ROBUST IMAGE CLASSIFICATION ALGORITHMS FOR MULTISPECTRAL AND HYPERSPECTRAL DATA IN REAL-TIME ENVIRONMENTS

A thesis submitted
in partial fulfillment for the degree of

Doctor of Philosophy

by

DUBACHARLA GYANESHWAR

DEPARTMENT OF EARTH AND SPACE SCIENCES

INDIAN INSTITUTE OF SPACE SCIENCE AND TECHNOLOGY
THIRUVANANTHAPURAM – 695547

JULY 2021

CERTIFICATE

This is to certify that the thesis titled **Robust Image Classification Algorithms for Multispectral and Hyperspectral Data in Real-Time Environments**, submitted by **Mr. Dubacharla Gyaneshwar**, to the Indian Institute of Space Science and Technology, Thiruvananthapuram, for the award of the degree of **Doctor of Philosophy**, is a bona fide record of the research work done by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Rama Rao Nidamanuri
Professor (Remote sensing and Image processing)
Supervisor
Department of Earth and Space Sciences

Thiruvananthapuram
July 2021

Counter signature of HOD with seal

DECLARATION

I declare that this thesis titled **Robust Image Classification Algorithms for Multispectral and Hyperspectral Data in Real-Time Environments** submitted in fulfillment of the Degree of Doctor of Philosophy is a record of original work carried out by me under the supervision of **Dr. Rama Rao Nidamanuri**, and has not formed the basis for the award of any degree, diploma, associateship, fellowship or other titles in this or any other Institution or University of higher learning. In keeping with the ethical practice in reporting scientific information, due acknowledgments have been made wherever the findings of others have been cited.

Dubacharla Gyaneshwar
SC16D041

Place: Thiruvananthapuram
July 2021

ACKNOWLEDGEMENTS

My Ph.D. journey was enriched significantly through the support of the people around me, and I owe my sincere gratitude to many individuals.

First and foremost, I express my sincere thanks to my supervisor Dr. Rama Rao Nidamanuri for his support throughout my Ph.D. studies and research; for his patience, motivation, enthusiasm, and insightful discussions. His guidance has motivated me to become an independent researcher, which helped me in writing this thesis and the work that preceded it.

Besides my supervisor, I would like to express my sincere thanks to the rest of my Doctoral committee members: Prof. B. Krishna Mohan, Dr. L Gnanappazham, Dr. J Sheeba Rani, Dr. Samir Mandal, Prof. Anandmayee Tej for their encouragement, comments, and suggestions. I would like to acknowledge Director, IIST, Dr. V. K. Dadhwal for providing me this opportunity and support throughout my Ph.D. I would like to express my appreciation to the entire staff and lab members of the department, library, and administrative staff at IIST for their timely help. I would like to thank the anonymous journal reviewers for their constructive criticism and invaluable suggestions.

I am extremely grateful to Dr. N. Srinivasa Rao, my master co-supervisor, for his inspiration, motivation, kind wishes, and positive energies that pushed me forward to pursue doctoral research.

Furthermore, my greatest appreciation goes to my parents, Mr. Dubacharla Satya Venkata Ramana and Mrs. Dubacharla Raja Lakshmi, and my elder brother, Mr. Dubacharla Jaya Shanker, for all the endless patience, trust, timely encouragement, and unconditional love that they have showered upon me throughout my life. Last but not least, I also like to convey my sincere thanks to all my teachers and friends for their love, support, and excitement.

Dubacharla Gyaneshwar

ABSTRACT

Image classification is one of the most prominent techniques of analysis for image interpretation and information mining tasks. It has been applied in a wide variety of remote sensing image-driven applications. The quality and operational-level utility of information from remote sensing imagery is primarily influenced by the adaptability of classification methods for information mining from imagery in a time-critical manner and with minimal human expert's involvement. The development of efficient image classification algorithms is essential to cope-up with the ever-changing and ongoing requirements of real-world applications. As a concrete example, advances in remote sensing such as very high-resolution multispectral images (MSIs) and hyperspectral images (HSIs) have opened up and fostered new opportunities in expanding the horizons of mapping applications using remotely-sensed imagery. In recent years, the increasing availability of high spatial-spectral resolution imaging has been evolving as the viable and cost-effective remotely sensed data source in various high precision and high accuracy remote sensing applications from the land surface and industrial perspectives. The bulk of classification approaches available offer good performance for the ideal scenario of having priori comprehensive ground truth information on the type, number, and spatial distribution of information classes in the imagery. In a realistic environment of imagery acquisition, these requirements are rarely met and there is always a demand for more ground truth information and hence better training of the classification model. As a result, most of the existing techniques are not efficient for the classification tasks under uncertainties such as *unseen*, *unknown*, and *dynamic environments* in terms of accuracy, training information dependencies, and minimizing computation times in a firm or near real-time environment, especially for high dimensionality data like HSIs.

