Oberpfaffenhofen dataset

Methods	Open Areas	Built-up Areas	Wood Land	OA	AA
Samples	766504	339784	268010	-	-
CV-CNN [136]	94.60	91.30	92.20	93.40	92.70
SRDNN [45]	93.92	89.90	94.53	93.09	92.78
Gk-OWN	96.28	71.83	86.08	88.47	84.71
$S_{(Gk-OWN)}$	**97.59**	80.83	88.87	91.75	89.09
$S{+}P_{(Gk-OWN)}$	95.67	85.82	93.04	92.72	91.51
$SP_{(Gk-OWN)}$	96.37	**92.81**	**94.30**	**95.09**	**94.49**

5.5.4 Results on San Francisco Bay Area dataset

The San Francisco dataset [93] is of size 1800×1380. It contains 2484000 total pixels out of which 1804087 are labeled. Due to large number of labeled pixels we used only 1% of them as training data and remaining 99% as testing data. Table 5.5 shows class-wise classification accuracy obtained by OWN, Gk-OWN, S_{Gk-OWN}, $S + P_{Gk-OWN}$ and SP_{Gk-OWN}. Here, C1-C5 indicate five classes of San Francisco dataset, namely, Developed, Water, Vegetation, High-Density Urban and Low-Density Urban respectively. Figure 5.6 shows corresponding classification maps.

As we can see, performance of Gk-OWN improves significantly in four out of five classes after incorporating spatial information. In case of Water class performance of Gk-OWN itself is excellent due to simplicity of terrain type. The Developed class is worst classified by the Gk-OWN. The Developed class contains residential buildings which are not aligned to radar line of sight. Because of that their back scattering appears similar to that of Vegetation. Same is evident from its Pauli decomposition image. Because of this Developed class, in most cases, was misclassified as Vegetation class by Gk-OWN. Incorporating the spatial information in Gk-OWN significantly improved the accuracy of Developed class. Same effect was observed in the case of High Density Urban and Low-Density Urban classes. They were misclassified as each other due to similar nature of scattering from both classes. The Vegetation class is more complex in terrain type compared to other four classes. It mainly consists of parks containing trees, roads, buildings, plain field etc. Because of this, it was misclassified as Developed or Low Density Urban class.

Table 5.5: Class-wise accuracies of San Francisco dataset

Methods	C1	C2	C3	C4	C5	OA	AA
Samples	80616	852078	237237	282975	351181	-	-
OWN	61.21	99.95	87.75	68.98	78.13	87.63	79.36
Gk-OWN	63.25	**99.99**	88.04	72.51	78.06	88.18	80.36
S_{Gk-OWN}	81.58	99.98	**93.89**	91.77	**97.48**	96.68	93.04
$S + P_{Gk-OWN}$	89.39	99.98	90.30	92.05	91.68	95.37	92.68
SP_{Gk-OWN}	**96.94**	**99.99**	92.42	**97.10**	96.97	**97.75**	**96.66**

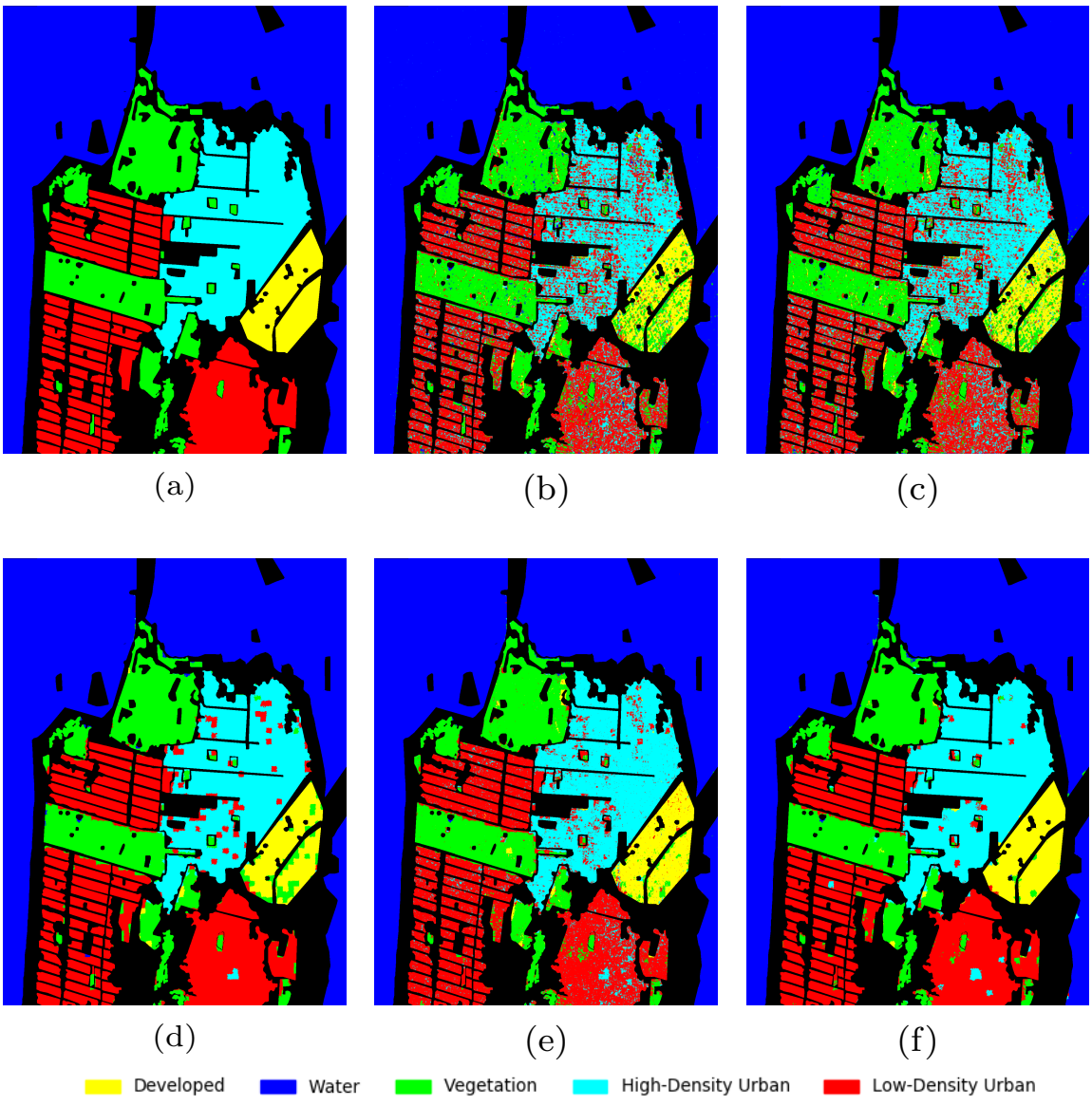


Figure 5.6: (a) Ground truth map of San Francisco dataset. Classification map obtained using (b) OWN, (c) Gk-OWN, (d) $S_{(Gk-OWN)}$, (e) $S + P_{(Gk-OWN)}$ and (f) $SP_{(Gk-OWN)}$

5.5.5 Results on Landes and Flevoland7 Dataset

Figure 5.8(a)-(e) show classification map of Landes dataset and Figure 5.8(f)-(j) of Flevoland7 obtained by WC [78], Gk-OWN, $S_{(Gk-OWN)}$, $S{+}P_{(Gk-OWN)}$ and $SP_{(Gk-OWN)}$. WC and Gk-OWN showed misclassification in both the datasets due to the presence of noisy pixels in the datasets. In case of $S_{(Gk-OWN)}$, most of the misclassification occurred due to badly formed superpixels. $S{+}P_{(Gk-OWN)}$ was able to achieve higher classification accuracy because it used both pixel and superpixel information. Best overall accuracy was obtained by $SP_{(Gk-OWN)}$ approach. Due to its dual branch architecture, $SP_{(Gk-OWN)}$ was able to combine both pixel and superpixel information efficiently. Table 5.6 and 5.7 show class-wise accuracies obtained by all five methods on Flevoland7 and Landes datasets.

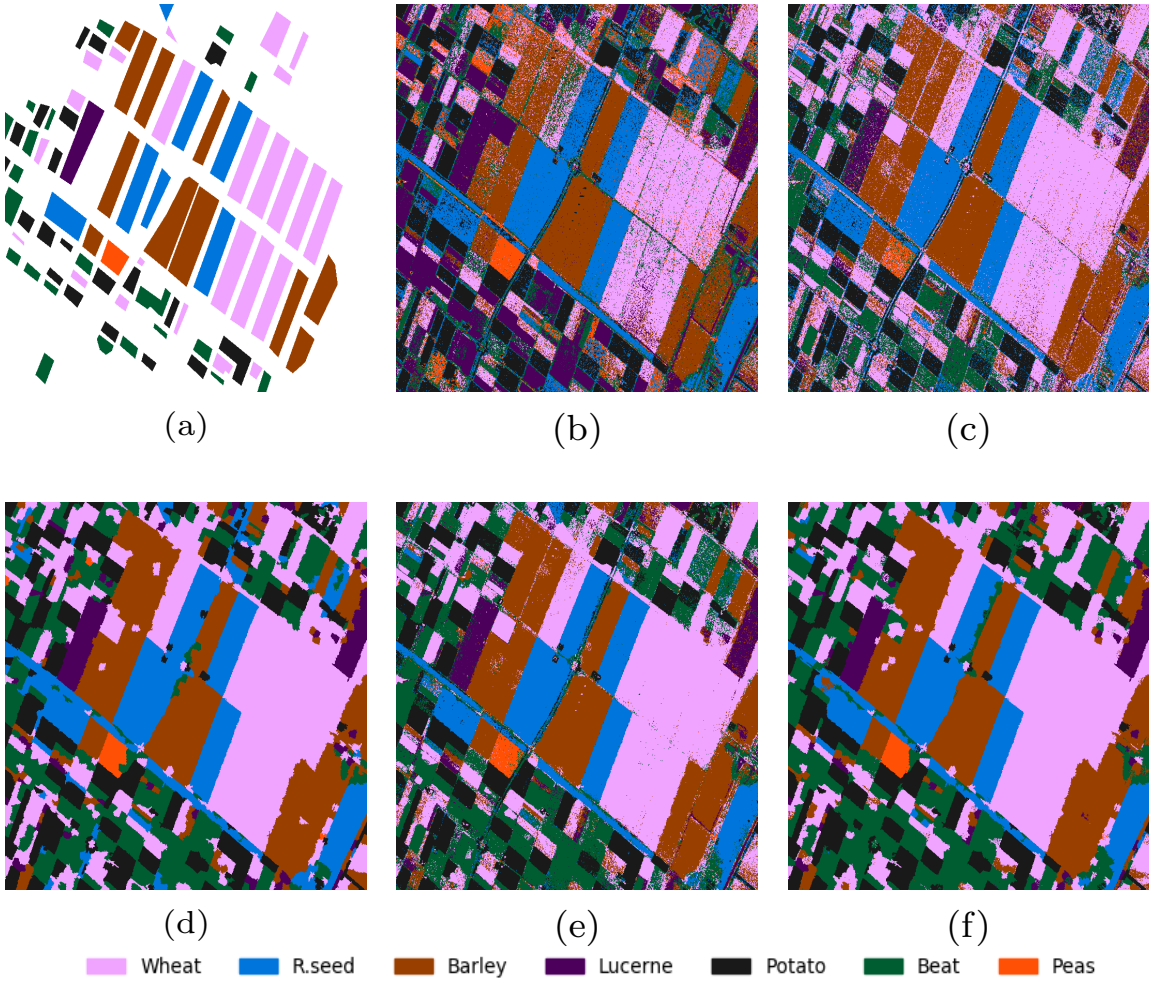


Figure 5.7: (a) Ground truth map of Flevoland7 dataset. Classification map obtained using (b) WC [78], (c) Gk-OWN, (d) $S_{(Gk-OWN)}$, (e) $S{+}P_{(Gk-OWN)}$, (f) $SP_{(Gk-OWN)}$.