The aim of this thesis is the development of efficient and robust image classification algorithms based on supervised learning approaches that are suitable for both static and dynamic real-time operational environments. To realize this aim, four different strategies, aka objectives, are considered in this study using various MSIs and HSIs of different spatial and spectral resolutions to realize the intended objective. First, a field-programmable gate array (FPGA) based real-time decision intelligence system architecture is designed using low-complexity rapid prototyping tools. Second, a novel classifier is developed that maintains stable classification performance in both static and dynamic environments. Third, a new category of shadow and illumination invariant image classification algorithms is proposed. Besides, a novel mechanism to determine the optimal threshold for the proposed version of spectral similarity matching methods (SMMs) is also proposed. Finally, a new category of open-set classifiers that combine the advantages of a hardware accelerator, i.e., FPGA-based real-time processing and open-set classification is proposed. A new classification model architecture is also presented to get the time-critical computational benefit of the proposed open-set classifier. A thorough experimental evaluation using different MSIs and HSIs, captured from multi-sensor and multi-platform modalities, demonstrates the proposed algorithms' practical and robust classification performance. The contributions of this thesis, methods, and algorithms for remote sensing imagery classification in real-time or near real-time perspective are vital in helping realize automatic image analytic workflows in operational stream imagery analyses application.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

1D-DCNN	One Dimensional Deep Convolutional Neural Network
2D	Two Dimensional
3D	Three Dimensional
AA	Average Accuracy
ASIC	Application Specific Integrated Circuit
AVIRIS	Airborne Visible Infrared Imaging Spectrometer
AVIRIS-NG	Airborne Visible Infrared Imaging Spectrometer Next Generation
BC	Binary Classifier
BIL	Band Interleaved by Line
BIP	Band Interleaved by Pixel
BIP	Band Interleaved by Pixel
BSQ	Band SeQuential
CA	Class Aware
CAG-SC^{2}S	Class Aware Global Supervised Cascaded Classifier System
CAG-SC^{2}S-I	Class Aware Global Supervised Cascaded Classifier System Type-I
CAG-SC^{2}S-II	Class Aware Global Supervised Cascaded Classifier System Type-II
CAL-SC^{2}S	Class Aware Local Supervised Cascaded Classifier System
CAL-SC^{2}S-I	Class Aware Local Supervised Cascaded Classifier System Type-I
CAL-SC^{2}S-II	Class Aware Local Supervised Cascaded Classifier System Type-II
CNN	Convolution Neural Network
CVA	Cross Validation Accuracy
DCNN	Deep Convolutional Neural Network
DFE	Data Flow Engine
DL	Deep Learning
DSP	Digital Signal Processing
ELM	Extreme Learning Machine
FCC	False Color Composite
FPGA	Field Programmable Gate Array

GPU	Graphics Processing Unit
G-SNC	Global Supervised Novelty Classifier
HDL	Hardware Description Language
HPC	High Performance Computing
HSI	HyperSpectral Image
HW	HardWare
ID	In Distribution
IL	In Light
IP	Intellectual Property
IS	In Shadow
JTAG	Joint Test Action Group
KC	Known Class
k-NN	k Nearest Neighbor
L-SNC	Local Supervised Novelty Classifier
LULC	Land Use and Land Cover
LUT	Look Up Table
MAP	Maximum A Posterior
MCC	Multi Class Classification
MSI	MultiSpectral Image
MSVM	Multi-class Support Vector Machine
NN	Neural Network
NNDR	Nearest Neighbor Distance Ratio
OA	Overall Accuracy
OAA	One-Against-All
OAO	One-Against-One
OCC	One-Class Classification
OCSVM	One-Class Support Vector Machines
OoD	Out of Distribution
OSNN	Open-Set Nearest Neighbor
OSNN^{CV}	Open-Set Nearest Neighbor Class Verification
$\text{OSNN}^{\text{NNDR}}$	Open-Set Nearest Neighbor Nearest Neighbors distance ratio

OVO	One-Versus-One
PA	Producer’s Accuracy
PCI	Peripheral Component Interconnect
PDF	Probability Distribution Function
PE	Processing Element
PSoC	Programmable System on Chip
P-SVM	Platt probability estimation-based Support Vector Machine
QP	Quadratic Programming
RAM	Random Access Memory
RF	Random Forest
RGB	Red, Green, and Blue
ROM	Read Only Memory
RSC	Rule-based Supervised Classifier
SAM	Spectral Angle Mapper
SC	Supervised Classifier
SC2S	Supervised Cascaded Classifier system
SCM	Spectral Correlation Mapper
SD	Standard Deviation
SI2CS	Shadow and Illumination Invariant Classifier System
SID	Spectral Information Divergence
SMM	Similarity Matching Method
SNC	Supervised Novelty Classifier
SV	Support Vector
SVM	Support Vector Machines
SW	SoftWare
SWIL	Sea, Water, Ice, and Land
UART	Universal Asynchronous Receiver Transmitter
UC	Unknown Class
USB	Universal Serial Bus
VA	Validation Accuracy
VHDL	Very High-Speed Integrated Circuit Hardware Description Language
XSG	Xilinx System Generator

CHAPTER 1

INTRODUCTION

1.1 Spectral Imaging

Optical images are the prime sources of information in remote sensing data analysis. Since their inception, they have played a vital role in the success of various information mining and image interpretation tasks since their inception (Prasad, Bruce and Chanussot, 2011). Contemporary examples include, but are not limited to, mapping (or classification) and monitoring applications of land use/land cover (LULC) change (Jun and Ghosh, 2013), health care (Halicek *et al.*, 2017), weather and climate forecasts (Piñeros, Ritchie and Tyo, 2011), and Planetary exploration missions (Clark *et al.*, 2003). Imaging systems use sensors, which are sensitive to one or more wavelengths of an electromagnetic spectrum, to capture images for the visual representation of an area. Depending on the imaging techniques used, the visual representation and structure of an image can take many forms to convey different levels of information of the desired scanning scene based on the surface reflectance, as shown in Figure 1.1. Some optical imaging techniques used in modern times include panchromatic imaging, red, green, and blue (RGB) imaging, spectral imaging, thermal imaging, etc. Spectral imaging remote sensing combines two technologies, namely imaging and spectroscopy. Spectral imaging for remote sensing can be divided into two main techniques based on the continuity of the data stored in the wavelength domain, multispectral imaging and hyperspectral imaging (Eismann, 2012).