Table 5.6: Class-wise Classification Accuracy Comparison on FLevoland7 dataset

Methods	Wheat	R.seed	Barley	Lucerne	Potato	Beet	Peas	OA
Samples	56060	25994	45497	4935	17035	14974	2055	-
WC	82.13	92.29	89.52	97.97	94.59	20.53	97.37	82.13
Gk-OWN	94.48	95.36	95.94	76.53	92.82	74.78	67.54	92.21
$S_{(Gk-OWN)}$	97.93	99.49	98.48	**100**	99.13	94.96	93.77	98.19
$S{+}P_{(Gk-OWN)}$	98.30	99.54	99.34	95.52	98.93	93.21	94.45	98.25
$SP_{(Gk-OWN)}$	**99.04**	**99.90**	**99.71**	99.35	**99.45**	**98.88**	**97.86**	**99.38**

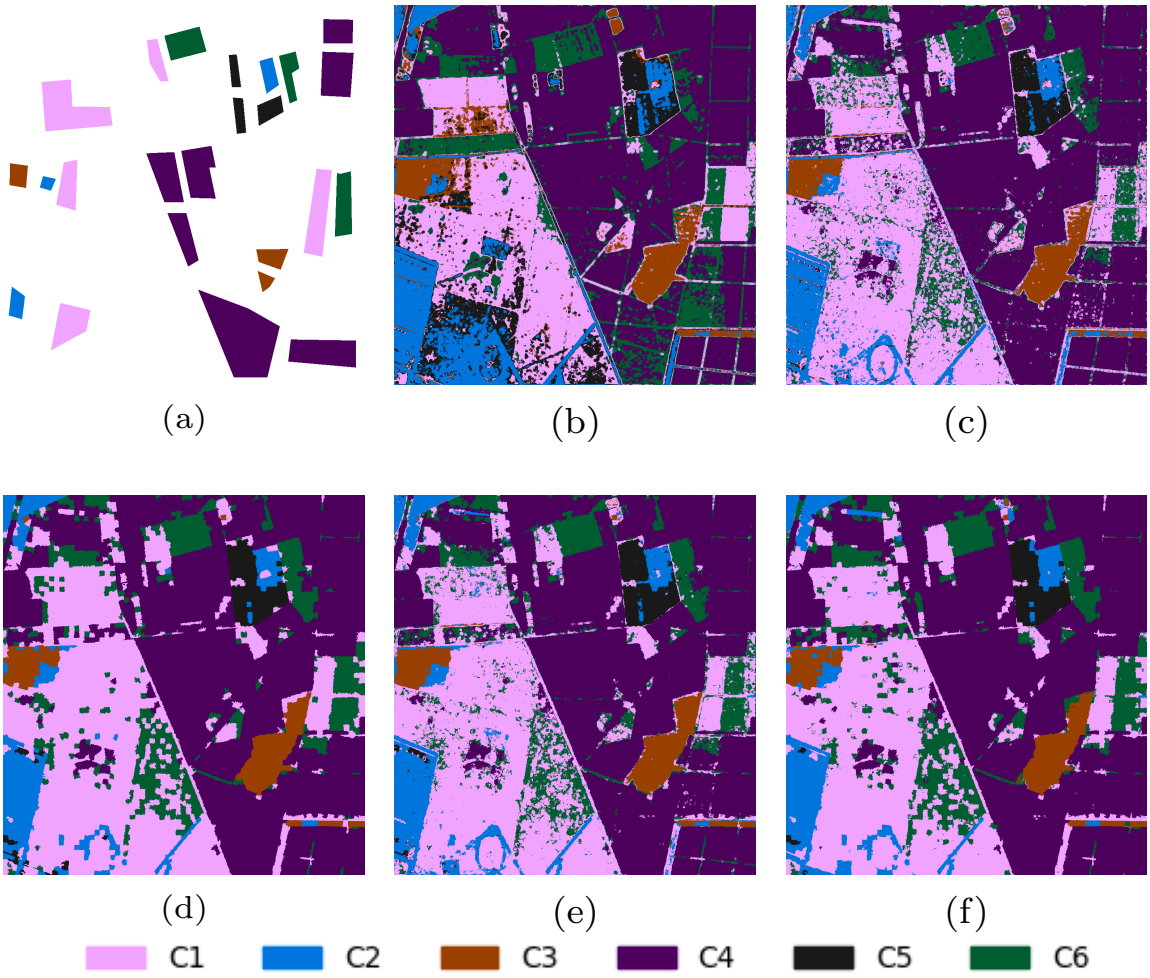


Figure 5.8: (a) Ground truth map of Landes dataset. Classification map obtained using (b) WC [78] (c) Gk-OWN, (d) $S_{(Gk-OWN)}$, (e) $S{+}P_{(Gk-OWN)}$, (f) $SP_{(Gk-OWN)}$.

Table 5.7: Class-wise Classification Accuracy Comparison on Landes dataset

Methods	C1	C2	C3	C4	C5	C6	OA
Samples	46079	7182	7874	83692	8410	18963	-
WC	35.13	84.13	99.58	84.01	81.63	39.23	66.60
Gk-OWN	84.96	92.43	99.77	97.41	95.90	90.11	93.10
$S_{(Gk-OWN)}$	91.08	93.04	**100**	**99.70**	99.06	0.959	96.68
$S{+}P_{(Gk-OWN)}$	90.66	96.81	**100**	99.22	99.41	95.74	96.49
$SP_{(Gk-OWN)}$	**95.99**	**99.40**	**100**	99.57	**99.63**	**98.68**	**98.53**

5.5.6 Effect of number of cluster centers (K) on classification

One of the important parameters of the proposed network is the value of K. It determines how many cluster centers we are selecting from each class. Goa *et al*. [43] showed that overall accuracy of classification improves while increasing value of K upto 10. To determine the influence of K, we conducted an experiment on the three datasets namely, Flevoland15, Landes and San Francisco. We changed the value of K from 1 to 10 and measured overall classification accuracies and total time required in training and testing of the network.

Figures 5.9 show effect of value of K on overall accuracy and classification time. As we can see in the case of Flevoland15 dataset we achieved 93.75% accuracy for $K = 1$. It increased upto 98.52% for $K = 5$. After that the improvement slowed down. The classification time for $K = 1$ was observed to be 12.3s which linearly increased with value of K. For $K = 10$, the classification time was 50.6s. We can observe similar results for Landes and Flevoland7 dataset. In case of Landes dataset, Figure 5.9(c), we obtained 86.27% for $K = 1$ which rose to 97.28% for $K = 5$. Similarly, classification time shown in Figure 5.9(d) linearly increased from 15.1s to 32.4s. In case of Landes dataset the slope of the line representing the classification time is lower compared to Flevoland15 dataset. Because the Landes dataset have less number of classes compared to Flevoland15 dataset while the size of both datasets are relatively similar. The small number of classes leads to smaller size of hidden layer and output layer and in turn reduces classification time. In case of San Francisco dataset we started with 92.54% accuracy for $K = 1$ which improved to 96.88% for $K = 5$. The classification time for $K = 1$ was 38.33s which is the highest in comparison with other two dataset. The San Francisco dataset has only five classes but the size of it is very large which leads to higher training and testing time.