In the past few years, spectral imaging systems have gained significant attention from researchers across various scientific and engineering disciplines (Bioucas-Dias *et al.*, 2013). Unlike traditional panchromatic and RGB imaging systems, spectral imaging systems use specialized sensors operating primarily from the visible through infrared

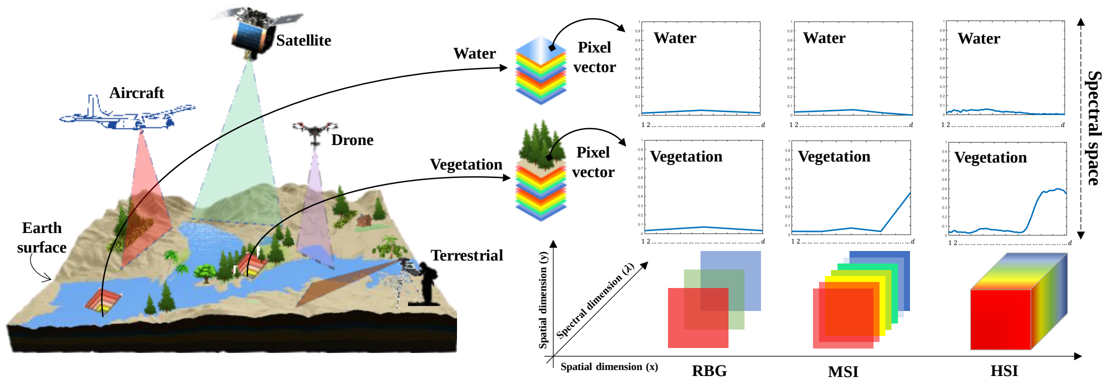


Figure 1.1: The concept of different types of optical imaging platforms, different types of images recorded by hyperspectral, multispectral, and RGB sensors. The spectral signatures of water and vegetation are also shown to compare the difference in the spectral information.

wavelength ranges to acquire images with more than three spectral bands (see Figure 1.1) (Khan *et al.*, 2018). Spectral imaging is the technique of producing images, such as multispectral images (MSIs) and hyperspectral images (HSIs), containing both spatial and spectral domain information of scanning area on the surface of the earth based on surface reflectance. Both MSIs and HSIs are three-dimensional data structures having two spatial dimensions and one spectral dimension. The main difference between MSI and HSI is in the usage of the sampling technique, the bandwidth of each spectral channel, and the number of bands (Eismann, 2012). That is to say, multispectral remote sensing collects MSI data that have several bands each sampled at discrete, often discontinuous, wavelengths with wider spectral bandwidths, i.e., low spectral resolution. While hyperspectral remote sensing collects HSI data having spectral bands, each sampled at contiguous wavelengths with narrow bandwidths (typically in 10nm or less), i.e., high spectral resolution (Prasad, Bruce and Chanussot, 2011).

In contrast to grayscale and color images, MSIs and HSIs provide the benefit of analyzing images in a pixel-by-pixel fashion accurately using spectral information alone (Chang, 2003). This benefit is due to the increasing order of pixel-wise spectral information from panchromatic, RBG, MSI, and HSI, as shown in Figure 1.1 (Khan *et al.*, 2018).

Thanks to the inherent high spectral information present in HSIs, accurate discrimination of materials is possible compared with MSIs. However, each imaging mode has its own advantages and disadvantages based on selected image analysis techniques and applications.

1.1.1 Image processing and analysis techniques

In optical remote sensing, imaging systems use passive sensors to record the incoming radiation in its instantaneous field of view for every given pixel includes not only the reflected or emitted radiation but also the scattered radiation. Compared to terrestrial or ground-based imaging, airborne and satellite imaging are mainly influenced by atmospheric scattering and thus, the images recorded using these platforms require calibration and correction processes to remove the effects of the atmosphere. So, this is one of the required preprocessing steps to reconstruct the surface reflectance values. After calibration and correction for sensor, atmosphere, and topographic effects, these preprocessed remotely sensed images can then be used for image processing and analysis tasks (Eismann, 2012).

A typical objective of MSI/HSI processing and analysis in different fields of application includes one or more of the techniques such as dimensionality reduction, target detection, and classification (Chang, 2003; Dubacharla and Nidamanuri, 2020). Dimensionality reduction refers to the process of reducing the number of attributes or features either by using feature selection or by feature extraction methods (Damodaran and Nidamanuri, 2014). Target detection refers to the process of identifying a target object from the background of the observed scene of the study (Bernabe *et al.*, 2011). Classification is a process of labeling every pixel or an object in an image using either a supervised or unsupervised approach. Among the image processing and analysis techniques, classification is one of the prominent techniques of analysis to mine the information present in images and a particularly active and vibrant area of research in remote sensing (Ghamisi *et al.*, 2017). This prominence is because the classification technique helps transform the large volumes of remotely sensed image data into useful information with multiple usages.