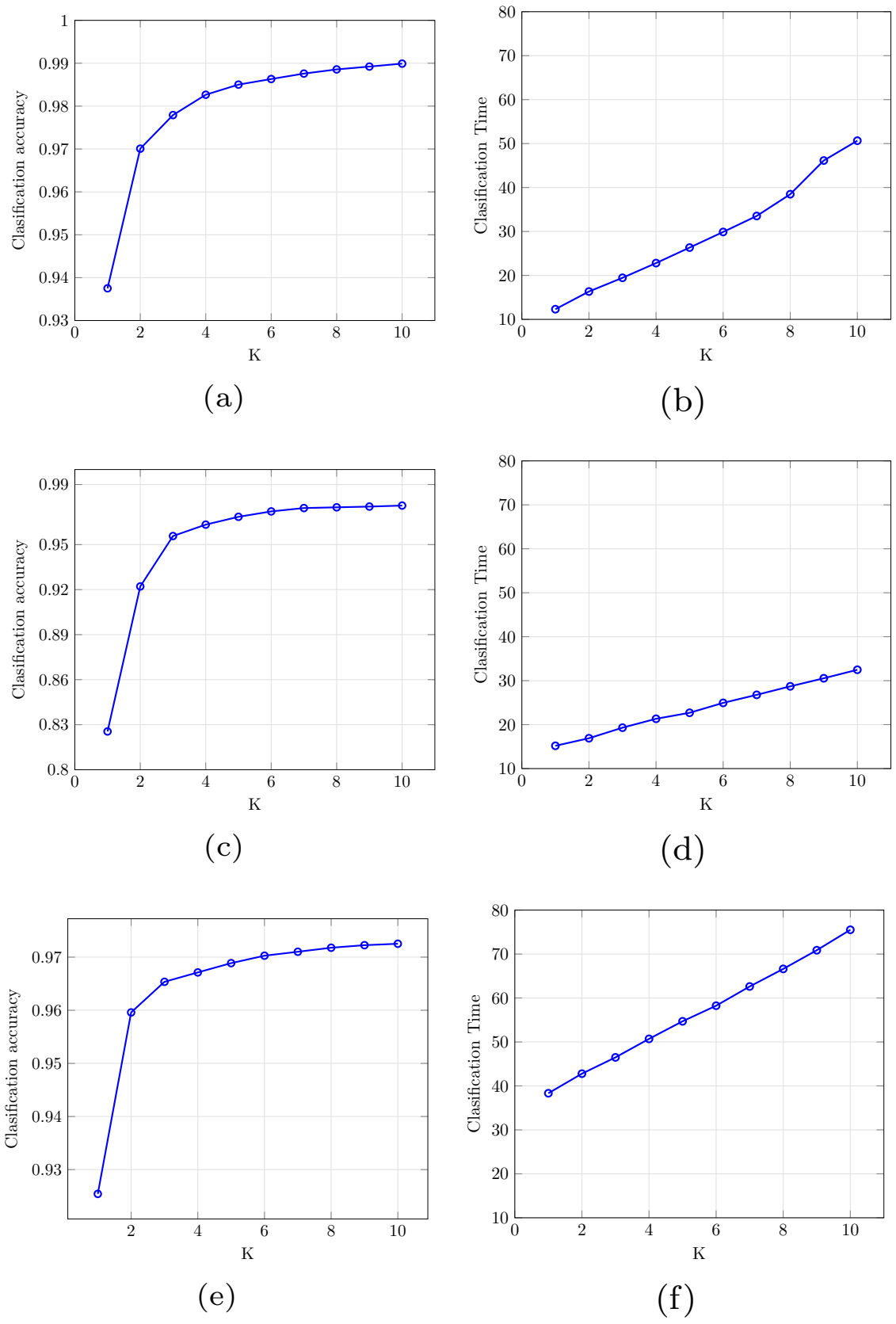


Figure 5.9: Classification accuracies and time taken with different value of K for (a) Flevoland15 dataset containing 15 classes, (c) Landes dataset containing 6 classes and (e) San Francisco dataset containing 5 classes. Classification time required different value of K for (b) Flevoland15 dataset, (d) Landes dataset and (f) San Francisco dataset.

5.6 Conclusion

In this chapter we proposed a superpixel driven optimized Wishart network for PolSAR image classification. In chapter 4, we showed that the application of fast global k-means algorithm successfully gave improved classification accuracy. Encouraged by that result first we used the fast global k-means algorithm for PolSAR data to obtain optimal cluster centers. Having done so we also addressed the possibility of a common problem that the pixel based classifier might fail to correctly classify a noisy pixel. To solve this problem we exploited spatial information of all individual pixels by using a superpixel algorithm. For this we computed two distance measures, (i) revised Wishart distance of a pixel from all cluster centers and (ii) revised Wishart distance of the corresponding superpixel to all cluster centers. Then we designed a novel network which automatically decided how to exploit these two distance measures for better classification of each pixel. We primarily explored two ways of incorporating spatial information, i.e, (i) by using additive distance measure of both distances named as $S{+}P_{(Gk-OWN)}$ and (ii) by using dual branch OWN and called it $SP_{(Gk-OWN)}$. We concluded that the proposed network can efficiently combine pixel and superpixel information to achieve high classification accuracy. We also observed that $SP_{(Gk-OWN)}$ exhibited the best overall accuracy for all the datasets used in the experiment.

CHAPTER 6
Stacked Autoencoder Based Multifrequency PolSAR Image Classification

6.1 Introduction

Elements of coherency matrix of a PolSAR pixel are widely used as input to the deep learning architecture. It is known that the elements of a coherency matrix are difficult to relate with the physical scattering behavior of an object, due to which deep learning architecture may not be able to extract polarimetric features [19]. Polarimetric features are features that have physical interpenetration. It has been shown that polarimetric features extracted from the coherency matrix produce better classification results than merely by the raw elements of coherency matrix [19, 33, 56, 119]. The goal of Target Decomposition (TD) theorem is to extract polarimetric features that contain physical scattering information of the observed scattering matrix or coherency matrix. It can be achieved by decomposing the scattering matrix or coherency matrix as a combination of elementary scattering surfaces such as single, double, volume or helix scattering, etc. Each of these scattering surfaces provides an intuitive interpretation of the underlying object. For example, the single bounce is observed from a flat surface, such as open field or water body. Double bounce is typically observed from dihedral structures such as buildings or trunk of a tree. Volume bounce occurs from complex structures such as branches of trees or crops. There is no single TD algorithm that can accurately estimate the contribution of various scattering surfaces in a pixel of a PolSAR image [33]. So, to obtain full polarimetric information from a PolSAR pixel, we can apply multiple TD methods and stack their results to achieve a high-dimensional polarimetric feature vector [33, 119].

In this chapter, we present an autoencoder based deep neural network for multifrequency PolSAR image classification. The proposed network consists of two modules of an autoencoder and a softmax classifier stacked together. First, we

use the five most popular TD methods and elements of the coherency matrix to extract polarimetric features from all bands of the multifrequency PolSAR data. These features are then stacked to get the high-dimensional input feature vector. Since this feature vector contains some redundant information, we apply the first module of the stacked autoencoder to reduce the feature vector's dimensionality. Then, we use this reduced dimensional feature vector to generate superpixels. After that, the second module of the network combines pixel and the corresponding superpixel information to construct a robust feature vector. Finally, the last module of the proposed network performs the classification task using a softmax classifier.

6.2 Proposed Methodology

With this little background of parameter extraction based on coherent and incoherent decomposition of a PolSAR image, we used 5 decomposition methods to get 33 polarimetric features from one frequency band of a PolSAR image. Hence, combining information of all three bands we get 99 dimensional feature vector corresponding to each PolSAR pixel. Table 6.1 lists all 33 features. Each TD algorithm tries to estimate the scattering contribution of various scatterers in a different way. So, the constructed polarimetric feature vector contains some redundant information. Figure 6.1 shows our proposed network architecture consisting of

Table 6.1: List of extracted features

Features	Description
$\lvert T_{13}\rvert/\sqrt{T_{11}T_{33}}$, $\lvert T_{23}\rvert/\sqrt{T_{33}T_{22}}$, T_{22}/S, T_{33}/S, $10log_{10}(S)$, $\lvert T_{12}\rvert/\sqrt{T_{11}T_{22}}$	6 features of Coherency matrix [137]
F_{dbl}, F_{odd}, F_{vol}	3 features of Freeman decomposition [38]
K_d, K_h, K_s, K_t	4 features of Krogager decomposition [73]
P_{dbl}, P_{hlx}, P_{odd}, P_{vol}	4 features of Yamaguchi decomposition [126]
A, B_0, B, C, D, E, F, G, H	9 features of Huynen decomposition [59]
α, β, δ, γ, λ, Entropy, Anisotropy	7 features of Cloude decomposition [20]

three modules. The first module is a two layer autoencoder network. The purpose of this module is to reduce the dimensionality of the input feature vector by learning efficient representation of combined PolSAR frequency bands information.

Let $\mathbf{X}_i$ be the input feature vector of i^{th} training sample. Let $\mathbf{W}_{11}$ and $\mathbf{W}_{12}$ be the weights and $\mathbf{b}_{11}$ and $\mathbf{b}_{12}$ be the biases of the two encoder layers of first module. A hidden representation of input feature vector $\mathbf{X}_i$ can be calculated as:

$$\mathbf{h}_i = g\Big(\mathbf{W}_{12}^T g\Big(\mathbf{W}_{11}^T \mathbf{X}_i + \mathbf{b}_{11}\Big) + \mathbf{b}_{12}\Big), \tag{6.1}$$

where g is a *ReLU* activation function. Let $\mathbf{W}_{21}$, $\mathbf{W}_{22}$, $\mathbf{b}_{21}$ and $\mathbf{b}_{22}$ be weights and biases of two decoder layers. A reconstructed input vector can be calculated as:

$$\mathbf{X}_i^{'} = g\Big(\mathbf{W}_{22}^T g\Big(\mathbf{W}_{21}^T \mathbf{h}_i + \mathbf{b}_{21}\Big) + \mathbf{b}_{22}\Big). \tag{6.2}$$

We have used mean square error to train the autoencoder. The cost function of first module of the proposed network is given as follows:

$$\begin{aligned} J_1 = \beta\Big(\|\mathbf{W}_{11}\|^2 + \|\mathbf{W}_{12}\|^2 + \|\mathbf{W}_{21}\|^2 + \|\mathbf{W}_{22}\|^2\Big) \\ + \alpha\frac{1}{N}\sum_{i=1}^{N}\|\mathbf{X}_i - \mathbf{X}_i^{'}\|^2. \end{aligned} \tag{6.3}$$

Here β is a weight decay parameter used for regularization, α is the learning rate and N is total number of training samples. The module is trained using Adam optimizer [68] with 0.01 learning rate and weight decay of 10^{-5}. Once the training of the first module is complete we disconnect the decoder layers.

Next, we feed the entire PolSAR image as an input to this network to obtain hidden representations of all pixels of the PolSAR image. We use this hidden representation of all pixels of the PolSAR image to generate superpixels. This process has two advantages: (i) it contains feature information of all bands and (ii) its dimensionality is substantially reduced in comparison to the input feature vector. To generate superpixels we have used simple linear iterative clustering (SLIC) [13]. Instead of giving an RGB image as an input to the SLIC algorithm, we give the hidden representation of the PolSAR image obtained from first module of the proposed network as input. Using SLIC we measure the distance between any two pixels which is given by eq. (6.4):

$$\begin{aligned} D_s &= \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}, \\ D_h &= \|\mathbf{h_i} - \mathbf{h_j}\|_2, \\ D &= \frac{m}{s}D_s + D_h. \end{aligned} \tag{6.4}$$

Here m is a parameter controlling the relative weight between D_s and D_h and s is a size of search space [13]. (x_i, y_i) and (x_j, y_j) are the positions of i^{th} and j^{th} PolSAR pixels on Euclidean plane. D_s measures how far two pixels are in a 2D image plane, pixels that are far from each other are less likely to be related. On the other hand D_h measures how similar two pixels are, pixels that are highly similar would probably belong to same class. The SLIC algorithm starts by sampling K superpixel cluster centers approximately equidistant. At each iteration, the al-

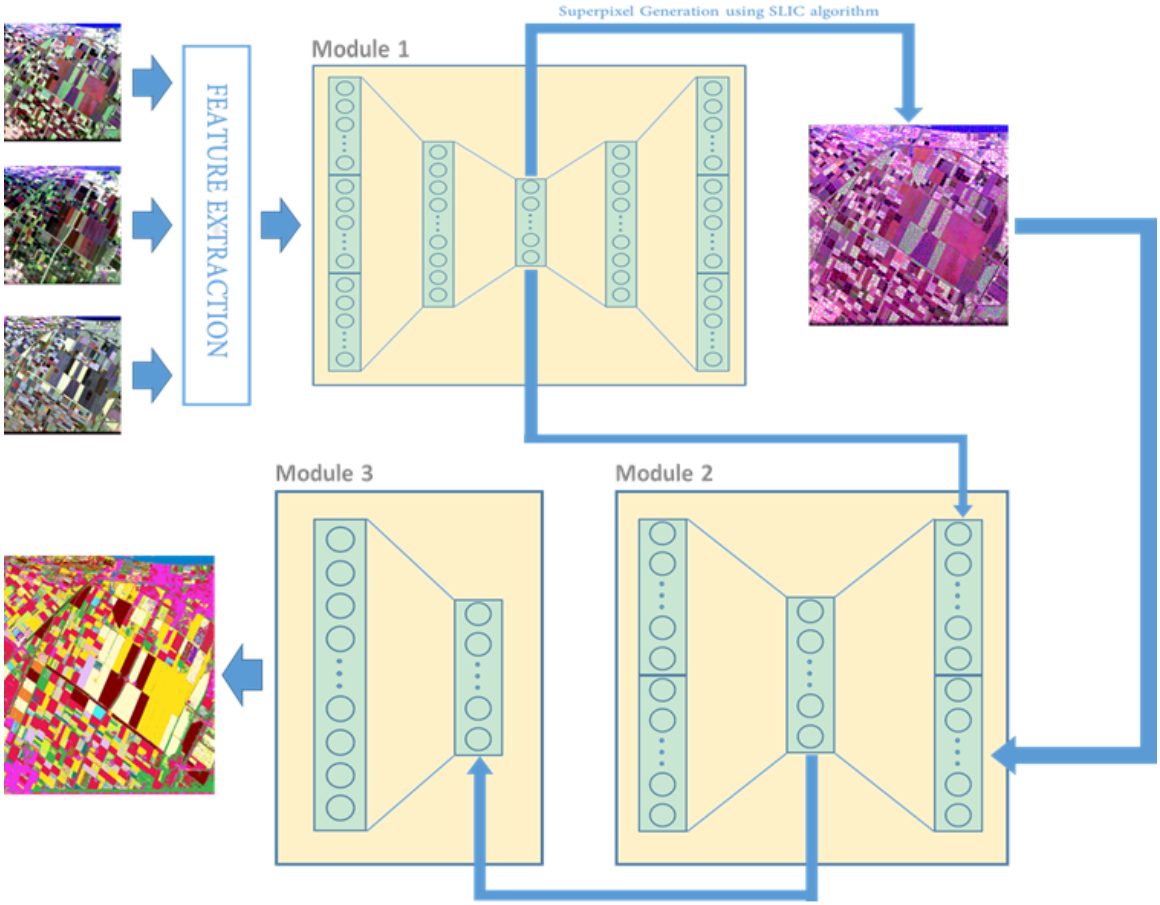


Figure 6.1: Proposed network architecture.

gorithm finds nearest cluster center of each pixel using distance measure shown in eq. (6.4). After that, new cluster centers are computed. These two steps are repeated until convergence.

The second module of our proposed architecture combines each pixel and corresponding superpixel information to construct robust feature vector. This is done by letting $\mathbf{S}_j$ be the j^{th} superpixel and $\mathbf{h}_i \in \mathbf{S}_j$. Let $\mathbf{c}_j$ be the cluster center of $\mathbf{S}_j$. To extract robust feature using both pixel and superpixel information input for second autoencoder $\mathbf{H}_i$ can be constructed as $\mathbf{H}_i = [\mathbf{h}_i; \mathbf{c}_j]$ [132]. Since the dimensionality of the hidden representation h_i is low, a single layer autoencoder is sufficient for an effective reconstruction. Let $\mathbf{r}_i = g(\mathbf{W}_{13}^T \mathbf{H}_i + \mathbf{b}_{13})$ be the activation value obtained at hidden layer of second autoencoder. We may then define $\mathbf{H}_i^{'}$ as the reconstructed input given as:

$$\mathbf{H}_i^{'} = g(\mathbf{W}_{23}^T \mathbf{r}_i + \mathbf{b}_{23}) \tag{6.5}$$

The cost function for the second autoencoder can now readily be described by:

$$J_2 = \beta(\|\mathbf{W}_{13}\|_2^2 + \|\mathbf{W}_{23}\|_2^2) + \lambda \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{H}_i - \mathbf{H}_i^{'}\|^2. \tag{6.6}$$

Once the training of the second module of the autoencoder is complete we again disconnect the decoder layer. From what we described above it is clear that the output of the second module of autoencoder contains both pixel and superpixel information. Therefore we expect better classification efficiency as we have seen in Chapter 5. We use softmax classifier to obtain predicted probability.

6.3 Experiments and Results

With this theoretical model we conduct the following experiments. We start with the details of dataset chosen for our experiments. After that we analyze the performance of the proposed network for different band combinations.

Experiments have been conducted on a dataset of Flevoland [1], an agricultural tract in the Netherlands, whose data was captured by the NASA/Jet Propulsion Laboratory on 15 June, 1991. This dataset has often been viewed as the benchmark dataset for PolSAR applications. The intensities after Pauli decomposition of the dataset have been used to form an RGB image as shown in Figures 6.2(a-c). The ground truth of the data set shown in Figure 6.2(d) identifies a total of 15 classes of land cover.

We have evaluated the accuracy of each class with respect to all possible combinations of the data acquired in the three frequency bands. From Table 6.2 it can be observed that in case of individual bands, the network with just the C band information recognizes Onions and Lucerne with better accuracy in comparison with L and P bands, but, it fails in the case of classes such as Beet or Oats. In case of P band, we observe higher accuracy in Beet and Fruit in comparison to C and L band, but, fails to accurately identify Onions and Peas. It can also be observed that the L band performs better in majority of the classes than the C or the P band. It performs poorly in case of Oats and Onions. Finally we also observe that combining different bands information produces even better classification results. The best overall accuracy of 99.59% was observed by combining all three bands (LPC).

A total of six PolSAR decomposition methods have been applied to the PolSAR data and their features have been given as input to the proposed network. The significance of each decomposition technique is evident from the Table 6.3. It can be seen that the Krogager and Freeman decompositions fail to provide enough

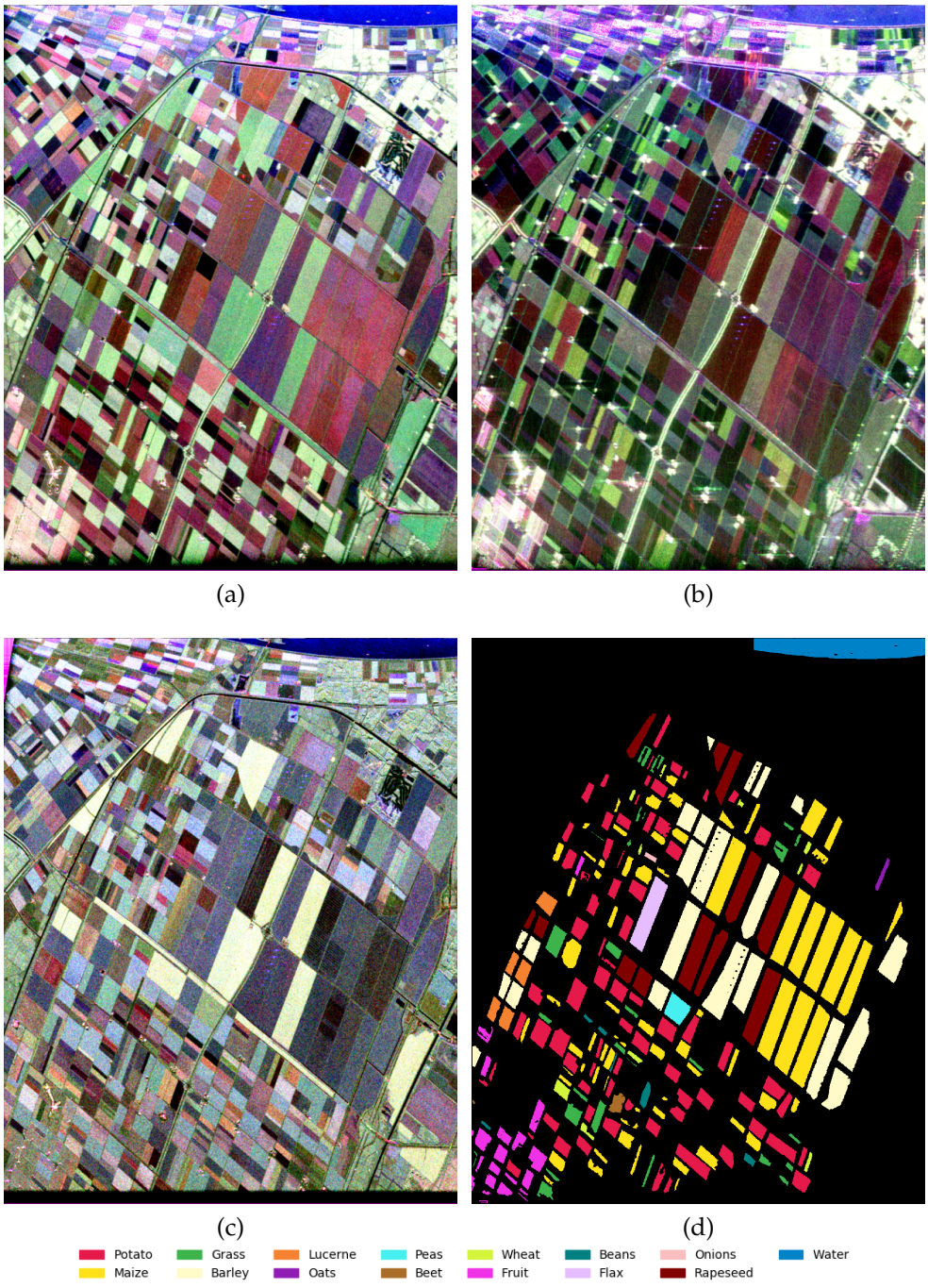


Figure 6.2: Pauli decomposition of (a) L band, (b) P band and (c) C band Flevoland dataset [1]. (d) Ground truth map of Flevoland dataset