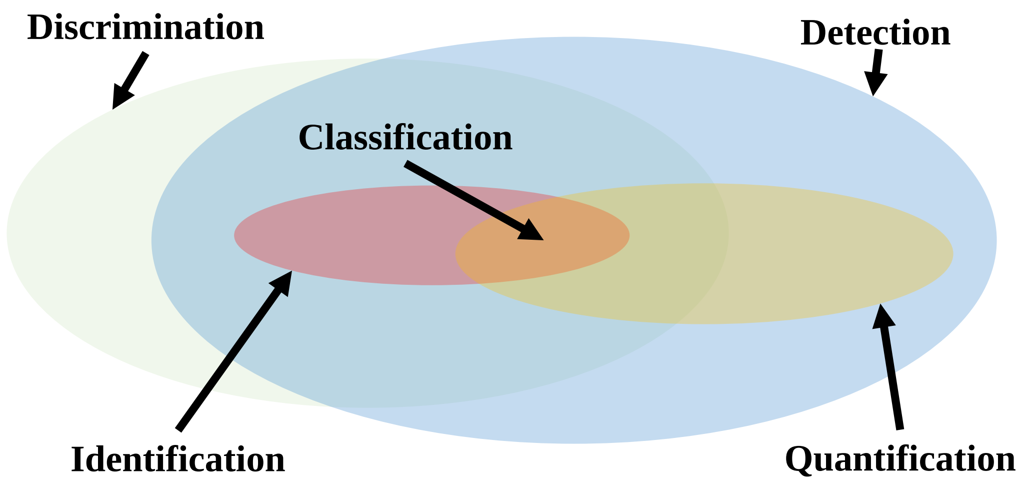


Figure 1.2: Venn diagram to illustrate the advantage of classification technique for including the subtle functionalities of discrimination, detection, and quantification.

Classification also includes subtle functionalities of detection, discrimination, identification, and quantification, as shown in Figure 1.2 (Chang, 2003). There are three different categories of classification approaches: pixel-level classification, object-level classification, and sub-pixel classification or unmixing (Bioucas-Dias *et al.*, 2013). The choice of approach depends mainly on the application requirement since they offer both advantages and limitations.

1.1.2 Applications

In the past, the challenge with remote sensing imagery was that data was costly, having low or medium resolutions, and simply not frequent enough to perform timely image analysis, especially with satellite imagery. The current generation of remote sensors is of very high and ultra-high spatial, spectral resolution imaging sensors with low operational costs and high revisit rates. These advancements have not only played a crucial role in achieving success in a wide variety of remote sensing applications but also have enabled the exploration of new applications (Behmann *et al.*, 2018).

For decades, spectral imagers have been crucial components of remote sensing and have significant footprints in the area of life sciences. The number of remote sensing applications using MSIs and HSIs continues to grow across various fields. The following are some examples of the applications where remotely sensed MSIs and HSIs are progressively used.

- ***Environmental monitoring and management***: MSIs and HSIs can be used to study the nature of the environment and track changes over time. For instance, they are used to map and monitor coastal and marine environments (Salem and Kafatos, 2001), air pollution, water quality, LULC classes, forest management, urban human settlements, alpine snow properties, and detection of hazardous materials (Dalponte *et al.*, 2009).
- ***Geology and mineralogy***: Since the introduction of MSIs and HSIs, geologists have been using the data for lithology, mapping various mineral resources and ore deposits (Clark *et al.*, 2003; Adep *et al.*, 2017).
- ***Precision agriculture***: MSIs and HSIs are very effective tools for monitoring crop health and evaluating soil productivity for site-specific applications of inputs tailored to the needs of the crop, such as water, pesticides, and fertilizers. They also help to detect pathogens and pests associated with crop production, such as weeds, insects, and diseases (Thenkabail and Lyon, 2016; Saari *et al.*, 2017).
- ***Health care***: Hyperspectral imaging has been recently used in health care to conduct non-invasive scans of exact areas of skin to detect diseased or malignant cells for expedited diagnosis even intraoperatively and rapid treatment of the exact regions affected (Madroñal *et al.*, 2017; Khan *et al.*, 2018).
- ***Military, defense, and homeland security***: MSIs and HSIs can be used for tactical reconnaissance, counter-countermeasure for detection, tracking, and classification of targets, including targets in varying degrees of hiding and camouflage (Manolakis, Marden and Shaw, 2003; Makki *et al.*, 2017).
- ***Industrial***: MSIs and HSIs can be used for automated quality inspection, manufacturing process control, and monitoring in various industrial sectors like

food processing, pharmaceutical, and sorting products (Gowen *et al.*, 2007; Sun, 2010).

Including the aforementioned applications, and many remote sensing applications benefit from using HSIs because they help to provide accurate detection and classification of materials relative to the use of MSIs (wider bandwidths). This benefit is due to the ability of HSIs to have subtle differences in signal along a continuous spectrum. However, some fixed or static environment applications require only a significant portion of the electromagnetic spectrum for analysis, and a band selective capturing technique such as multispectral imaging can be very well suitable. In such cases, hyperspectral imaging can be used as a reference for the initial evaluation of bands and later can make selective band capturing (custom made) for specific applications. This will also reduce the time for image capturing, processing, and analysis of the HSIs for high-end, high-precision applications.

1.1.3 Challenges

The evolution of very high-resolution MSIs and HSIs has proved to be particularly advantageous for most remote sensing image interpretation tasks. But there are challenges associated with the use of MSIs and HSIs for transforming the rich spectral data into information for applications. The most common limiting factors known to increase the risk in analyzing MSIs and HSIs are briefly mentioned below.