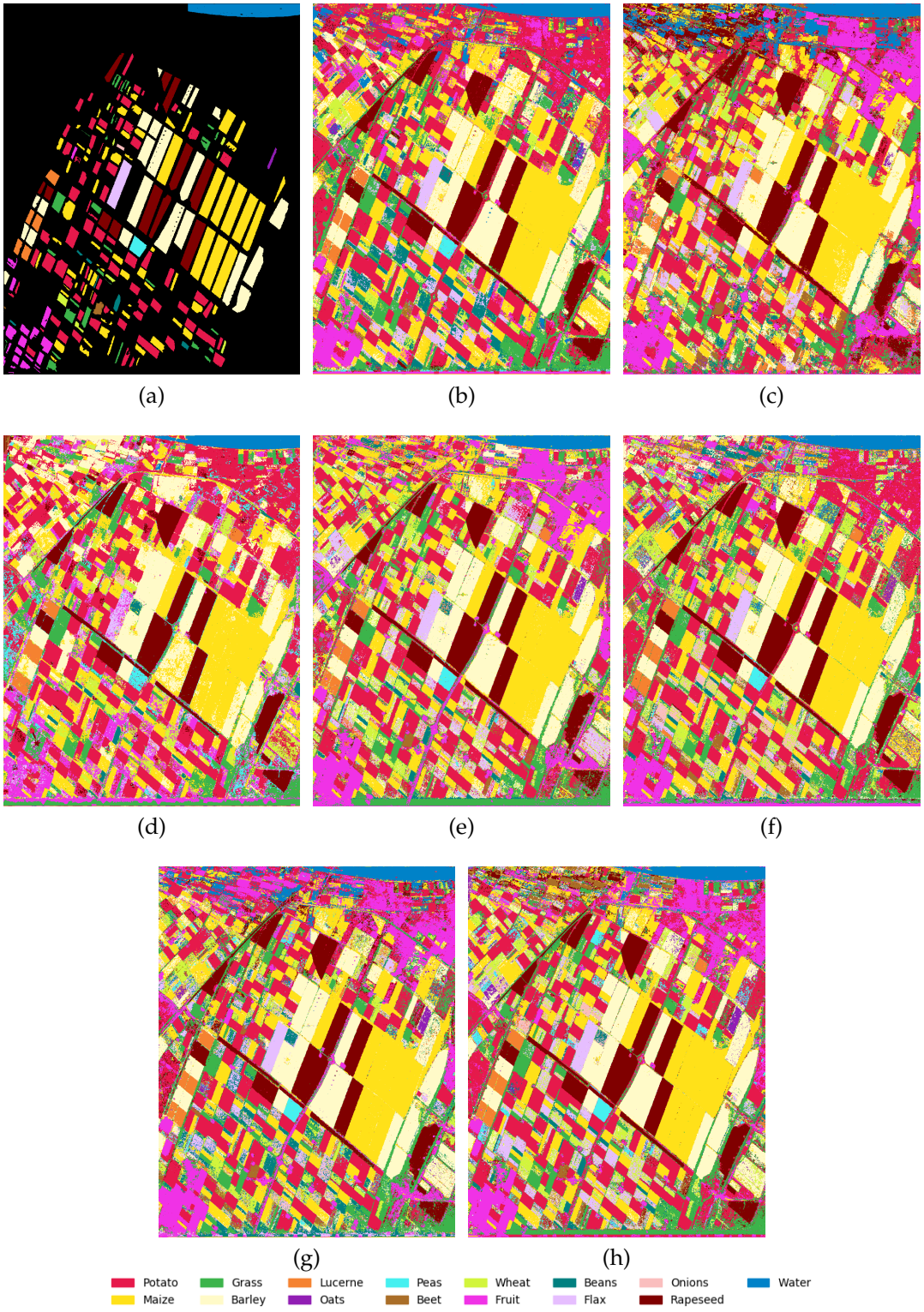


Figure 6.3: (a) Ground truth map, Classification map of Flevoland dataset obtained using (b) L band, (c) P band, (d) C band, (e) P and C bands , (f) L and C bands, (g) L and P bands and (h) L, P and C bands.

Table 6.2: Class-wise accuracies for different band combinations

Class/Bands	L	P	C	LP	LC	PC	LPC
Potato	99.12	96.36	98.90	99.77	99.61	99.31	99.75
Maize	98.59	97.45	92.92	99.63	99.32	99.17	99.79
Grass	93.39	82.82	85.85	97.41	94.57	96.08	97.73
Barley	97.57	97.58	88.48	99.60	99.30	98.57	99.79
Lucerne	95.06	81.48	95.73	97.98	98.79	96.38	98.69
Oats	46.41	17.71	00.00	89.16	62.75	74.66	87.18
Peas	96.42	00.24	89.75	99.52	98.27	98.07	99.52
Beet	80.24	94.90	16.67	96.08	95.45	96.27	97.72
Wheat	87.56	69.40	47.92	93.75	93.28	86.33	95.38
Fruit	95.05	98.19	75.91	99.31	96.74	99.06	99.44
Beans	89.54	49.90	87.84	94.10	97.67	95.34	98.50
Flax	98.56	94.09	67.49	99.84	98.60	97.18	99.66
Onions	23.53	00.00	88.93	47.06	92.73	29.07	92.75
Rapeseed	99.52	99.51	99.82	99.86	99.94	99.88	99.94
Water	99.79	99.64	99.97	99.75	99.83	99.83	99.95
OA	**97.84**	**95.08**	**92.11**	**99.38**	**99.00**	**98.69**	**99.59**

information to recognize Beet and Onions. On the other hand Cloude decomposition and Huynen decomposition provide sufficient information for these crops respectively.

The proposed method for classification of multifrequency PolSAR image is also compared with other methods available in the literature. Comparison of overall accuracies is reported in Table 6.4. ANN [22] and e-OWN [41] have used small subset of Flevoland dataset containing 7 classes. On the other hand Stein-SRC [128] used ground truth with 14 classes. For parity in comparison we have calculated the overall accuracy of the proposed method using ground truth of 7, 14 as well as 15 classes. For the brevity of comparison these are listed in Table 6.4. One can readily see that our proposed method outperforms all the other three methods [22, 41, 128].

Table 6.3: Class-wise accuracies for different features

/	Yamaguchi	Coherency	Krogager	Freeman	Huynen	Cloude
Potato	0.9820	0.9860	0.9879	0.9721	0.9933	0.9941
Maize	0.9672	0.9696	0.9880	0.9465	0.9904	0.9884
Grass	0.9243	0.8898	0.8490	0.6757	0.9256	0.9555
Barley	0.9653	0.9516	0.9814	0.9554	0.9920	0.9846
Lucerne	0.6690	0.8568	0.4815	0.4820	0.9695	0.9522
Oats	0.0626	0.0031	0.2168	0.0000	0.3420	0.7252
Peas	0.9779	0.8686	0.7440	0.5245	0.9546	0.9904
Beet	0.1658	0.6794	0.0000	0.0000	0.8834	0.9244
Wheat	0.1460	0.7210	0.6841	0.0000	0.8916	0.9428
Fruit	0.9618	0.9661	0.9827	0.9027	0.9790	0.9874
Beans	0.7831	0.9265	0.8758	0.0160	0.9767	0.9167
Flax	0.9346	0.9636	0.7162	0.7470	0.9790	0.9688
Onions	0.0000	0.2561	0.0000	0.0000	0.8720	0.1246
R.seed	0.9921	0.9949	0.9976	0.9786	0.9976	0.9986
Water	0.9889	0.9957	0.9958	0.9926	0.9981	0.9968
OA	**0.9509**	**0.9609**	**0.9562**	**0.9105**	**0.9859**	**0.9856**

Table 6.4: Classification's overall accuracy (OA) comparison of Flevoland dataset

Method	Number of classes	Accuracy
ANN [22]	7	98.23
e-OWN [41]	7	98.56
Stein-SRC [128]	14	99.00
Proposed Method	7	99.93
Proposed Method	14	99.69
Proposed Method	15	99.59

6.4 Conclusion

In this Chapter a stacked autoencoder based classification network for multifrequency PolSAR data is proposed to exploit maximum features extracted from the scattered waves using different available decomposition techniques in the literature. We could collect 33 features from five decomposition techniques per band giving 99 features in total following this approach. Then we proposed a network which involved three modules. First module reduced the dimensionality of the input feature vector. To ensure best classification result we also took care of the speckle noise in the PolSAR images. For this we recalled the superpixels formulation as we did in Chapter 5. Output of the first module was used to generate superpixels. The second module constructed robust feature vector using each pixel and its corresponding superpixel information. Finally the last module of the proposed network conducted classification using softmax classifier. It is observed that combining multiple frequency bands information improved overall classification accuracy. We validated our proposed network on Flevoland dataset that resulted in 99.59% overall classification accuracy. Experimental result showed that this proposed network outperformed other reported methods available in the literature.

CHAPTER 7
Thesis Summary

In this thesis we have addressed the problem of classification of single frequency and multifrequency PolSAR images by providing novel neural network based techniques. The problem of classification is a classic problem of literature. We described in Chapter 1 that one of the most popular technique was to measure Wishart distance. We began with the proposal of an Optimized Wishart Network (OWN) for single frequency PolSAR data. It uses a novel real-valued vector representation of a complex-valued coherency matrix that substantially reduces the computational overhead yet retains the phase information of the coherency matrix. Next, a better technique of initializing weights was introduced for increasing number of hidden units. The basic premises of a Wishart Network is that it assumes each class of a PolSAR image as a homogeneous. But in true and practical terms this is not the case. It means we have to consider heterogeneity within each class of a PolSAR image. To address this issue we proposed to use k-means algorithm to find multiple cluster centers within each class. These cluster centers are then used to initialize the weights of input-hidden layer of the OWN. We showed that proposed OWN was faster by at least two times than the WN due to real-valued vector representation of the coherency matrix. We also achieved better classification accuracy compared to the WN by including heterogeneity in each class through the weight initialization of input-hidden layer.

Having done so it was natural to examine the same for multifrequency bands of a given PolSAR image. For that we proposed an extended Optimized Wishart Network (e-OWN) for the classification of multifrequency PolSAR data. It is known that the different frequency bands of a PolSAR image provide different information of the target. Each band has different wavelength, hence has different penetration capability, so it interacts with different part of the target. Therefore in this proposal we combine information of multiple frequencies that resulted into improved overall classification accuracy. To effectively combine information from multiple frequencies, we proposed multi input branch single hidden layer neural

network called e-OWN. Input layer of e-OWN has multiple branches, one for each frequency band of a PolSAR image. Hidden layer stacks the output of all input branches. Hence, it represents combined information of all frequency bands. We showed that proposed approach was able to combine different bands information effectively and produced better classification in comparison to the ensemble learning of OWN. We also showed that even with only one hidden layer e-OWN was able to outperform deep learning based architecture containing multiple hidden layers.

Next, we explored fast global k-means algorithm to extract optimal cluster centers from the PolSAR image. Our proposal addressed two issues associated with k-means algorithm used in OWN and e-OWN. First, k-means uses Euclidean distance to determine the similarity between two PolSAR pixel. Since coherency matrix of a PolSAR pixel follows Wishart distribution, the Euclidean distance is not a valid distance metric. Hence, we propose the use of revised Wishart distance which considers the underlining probability distribution of the coherency matrix. Second, k-means algorithm does not guarantee the optimal cluster center because it starts with the randomly initialized cluster centers. To mitigate that we propose to use the fast global k-means algorithm which guarantees the optimal cluster centers. Combining the two proposal we designed the fast global k-means algorithm for PolSAR image. Our proposal reduces the computational overhead by employing fast determent and inverse calculation of coherency matrix using its real-valued vector representation. Using the proposed fast global k-means we developed Gk-OWN. We showed that Gk-OWN converges faster and achieves better classification accuracy in comparison with OWN.

Till now we discussed pixel based classification methods where each pixel was classified independently ignoring the spatial information present in the image. We also know that the PolSAR image suffers heavily from the speckle noise arising from the constructive and destructive interference of the scattered wave. The speckle noise affects the classification performance of the pixel-wise classifiers. To address this issue we proposed to use the spatial information in the image to reduce the effect of the speckle noise. To do so, we used SLIC algorithm to divide the PolSAR image into small homogeneous regions called superpixels. To incorporate superpixel information into the classification process we computed two distance measures: (i) revised Wishart distance of a pixel from all cluster centers and (ii) revised Wishart distance of the corresponding superpixel to all cluster centers. To use these distance measures, we primarily explored two ways , e.g., (i) by using additive distance measure of both distances explained as $S{+}P_{(Gk-OWN)}$

and (ii) by using dual branch OWN explained in $SP_{(Gk-OWN)}$. We concluded that the proposed network could efficiently combine pixel and superpixel information to achieve high classification accuracy.

Finally, we proposed deep learning based classification network for multifrequency PolSAR data. Proposed network involved three modules. In order to do so, 33 polarimetric features were extracted from each band of the PolSAR image using six standard decomposition techniques. Since this high dimensional feature vector contains some redundant information, the first module of the proposed network reduces the dimensionality of the input feature vector. Next, the output of the first module was used to generate superpixels. The second module constructs robust feature vector using each pixel and its corresponding superpixel information. Finally, the last module of the proposed network conducts classification using softmax classifier. It is observed that combining multiple frequency bands information improves overall classification accuracy. We validated our proposed network on Flevoland dataset resulted in 99.59% overall classification accuracy. Experimental result shows that this proposed network outperforms other reported methods available in the literature.

7.1 Future Scope of the Work

In case of traditional multilayer neural network, weights of all layers are initialized randomly. So, increasing or decreasing size of any hidden layer is straightforward. But, in case of OWN based networks, the size of hidden layer is dependent of the number of classes. Because weights associated with all hidden units are not random, but, are initialized as a cluster centers from all classes. So, increasing the size of hidden layer indefinitely is not possible.

In this thesis, we proposed two different approaches for initializing weights of the OWN based networks, namely, k-means with Euclidean distance and global k-means with RWD. Both of these methods uses training samples along with its label information in order to find the cluster centers. Typically, number of training samples used in PolSAR image classification are small. In future, one can develop method of using unlabeled samples to find more effective initialization of weights in an unsupervised manner.

Proposed OWN based networks are single hidden-layer networks. We showed that even with shallow architecture, proposed approaches are comparable and in some case outperforms deep architectures. In future, OWN based networks can be extended to deep architectures by stacking modules of such networks. This

may lead to better generalization capability then the current shallow network.

Currently, the number of cluster centers selected from each class are set as same. In reality each class can have different number of clusters. In future this problem can be addressed by finding optimal number of clusters from each class.

References

[1] Dataset: AIRSAR, NASA 1991. Retrieved from ASF DAAC on 7 December, 2018.

[2] Seasat data 1978 (NASA). Processed by ASF DAAC 2013. Retrieved from ASF DAAC 7 December 2019.

[3] Dataset: ERS-1, ESA 1993. Retrieved from ASF DAAC 7 December 2019.

[4] Dataset: JERS-1, ESA 1993. Retrieved from ASF DAAC 7 December 2019.

[5] Dataset: © JAXA/METI ALOS PALSAR L1.0 2007. Accessed through ASF DAAC 11 June 2019.

[6] Dataset: RADARSAT-1, CSA 1999. Retrieved from ASF DAAC 7 December 2019.

[7] Copernicus Sentinel data 2015. Retrieved from ASF DAAC 7 December 2019, processed by ESA.

[8] SMAP data 2015 (NASA). Dataset: SMAP SMAP_L1B_S0_LoRes_V2. Retrieved from ASF DAAC 7 December 2019.

[9] Dataset: UAVSAR, NASA 2011. Retrieved from ASF DAAC 7 June 2019.

[10]

[11] Airsar data set, the polarimetric data provided by the nasa/jpl-caltech., Accessed: July, 8, 2020. [Online]. Available: https://earth.esa.int/web/polsarpro/data-sources/.

[12] E-sar data set, the polarimetric data provided by dr. kostas papathanassiou., Accessed: July, 8, 2020. [Online]. Available: https://earth.esa.int/web/polsarpro/data-sources/.

[13] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, Nov 2012.

[14] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine learning*, 36(1-2):105–139, 1999.

[15] H. Bi, J. Sun, and Z. Xu. A graph-based semisupervised deep learning model for polsar image classification. *IEEE Transactions on Geoscience and Remote Sensing*, pages 1–17, 2018.

[16] Y. Cao, Y. Wu, P. Zhang, W. Liang, and M. Li. Pixel-wise PolSAR image classification via a novel complex-valued deep fully convolutional network. *Remote Sensing*, 11(22):2653, Nov. 2019.

[17] E. T. C. Chagas, A. C. Frery, O. A. Rosso, and H. S. Ramos. Analysis and classification of sar textures using information theory. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14:663–675, 2021.

[18] K. S. Chen, W. P. Huang, D. H. Tsay, and F. Amar. Classification of multifrequency polarimetric sar imagery using a dynamic learning neural network. *IEEE Transactions on Geoscience and Remote Sensing*, 34(3):814–820, 1996.

[19] S. W. Chen and C. S. Tao. Polsar image classification using polarimetric feature driven deep convolutional neural network. *IEEE Geoscience and Remote Sensing Letters*, 15(4):627–631, April 2018.

[20] S. R. Cloude and E. Pottier. An entropy based classification scheme for land applications of polarimetric sar. *IEEE Transactions on Geoscience and Remote Sensing*, 35(1):68–78, Jan 1997.

[21] D. Coelho, R. Cintra, A. Frery, and V. Dimitrov. Fast matrix inversion and determinant computation for polarimetric synthetic aperture radar. *Computers & Geosciences*, 119:109–114, 2018.

[22] S. De, D. Ratha, D. Ratha, A. Bhattacharya, and S. Chaudhuri. Tensorization of multifrequency polsar data for classification using an autoencoder network. *IEEE Geoscience and Remote Sensing Letters*, 15(4):542–546, April 2018.

[23] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[24] X. Deng and C. Lopez-Martinez. Higher order statistics for texture analysis and physical interpretation of polarimetric sar data. *IEEE Geoscience and Remote Sensing Letters*, 13(7):912–916, 2016.

[25] L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer New York, 1996.

[26] W. Dierking and C. Wesche. C-band radar polarimetry-useful for detection of icebergs in sea ice? *IEEE Transactions on Geoscience and Remote Sensing*, 52(1):25–37, 2014.

[27] T. G. Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

[28] J. Ding, B. Chen, H. Liu, and M. Huang. Convolutional neural network with data augmentation for sar target recognition. *IEEE Geoscience and Remote Sensing Letters*, 13(3):364–368, March 2016.

[29] Z. Ding, T. Zeng, F. Dong, L. Liu, W. Yang, and T. Long. An improved polsar image speckle reduction algorithm based on structural judgment and hybrid four-component polarimetric decomposition. *IEEE Transactions on Geoscience and Remote Sensing*, 51(8):4438–4449, Aug 2013.

[30] A. P. Doulgeris, S. N. Anfinsen, and T. Eltoft. Classification with a non-gaussian model for polsar data. *IEEE Transactions on Geoscience and Remote Sensing*, 46(10):2999–3009, Oct 2008.

[31] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

[32] J. Fan, X. Wang, X. Wang, J. Zhao, and X. Liu. Incremental wishart broad learning system for fast polsar image classification. *IEEE Geoscience and Remote Sensing Letters*, 16(12):1854–1858, 2019.

[33] J. Feng, Z. Cao, and Y. Pi. Polarimetric contextual classification of polsar images using sparse representation and superpixels. *Remote Sensing*, 6(8):7158–7181, 2014.

[34] P. Ferrazzoli, S. Paloscia, P. Pampaloni, G. Schiavon, S. Sigismondi, and D. Solimini. The potential of multifrequency polarimetric sar in assessing agricultural and arboreous biomass. *IEEE Transactions on Geoscience and Remote Sensing*, 35(1):5–17, Jan 1997.

[35] A. Ferretti, C. Prati, and F. Rocca. Nonlinear subsidence rate estimation using permanent scatterers in differential sar interferometry. *IEEE Transactions on Geoscience and Remote Sensing*, 38(5):2202–2212, 2000.

[36] L. Ferro-Famil, E. Pottier, and J.-S. Lee. Unsupervised classification of multifrequency and fully polarimetric sar images based on the h/a/alpha-wishart classifier. *IEEE Transactions on Geoscience and Remote Sensing*, 39(11):2332–2342, Nov 2001.

[37] L. FRASER and E. Siegler. High altitude research using the v-2 rocket. Technical report, JOHNS HOPKINS UNIV LAUREL MD APPLIED PHYSICS LAB, 1948.