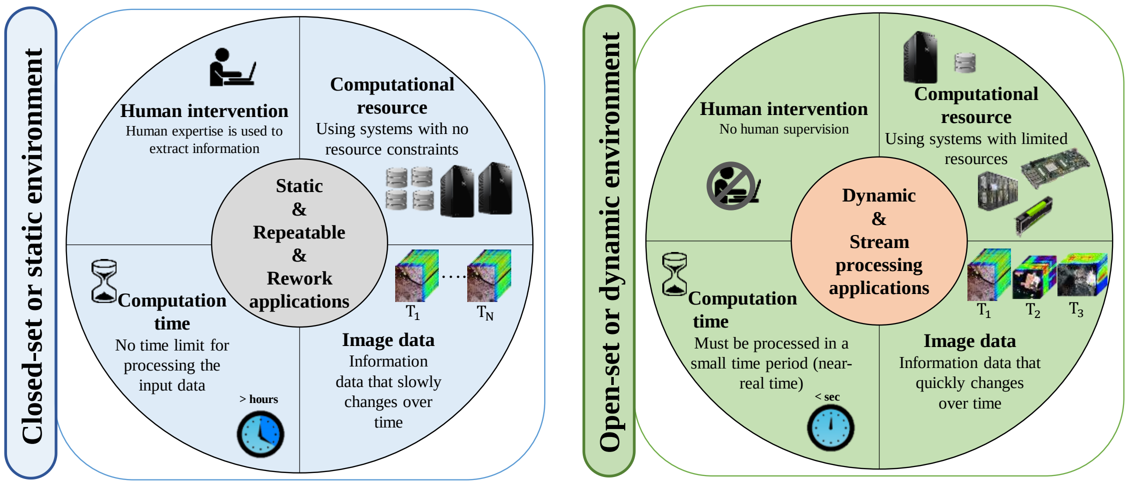
- ***Class separability***: Due to the low spectral resolution, the separability of classes of interest (say within classes and between classes) is limited with MSIs. Whereas with HSIs, class separability is largely owing to the high spectral resolution. However, the increase in the dimensionality of the data increases the possibility of problems with the curse of dimensionality.
- ***Limited labeled samples***: In practice, it is difficult or not possible to collect complete information about the training samples of classes of interest. Only a limited number of labeled samples of classes of interest are only available at hand before training.

- ***Uncertainties***: Due to the stochastic nature of landscapes, there are different types of uncertainties ranging from image acquisition to image processing and analysis, including training and classification. Some of the uncertainties are sensor noise, limited training knowledge of classes of interest referred to as known classes (KCs), incomplete knowledge about the actual number of classes in the imagery, i.e., presence of unknown classes (UCs), which are unknown or unseen during training. These uncertainties are one of the significant factors responsible for omission and commission errors (Dubacharla and Nidamanuri, 2020).
- ***Computation time***: Time-critical and real-time applications such as target detection, diurnal change detection, industrial quality inspection require real or near real-time computation solutions and classification frameworks (Bernabe *et al.*, 2011). In particular, there are a plethora of applications, which bank upon MSIs and HSIs, needing real-time processing systems for image analysis.

The abovementioned complexities and challenges in using these images can slow adoption in some applications. Therefore, the development of novel algorithms is needed to exploit the full latent potential of spectral information present in MSIs and HSIs.

1.2 Real-world, Real-time Environments

Till now, the discussion was focused on the challenges faced in the usage of HSIs and MSIs. But we also need to consider the computation challenges and complexities associated with algorithms when deployed in real-world, real-time application environments (González *et al.*, 2013; Dubacharla and Nidamanuri, 2020). A real-world environment can be defined as an operational environment where practical or actual instances exist, unlike simulated or laboratory settings. A real-time environment can be defined as an application environment where a strictly limited time frame is assigned to process the current inputs before it disappears or a new one arrives. For example, the airborne visible infrared imaging spectrometer (AVIRIS) instrument data rate is around 20.4 Mbps with a real-time specification of 131.7 μs for processing each pixel vector of length 224 bands with 12-bit encoding. Then an image analysis system, say performing



Closed-set or static environment
Human intervention
Human expertise is used to extract information
Computational resource
Using systems with no resource constraints
Static & Repeatable & Rework applications
Computation time
No time limit for processing the input data
> hours
Image data
Information data that slowly changes over time
T1
TN
Open-set or dynamic environment
Human intervention
No human supervision
Computational resource
Using systems with limited resources
Dynamic & Stream processing applications
Computation time
Must be processed in a small time period (near-real time)
< sec
Image data
Information data that quickly changes over time
T1
T2
T3

Figure 1.3: The illustration of an abstract view of closed-set or static environments and open-set or dynamic environments. Each environment has four partitions: human intervention, computational resource, computation time, and data, to show the prime differences.

classification, is said to the real-time system if it processes each pixel vector in less than 131.7 μs. As another example, the next-generation sensor of AVIRIS, i.e., AVIRIS-NG, has a real-time constraint of 15.6 μs for every pixel vector of 427 channels with 16-bit encoding (Basterretxea *et al.*, 2016; Dubacharla and Nidamanuri, 2020).

In this section, the focus will be directed towards the critical challenges faced from an algorithmic and application environment perspective. In general, the final objective of any research is to deploy the developed system or method in real-time environments and operate in accordance with the specifications for the specified application. As mentioned earlier, the real-time environment is the term used to describe the application environment which hosts the system or algorithm during various stages of its life cycle. Based on the rate of data change in the test site, an application environment can be grouped into one of two main disjoint categories, namely static environment and dynamic environment (Chang, 2016), as shown in Figure 1.3.

An environment is considered static or fixed if the nature of information classes in the scene observed is repeatable or does not change over time, and there are no constraints of turnaround time for image analysis (see Figure 1.3). Whereas in dynamic or non-stationary environments, the nature of information classes may change rapidly over time, and there is a short turnaround time available to process the current data before new changes come into effect (see Figure 1.3). These short turnaround times for processing the inputs are referred to as timing constraints and are application-specific. In addition to the timing constraints, one also has to deal with uncertainties present in dynamic environments. For instance, uncertainties in MSI/HSI classification tasks include noise, incomplete prior knowledge about the information classes and spectral classes, etc. A real-world application environment is often dynamic in nature and sometimes a combination of both static and dynamic settings. Therefore, algorithms capable of producing deterministic outputs with low-latency performance are essential for demanding real-time applications operating in real-world, real-time environments.

Due to the stochastic nature of real-time environments, apart from image processing and analysis in real-time or near-real time constraints, various types of uncertainties exist at different stages that may affect performance. Examples of uncertainties in real-world, real-time environments include open-set problems, shadows, and illumination effects, etc.