[38] A. Freeman and S. L. Durden. A three-component scattering model for polarimetric sar data. *IEEE Transactions on Geoscience and Remote Sensing*, 36(3):963–973, May 1998.

[39] C. C. Freitas, A. C. Frery, and A. H. Correia. The polarimetric g distribution for sar data analysis. *Environmetrics*, 16(1):13–31, 2005.

[40] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, Apr. 1980.

[41] T. Gadhiya and A. K. Roy. Optimized wishart network for an efficient classification of multifrequency polsar data. *IEEE Geoscience and Remote Sensing Letters*, 15(11):1720–1724, Nov 2018.

[42] F. Gao, T. Huang, J. Wang, J. Sun, A. Hussain, and E. Yang. Dual-branch deep convolution neural network for polarimetric SAR image classification. *Applied Sciences*, 7(5):447, Apr. 2017.

[43] W. Gao, J. Yang, and W. Ma. Land cover classification for polarimetric sar images based on mixture models. *Remote Sensing*, 6(5):3770–3790, 2014.

[44] J. Geng, J. Fan, H. Wang, X. Ma, B. Li, and F. Chen. High-resolution sar image classification via deep convolutional autoencoders. *IEEE Geoscience and Remote Sensing Letters*, 12(11):2351–2355, Nov 2015.

[45] J. Geng, X. Ma, J. Fan, and H. Wang. Semisupervised classification of polarimetric sar image via superpixel restrained deep neural network. *IEEE Geoscience and Remote Sensing Letters*, 15(1):122–126, Jan 2018.

[46] D. G. Goodenough, H. Chen, A. Richardson, S. Cloude, W. Hong, and Y. Li. Mapping fire scars using radarsat-2 polarimetric sar data. *Canadian Journal of Remote Sensing*, 37(5):500–509, 2012.

[47] N. R. Goodman. Statistical analysis based on a certain multivariate complex gaussian distribution (an introduction). *The Annals of Mathematical Statistics*, 34(1):152–177, Mar. 1963.

[48] F. Gu, H. Zhang, and C. Wang. A classification method for polsar images using slic superpixel segmentation and deep convolution neural network. In *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, pages 6671–6674, July 2018.

[49] Y. Guo, L. Jiao, S. Wang, S. Wang, F. Liu, and W. Hua. Fuzzy superpixels for polarimetric sar images classification. *IEEE Transactions on Fuzzy Systems*, 26(5):2846–2860, Oct 2018.

[50] R. Haensch and O. Hellwich. Complex-valued convolutional neural networks for object detection in polsar data. In *8th European Conference on Synthetic Aperture Radar*, pages 1–4, 2010.

[51] I. Hajnsek, T. Jagdhuber, H. Schon, and K. P. Papathanassiou. Potential of estimating soil moisture under vegetation cover by means of polsar. *IEEE Transactions on Geoscience and Remote Sensing*, 47(2):442–454, 2009.

[52] R. Hansch and O. Hellwich. Skipping the real world: Classification of polsar images without explicit feature extraction. *ISPRS Journal of Photogrammetry and Remote Sensing*, 140:122 – 132, 2018.

[53] Y. Hara, R. G. Atkins, S. H. Yueh, R. T. Shin, and J. A. Kong. Application of neural networks to radar image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 32(1):100–109, 1994.

[54] M. Hellmann, G. Jager, E. Kratzschmar, and M. Habermeyer. Classification of full polarimetric sar-data using artificial neural networks and fuzzy algorithms. In *IEEE 1999 International Geoscience and Remote Sensing Symposium. IGARSS'99 (Cat. No.99CH36293)*, volume 4, pages 1995–1997 vol.4, 1999.

[55] B. Hou, H. Kou, and L. Jiao. Classification of polarimetric sar images using multilayer autoencoders and superpixels. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(7):3072–3081, July 2016.

[56] B. Hou, C. Yang, B. Ren, and L. Jiao. Decomposition-feature-iterative-clustering-based superpixel segmentation for polsar image classification. *IEEE Geoscience and Remote Sensing Letters*, 15(8):1239–1243, Aug 2018.

[57] Y. Hu, J. Fan, and J. Wang. Classification of polsar images based on adaptive nonlocal stacked sparse autoencoder. *IEEE Geoscience and Remote Sensing Letters*, 15(7):1050–1054, July 2018.

[58] D. H. Hubel and T. N. Wiesel. Receptive fields and functional architecture of monkey striate cortex. *The Journal of Physiology*, 195(1):215–243, Mar. 1968.

[59] J. R. Huynen. Phenomenological theory of radar targets. *Electronmagnetic Scattering*, 1970.

[60] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017.

[61] Y. Ito and S. Omatu. Polarimetric SAR data classification using competitive neural networks. *International Journal of Remote Sensing*, 19(14):2665–2684, Jan. 1998.

[62] J. Ji and K. Wang. A robust nonlocal fuzzy clustering algorithm with between-cluster separation measure for sar image segmentation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(12):4929–4936, 2014.

[63] L. Jiao and F. Liu. Wishart deep stacking network for fast polsar image classification. *IEEE Transactions on Image Processing*, 25(7):3273–3286, July 2016.

[64] Jong-Sen Lee, M. R. Grunes, and E. Pottier. Quantitative comparison of classification capability: fully polarimetric versus dual and single-polarization sar. *IEEE Transactions on Geoscience and Remote Sensing*, 39(11):2343–2351, 2001.

[65] M. Kajimoto and J. Susaki. Urban density estimation from polarimetric sar images based on a poa correction method. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 6(3):1418–1429, 2013.

[66] P. R. Kersten, J.-S. Lee, and T. L. Ainsworth. Unsupervised classification of polarimetric synthetic aperture radar images using fuzzy clustering and em clustering. *IEEE Transactions on Geoscience and Remote Sensing*, 43(3):519–527, March 2005.

[67] J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23(3):462–466, Sept. 1952.

[68] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. 2014.

[69] D. P. Kingma and M. Welling. Auto-encoding variational bayes. 2013.

[70] J. Kong, A. Swartz, H. Yueh, L. Novak, and R. Shin. Identification of terrain cover using the optimum polarimetric classifier. *Journal of Electromagnetic Waves and Applications*, 2(2):171–194, 1988.

[71] C. N. Koyama, H. Gokon, M. Jimbo, S. Koshimura, and M. Sato. Disaster debris estimation using high-resolution polarimetric stereo-sar. *ISPRS Journal of Photogrammetry and Remote Sensing*, 120:84 – 98, 2016.

[72] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, May 2017.

[73] E. Krogager. New decomposition of the radar target scattering matrix. *Electronics Letters*, 26(18):1525–1527, Aug 1990.

[74] F. Kugler, S. Lee, I. Hajnsek, and K. P. Papathanassiou. Forest height estimation by means of pol-insar data inversion: The role of the vertical wavenumber. *IEEE Transactions on Geoscience and Remote Sensing*, 53(10):5294–5311, 2015.

[75] C. Lardeux, P. Frison, C. Tison, J. Souyris, B. Stoll, B. Fruneau, and J. Rudant. Support vector machine for multifrequency sar polarimetric data classification. *IEEE Transactions on Geoscience and Remote Sensing*, 47(12):4143–4152, Dec 2009.

[76] T. Le Toan, A. Beaudoin, J. Riom, and D. Guyon. Relating forest biomass to sar data. *IEEE Transactions on Geoscience and Remote Sensing*, 30(2):403–411, 1992.

[77] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, Dec. 1989.

[78] J. S. Lee and M. R. Grunes. Classification of multi-look polarimetric sar data based on complex wishart distribution. In *[Proceedings] NTC-92: National Telesystems Conference*, pages 7/21–7/24, May 1992.

[79] J. S. Lee, M. R. Grunes, and S. A. Mango. Speckle reduction in multipolarization, multifrequency sar imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 29(4):535–544, Jul 1991.

[80] J.-S. Lee, M. R. Grunes, E. Pottier, and L. Ferro-Famil. Unsupervised terrain classification preserving polarimetric scattering characteristics. *IEEE Transactions on Geoscience and Remote Sensing*, 42(4):722–731, April 2004.

[81] X. Li, C. Li, P. Wang, Z. Men, and H. Xu. Sar atr based on dividing cnn into cae and snn. In *2015 IEEE 5th Asia-Pacific Conference on Synthetic Aperture Radar (APSAR)*, pages 676–679, Sept 2015.

[82] A. Likas, N. Vlassis, and J. J. Verbeek. The global k-means clustering algorithm. *Pattern Recognition*, 36(2):451 – 461, 2003.

[83] B. Liu, H. Hu, H. Wang, K. Wang, X. Liu, and W. Yu. Superpixel-based classification with an adaptive number of classes for polarimetric sar images. *IEEE Transactions on Geoscience and Remote Sensing*, 51(2):907–924, Feb 2013.

[84] C. Liu and C. H. Gierull. A new application for polsar imagery in the field of moving target indication/ship detection. *IEEE Transactions on Geoscience and Remote Sensing*, 45(11):3426–3436, 2007.

[85] C. Liu, J. Shang, P. W. Vachon, and H. McNairn. Multiyear crop monitoring using polarimetric radarsat-2 data. *IEEE Transactions on Geoscience and Remote Sensing*, 51(4):2227–2240, 2013.

[86] F. Liu, L. Jiao, B. Hou, and S. Yang. Pol-sar image classification based on wishart dbn and local spatial information. *IEEE Transactions on Geoscience and Remote Sensing*, 54(6):3292–3308, June 2016.

[87] F. Liu, L. Jiao, and X. Tang. Task-oriented gan for polsar image classification and clustering. *IEEE transactions on neural networks and learning systems*, 30(9):2707–2719, 2019.

[88] H. Liu, Q. Min, C. Sun, J. Zhao, S. Yang, B. Hou, J. Feng, and L. Jiao. Terrain classification with polarimetric sar based on deep sparse filtering network. In *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 64–67, July 2016.

[89] H. Liu, S. Yang, S. Gou, P. Chen, Y. Wang, and L. Jiao. Fast classification for large polarimetric sar data based on refined spatial-anchor graph. *IEEE Geoscience and Remote Sensing Letters*, 14(9):1589–1593, 2017.

[90] H. Liu, S. Yang, S. Gou, D. Zhu, R. Wang, and L. Jiao. Polarimetric sar feature extraction with neighborhood preservation-based deep learning. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(4):1456–1466, April 2017.

[91] H. Liu, D. Zhu, S. Yang, B. Hou, S. Gou, T. Xiong, and L. Jiao. Semisupervised feature extraction with neighborhood constraints for polarimetric sar classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(7):3001–3015, July 2016.