1.3 Aim, Motivation, and Objectives

This thesis aims to design robust techniques for image classification of MSIs/HSIs that minimizes runtime and training data dependence to offer stable performance in real-time environments. To satisfy the ever-changing application requirements in a highly variable environment, the development of efficient and robust classification algorithms is crucial. This reason has motivated us to define the scope of our research, and we set our goal to develop necessary MSI/HSI analysis and pattern recognition techniques for setting up a low-cost, portable custom-made processing system for enabling real-time operational image analysis.

A series of four specific objectives, listed below, is proposed to systematically develop image classification algorithms for real-time and operational environments.

1. To design and develop a scalable FPGA-based architecture for real-time classification using hyperspectral imagery.
2. To develop a new generic algorithmic framework for addressing the open-set classification problems associated with the remote sensing image analysis in operational application environments.
3. To construct a novel algorithmic scheme for mitigating shadow-related classification problems associated with the remote sensing image analysis in operational environments.
4. To realize the real-time processing of open-set image classification for multi-resolution remote sensing imagery.

1.4 Structure of Thesis

The remainder of this thesis is organized in such a way that the specific objectives listed above are presented in different chapters. Each chapter is made self-contained and includes an introduction, research problem statement, data used, methodology, results and analysis, discussion, and conclusions.

- Chapter 2 presents a brief background knowledge of the data intake procedure for MSI/HSI classification tasks, real-time processing, and different classification techniques used in later chapters. This chapter also provides an extensive and systematic survey of the literature on the existing image classification methods from algorithmic and computational perspectives and discusses the problems and current solutions.
- Chapter 3 proposes a new low-complexity design approach for realizing real-time HSI classification using FPGA as a hardware accelerator. The proposed computationally efficient FPGA design and implementation of HSI classification were evaluated in terms of classification accuracy, throughput, latency, computation time, and power usage. Experiments were carried out on four different multi-sensor and platform HSIs with changing dimensionality and real-time constraints.

- Chapter 4 presents a novel supervised image classification technique specifically proposed to provide stable and robust performance even in the presence of UCs in any given application environment. Assessment using exclusively designed case studies comprising various classification scenarios of closed-set and open-set problems is done to test the stability and robustness of classifiers. The case studies are created using four different remote sensing imageries with different LULC settings. The classification performance of the proposed method was compared against other state-of-the-art closed-set and open-set methods using several accuracy metrics and thematic maps.
- Chapter 5 introduces a set of new shadow and illumination invariant classification algorithms using a supervised approach. The proposed algorithms use spectral information from MSIs/HSIs to perform class predictions even in the presence of shadows and varying illumination conditions. Experiments were carried on different classification scenarios of varying shadow and illumination complexities designed using six different MSI and HSIs. Further, a comprehensive analysis of the obtained classification results of the methods considered in this experiment is also presented in this chapter.
- Chapter 6 presents and discusses a set of novel supervised classification approaches for real-time environments using different MSIs and HSIs. The computational efficiency of the proposed algorithms was demonstrated using software and FPGA-based hardware design implementation. In this study, SVM and deep convolutional neural networks (DCNN) were used as supervised classifiers in the second stage of the proposed methods. A comprehensive evaluation of the methods was performed using three different case studies of various classification scenarios of open-set and closed-set settings. And a detailed analysis of the results obtained in both software and hardware implementation is presented in this chapter.
- Chapter 7 provides a comprehensive discussion on the overall research findings of the studies conducted in the previous chapters and provides a detailed summary of the main contributions of this thesis to the community.

- Chapter 8 presents a summary of the overall conclusions of the previous chapters, recommendations and future directions of this thesis, and acknowledgments for the multispectral and hyperspectral data.

CHAPTER 2

THEORETICAL BACKGROUND AND LITERATURE REVIEW

Prelude: In this chapter, the theoretical background and mathematical foundation of data intake processing, supervised classification methods, real-time processing, and FPGA design of MSIs/HSIs for pattern recognition and image classification tasks, which are used in the later chapters of this thesis, are briefly described. It also includes a fairly detailed description of the stages involved in performing the supervised learning approaches for image classification, such as model training and classification. Further, a comprehensive review of the related works on the various aspects of existing methods in the literature to solve the stated objectives are presented.

2.1 Data Intake Processing

In this thesis, we consider HSIs and MSIs as the choice of input remote sensing imagery data for image classification tasks. Notationally, let us consider a 3D MSI/HSI data cube denoted by $\mathbf{X} \in \mathbb{R}^{m\times n\times d}$ has m rows or lines, n columns or samples, and d spectral bands or dimensionality. For convenience, the 3D data cube $\mathbf{X}$ of size $m \times n \times d$ can be written into a 2D $d \times N$ (where $N = m \times n$) matrix form, say $\boldsymbol{X} \in \mathbb{R}^{d\times N}$, known as data matrix as shown in Equation (2.1) (Eismann, 2012). The data matrix $\boldsymbol{X}$, is typically defined as the rastered ordering of N column pixel vectors $\mathbf{x}_i \in \mathbb{R}^d$, such that $\boldsymbol{X} = [\mathbf{x}_1\ \mathbf{x}_2\ ...\ \mathbf{x}_N] \in \mathbb{R}^{d\times N}$. Where each pixel vector $\mathbf{x}_i$, see Equation (2.2), represents a spectral measurement and N is the total number of pixel vectors, i.e., N is product of m and n.

$$\boldsymbol{X} = [\mathbf{x}_1\ \mathbf{x}_2\ ...\ \mathbf{x}_N]_{d\times N} = \begin{bmatrix} x_{11} & x_{21} & & x_{N1} \\ x_{12} & x_{22} & \dots & x_{N2} \\ \vdots & \vdots & & \vdots \\ x_{1d} & x_{2d} & & x_{Nd} \end{bmatrix}_{d\times N} = \begin{bmatrix} x_{11} & \cdots & x_{N1} \\ x_{12} & \cdots & x_{N2} \\ \vdots & \ddots & \vdots \\ x_{1d} & \cdots & x_{Nd} \end{bmatrix}_{d\times N} \in \mathbb{R}^{d\times N} \qquad (2.1)$$

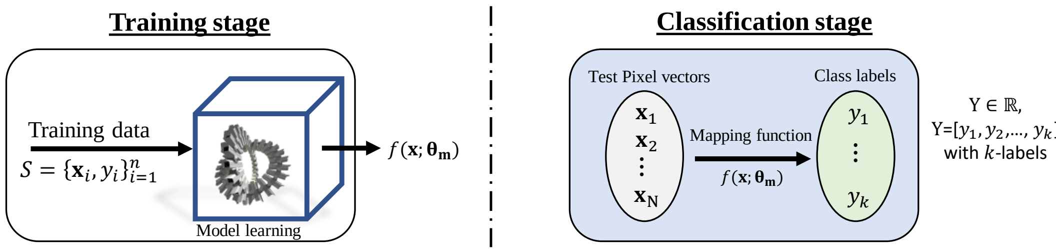

Training stage
Training data
$S = \{\mathbf{x}_i, y_i\}_{i=1}^{n}$
Model learning
$f(\mathbf{x}; \boldsymbol{\theta}_\mathbf{m})$
Classification stage
Test Pixel vectors
$\mathbf{x}_1$ $\mathbf{x}_2$ ⋮ $\mathbf{x}_N$
Mapping function
$f(\mathbf{x}; \boldsymbol{\theta}_\mathbf{m})$
Class labels
y_1 ⋮ y_k
Y ∈ ℝ,
Y=[$y_1, y_2,\ldots, y_k$]
with k-labels

Figure 2.1: Illustrating the overview of training stage and classification stage in supervised learning approach.

$$\mathbf{x}_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{id} \end{bmatrix}_{d\times 1} \in \mathbb{R}^d \tag{2.2}$$

For a pixel-by-pixel-based image analysis, each d dimensional pixel vector is given as input to the algorithm as one-by-one in a streaming fashion. For instance, assume that the MSI/HSI arrives in a pixel-by-pixel manner from a data matrix or in band-interleaved-by-pixel (BIP) format from a sensor. As a result, a pixel vector-based analysis can be performed. In other two image data formats, such as band-interleaved-by-line (BIL) and band sequential (BSQ), the 3D data cube is transformed to a 2D data matrix and then streamed pixel-wise to the classification algorithm.

2.2 Supervised Image Classification

Supervised learning is one of the most widely researched approaches in the image classification domain. This is because of its ability to provide accurate predictions for a given training set. In supervised classification, the user or image analyst offers a representative set of labeled training samples containing class-specific information for learning the classification model (Ghamisi *et al.*, 2017). The process of learning a classification model is known as model training or simply training. Then, the trained classification model is used for class predictions. The key idea in supervised classification can be summarized as the task of learning a mapping function f that maps inputs, say $\mathbf{x}$, to a unique label from a prefixed set of discrete outputs labels, categories or target, say y.

Mathematically, we can express the supervised classification as $f\colon \mathbf{x} \rightarrow y$ or $y = f(\mathbf{x})$ as shown in Figure 2.1.

Based on the assumptions made on the mapping function, supervised learning approaches can be split into parametric and non-parametric. The parametric techniques are more suited for classification tasks with few training inputs that have lower dimensions. So, parametric approaches would be inefficient compared to non-parametric approaches for performing supervised MSI/HSI classification tasks. Unlike parametric methods, non-parametric methods do not make any assumptions about the mapping function and are free to learn any functional form from the training set (Duda and Hart, 1973). Some of the popular non-parametric methods from the machine learning paradigm are neural networks (NNs), support vector machines (SVMs), random forests (RFs), deep learning (DL), etc. As shown in Figure 2.1, there are generally two phases or stages involved in using any supervised classification technique, namely the training phase and the classification phase (Dubacharla and Nidamanuri, 2020). A brief about the two phases in supervised image classification is given as follows.

2.2.1 Training and classification

In the training stage of the supervised learning process, a discriminant function f is formulated using a training set S consisting of l labeled pixel vectors, i.e., $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ with training samples ($\mathbf{x}_i \in \mathbb{R}^d$), and the corresponding class labels ($y_i \in \mathbb{R}^l$) as shown in Figure 2.1. To produce a generalized supervised classification model f using S, a grid search method with a cross-validation (CV) strategy is adopted to find the optimal values of model parameters and free parameters. During training, an algorithm-specific free parameter $\boldsymbol{\theta}_{\mathbf{f}} = \{\theta'_{f1}, \theta'_{f2}, \dots, \theta'_{fi}\}$ are given as input to the training model to produce optimal model parameters $\boldsymbol{\theta}_{\mathbf{m}} = \{\theta'_{m1}, \theta'_{m2}, \dots, \theta'_{mj}\}$ to make the model fit the data. The model parameters are used to define the f and are dependent on free parameters. Based on the nature of each free parameter, the supervisor assigns a fixed range of values or defines the parameter space for each free parameter. The available labeled reference training data