[92] M. Liu, H. Zhang, C. Wang, and F. Wu. Change detection of multilook polarimetric sar images using heterogeneous clutter models. *IEEE Transactions on Geoscience and Remote Sensing*, 52(12):7483–7494, 2014.

[93] X. Liu, L. Jiao, and F. Liu. Polsf: Polsar image dataset on san francisco, 2019.

[94] X. Liu, L. Jiao, X. Tang, Q. Sun, and D. Zhang. Polarimetric convolutional network for polsar image classification. *IEEE Transactions on Geoscience and Remote Sensing*, pages 1–15, 2018.

[95] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[96] C. Lopez-Martinez and X. Fabregas. Polarimetric sar speckle noise model. *IEEE Transactions on Geoscience and Remote Sensing*, 41(10):2232–2242, 2003.

[97] Y. Maghsoudi, M. J. Collins, and D. G. Leckie. Radarsat-2 polarimetric sar data for boreal forest classification using svm and a wrapper feature selector. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 6(3):1531–1538, 2013.

[98] L. Mascolo, J. M. Lopez-Sanchez, F. Vicente-Guijalba, F. Nunziata, M. Migliaccio, and G. Mazzarella. A complete procedure for crop phenology estimation with polsar data based on the complex wishart classifier. *IEEE Transactions on Geoscience and Remote Sensing*, 54(11):6505–6515, 2016.

[99] M. Migliaccio, A. Gambardella, and M. Tranfaglia. Sar polarimetry to observe oil spills. *IEEE Transactions on Geoscience and Remote Sensing*, 45(2):506–511, 2007.

[100] A. Moreira, P. Prats-Iraola, M. Younis, G. Krieger, I. Hajnsek, and K. P. Papathanassiou. A tutorial on synthetic aperture radar. *IEEE Geoscience and Remote Sensing Magazine*, 1(1):6–43, 2013.

[101] T. Nagler and H. Rott. Retrieval of wet snow by means of multitemporal sar data. *IEEE Transactions on Geoscience and Remote Sensing*, 38(2):754–765, 2000.

[102] C. Oliver and S. Quegan. *Understanding synthetic aperture radar images*. SciTech Publishing, 2004.

[103] G. A. Petsko. The blue marble. *Genome Biology*, 12(4):112, 2011.

[104] E. Pottier. Dr. j. r. huynen's main contributions in the development of polarimetric radar techniques and how the 'radar targets phenomenological concept' becomes a theory. In H. Mott and W.-M. Boerner, editors, *Radar Polarimetry*. SPIE, Feb. 1993.

[105] E. Pottier and J. Saillard. On radar polarization target decomposition theorems with application to target classification, by using neural network method. In *1991 Seventh International Conference on Antennas and Propagation, ICAP 91 (IEE)*, pages 265–268 vol.1, 1991.

[106] F. Qin, J. Guo, and F. Lang. Superpixel segmentation for polarimetric sar imagery using local iterative clustering. *IEEE Geoscience and Remote Sensing Letters*, 12(1):13–17, 2015.

[107] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[108] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the*

28th International Conference on International Conference on Machine Learning, ICML'11, pages 833–840, Madison, WI, USA, 2011. Omnipress.

[109] A. Romero, C. Gatta, and G. Camps-Valls. Unsupervised deep feature extraction for remote sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 54(3):1349–1362, March 2016.

[110] D. Schuler, J. Lee, D. Kasilingam, and E. Pottier. Measurement of ocean surface slopes and wave spectra using polarimetric sar image data. *Remote Sensing of Environment*, 91(2):198 – 211, 2004.

[111] A. Sharifi, J. Amini, J. T. S. Sumantyo, and R. Tateishi. Speckle reduction of polsar images in forest regions using fast ica algorithm. *Journal of the Indian Society of Remote Sensing*, 43(2):339–346, 2015.

[112] T. Shibayama and Y. Yamaguchi. A landslide detection based on the change of scattering power components between multi-temporal polsar data. In *2014 IEEE Geoscience and Remote Sensing Symposium*, pages 2734–2737, 2014.

[113] W. Song, M. Li, P. Zhang, Y. Wu, and X. Tan. Superpixel-based hybrid discriminative random field for fast polsar image classification. *IEEE Access*, 7:24547–24558, 2019.

[114] T. Strozzi, A. Luckman, T. Murray, U. Wegmuller, and C. L. Werner. Glacier motion estimation using sar offset-tracking procedures. *IEEE Transactions on Geoscience and Remote Sensing*, 40(11):2384–2391, 2002.

[115] Q. Sun, X. Li, L. Li, X. Liu, F. Liu, and L. Jiao. Semi-supervised complex-valued gan for polarimetric sar image classification. In *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*, pages 3245–3248. IEEE, 2019.

[116] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2015.

[117] H. Taubenbock, T. Esch, A. Felbier, M. Wiesner, A. Roth, and S. Dech. Monitoring urbanization in mega cities from space. *Remote Sensing of Environment*, 117:162–176, 2012. Remote Sensing of Urban Environments.

[118] T. Tieleman and G. Hinton. Lecture 6.5–RmsProp: Divide the gradient by a running average of its recent magnitude. 2012.

[119] S. T. Tu, J. Y. Chen, W. Yang, and H. Sun. Laplacian eigenmaps-based polarimetric dimensionality reduction for sar image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 50(1):170–179, 2012.

[120] Y. C. Tzeng and K. S. Chen. A fuzzy neural network to sar image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 36(1):301–307, 1998.

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

[122] Y. Wada, M. Ukawa, Y. Yamaguchi, and H. Ohkura. A land-cover monitoring for volcanoes by using alos-palsar quad-pol. data. In *IGARSS 2008 - 2008 IEEE International Geoscience and Remote Sensing Symposium*, volume 3, pages III – 19–III – 22, 2008.

[123] S.-H. Wang, J. Sun, P. Phillips, G. Zhao, and Y.-D. Zhang. Polarimetric synthetic aperture radar image segmentation by convolutional neural network using graphical processing units. *Journal of Real-Time Image Processing*, pages 1–12, 2017.

[124] W. Xie, L. Jiao, B. Hou, W. Ma, J. Zhao, S. Zhang, and F. Liu. Polsar image classification via wishart-ae model or wishart-cae model. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(8):3604–3615, Aug 2017.

[125] Y. Yamaguchi. Disaster monitoring by fully polarimetric sar data acquired with alos-palsar. *Proceedings of the IEEE*, 100(10):2851–2860, 2012.

[126] Y. Yamaguchi, T. Moriyama, M. Ishido, and H. Yamada. Four-component scattering model for polarimetric sar image decomposition. *IEEE Transactions on Geoscience and Remote Sensing*, 43(8):1699–1706, Aug 2005.

[127] C. Yang, B. Hou, B. Ren, Y. Hu, and L. Jiao. Cnn-based polarimetric decomposition feature selection for polsar image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 57(11):8796–8812, 2019.

[128] F. Yang, W. Gao, B. Xu, and J. Yang. Multi-frequency polarimetric sar classification based on riemannian manifold and simultaneous sparse representation. *Remote Sensing*, 7(7):8469–8488, 2015.

[129] K. Yoong and T. Bretschneider. Polsar speckle filtering with structural feature and scattering property preservation. In *Proceedings. 2005 IEEE International Geoscience and Remote Sensing Symposium, 2005. IGARSS '05.*, volume 1, page 4, July 2005.

[130] S. Yueh, J. A. Kong, J. Jao, R. Shin, and L. Novak. K-distribution and polarimetric terrain radar clutter. *Journal of Electromagnetic Waves and Applications*, 3(8):747–768, 1989.

[131] F. Zhang. *Hermitian Matrices*. Springer New York, New York, NY, 2011.

[132] L. Zhang, W. Ma, and D. Zhang. Stacked sparse autoencoder in polsar data classification using local spatial information. *IEEE Geoscience and Remote Sensing Letters*, 13(9):1359–1363, 2016.

[133] L. Zhang, L. Zhang, D. Tao, and X. Huang. On combining multiple features for hyperspectral remote sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 50(3):879–893, March 2012.

[134] L. Zhang, Q. Zhang, B. Du, X. Huang, Y. Y. Tang, and D. Tao. Simultaneous spectral-spatial feature selection and extraction for hyperspectral images. *IEEE Transactions on Cybernetics*, 48(1):16–28, Jan 2018.

[135] Y. Zhang, H. Zou, T. Luo, X. Qin, S. Zhou, and K. Ji. A fast superpixel segmentation algorithm for polsar images based on edge refinement and revised wishart distance. *Sensors*, 16(10):1687, 2016.

[136] Z. Zhang, H. Wang, F. Xu, and Y. Q. Jin. Complex-valued convolutional neural network and its application in polarimetric sar image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(12):7177–7188, Dec 2017.

[137] Y. Zhou, H. Wang, F. Xu, and Y. Q. Jin. Polarimetric sar image classification using deep convolutional neural networks. *IEEE Geoscience and Remote Sensing Letters*, 13(12):1935–1939, Dec 2016.

Our Publications

- **Journal:**

  1. T. Gadhiya and A. K. Roy, "Superpixel-Driven Optimized Wishart Network for Fast PolSAR Image Classification Using Global k-means Algorithm," in IEEE Transactions on Geoscience and Remote Sensing, vol. 58, no. 1, pp. 97-109, Jan. 2020, doi: 10.1109/TGRS.2019.2933483.
  2. T. Gadhiya and A. K. Roy, "Optimized Wishart Network for an Efficient Classification of Multifrequency PolSAR Data," in IEEE Geoscience and Remote Sensing Letters, vol. 15, no. 11, pp. 1720-1724, Nov. 2018, doi: 10.1109/LGRS.2018.2861081.

- **Conferences:**

  1. T. Gadhiya, S. Tangirala and A. K. Roy, "Stacked Autoencoder Based Feature Extraction and Superpixel Generation for Multifrequency PolSAR Image Classification," in the proceedings of the Pattern Recognition and Machine Intelligence (PReMI), Tezpur, India, 2019, LNCS 11942, pp. 331-339, doi: 10.1007/978-3-030-34872-4_37.
  2. T. Gadhiya and A. K. Roy, "Classification of Polarimetric Synthetic Aperture Radar Images Using Revised Wishart Distance," 2018 15th IEEE India Council International Conference (INDICON), Coimbatore, India, 2018, pp. 1-6, doi: 10.1109/INDICON45594.2018.8987089.