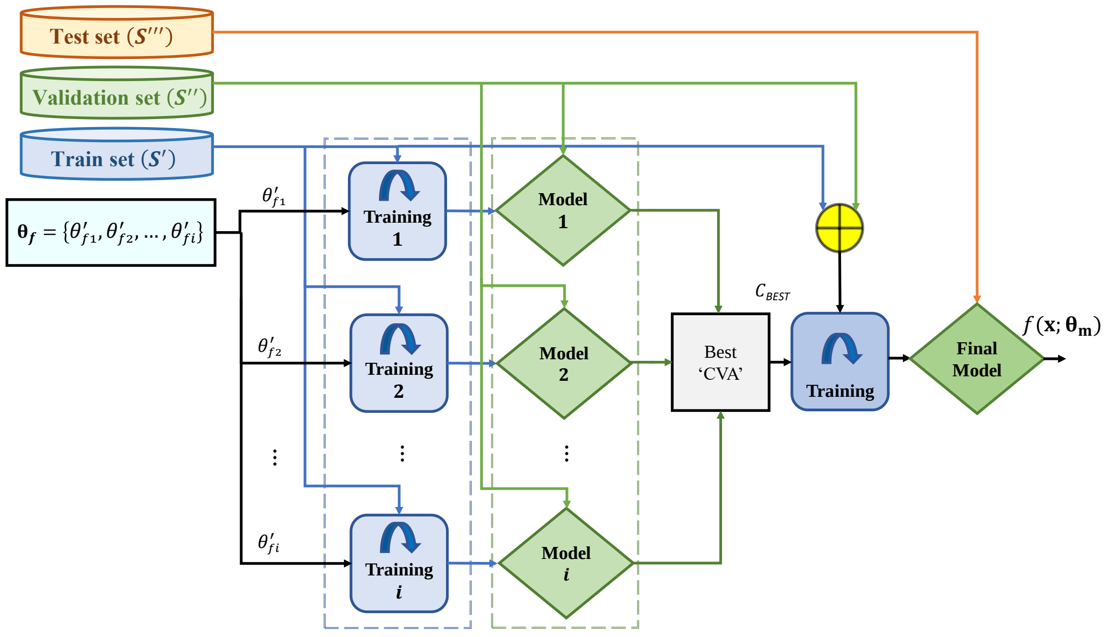


Figure 2.2: Detailed process of training stage in supervised learning approach.

S is first split into three different sample subsets, namely train set (say $S' \subseteq S$), validation set (say $S'' \subseteq S$), and test (say $S''' \subseteq S$) as shown in Figure 2.2 (Dubacharla and Nidamanuri, 2020). In this thesis, the splitting or sampling procedure of reference data S is performed in two steps. First, S is split into two folds of sample data using the hold-out method, i.e., $S^{\perp}$ and S'''. Second, the sample $S^{\perp}$ is again partitioned into two groups, i.e., S' and S'' of K-folds using K-fold technique for CV. For each unique combination of input free parameter values out of L total combinations, a model trained using one of K-fold sample S'. Then, an unbiased evaluation of the trained model is performed using S'' to provide validation accuracy (VA) score, and the value is preserved for future reference. This process is repeated for K times to produce K trained classification models and VA scores. Then for each of L, a K-fold cross VA (CVA) is calculated using

$$\mathrm{CVA}_L = mean\{[VA_{cv}]_{cv=1}^{K}\} \tag{2.3}$$

The optimal combination of best free parameters is selected for the highest CVA or best CVA from L CVA values. Till now, the procedure is done for tuning or configure the optimal free parameters. Next, a final model fit on the dataset $S^{\perp}$ and the tuned free parameters. Finally, the unseen sample of test data S''' is used to provide an unbiased evaluation of the final model in terms of test accuracy. The above-described training and testing procedures for a given supervised algorithm are also illustrated in Figure 2.2. In the second phase of the supervised approach, the trained classification model (i.e., $f(\mathbf{x};\boldsymbol{\theta_m})$) is used for class prediction on test pixel vectors as shown in Figure 2.2.

2.3 Supervised Image Classification Methods

A brief description of different types of supervised image classification techniques which are used in this thesis is provided as follows.

2.3.1 One-class classification (OCC) methods

Unlike binary classification, OCC builds a classification model by learning from a training dataset containing only the samples from one class, i.e., the target class. The objective of a trained OCC model is to label the test samples as either the target or the outlier class (Mũnoz-Marí *et al.*, 2010). For a given test pixel vector $\mathbf{z}$, the output of trained OCC model $f_{occ}(\mathbf{z}, \boldsymbol{\theta}_{\mathrm{m}}) \in \{-1, +1\}$ where -1 represents a non-target class label and $+1$ represents a target class label. Other multiple names in use for target class in OCC include positive class, normal class, inliers, in-distribution (ID), and KC. In the case of non-target class, the multiple identities include negative class, abnormal class, outliers, out-of-distribution (OoD), and UC. The supervised OCC methods consist of one-class SVM (OCSVM) and support vector data description (SVDD).

2.3.1.1 One-class support vector machine (OCSVM)

OCSVM is one of the widely used supervised OCC techniques to characterize a single class that is well described by the training dataset containing only the target class examples (Schölkopf *et al.*, 2000). It is also commonly used for anomaly or novelty detection tasks to identify the abnormal or outlier events that fail to match the target description. The OCSVM is a particular form of the conventional binary class SVM, where an optimal hyperplane with maximum margin is built between the target samples and the origin

(Mũnoz-Marí *et al.*, 2010). In OCSVM training, the origin is considered as the only available non-target sample. Similar to SVM, a hyperplane is determined by a small set of training samples called support vectors and non-zero slack variables ξ_i are introduced to allow few margin errors in the training set. The objective function of the OCSVM classifier is the following minimization problem

$$\min_{\mathbf{w},\rho,\xi}\left(\frac{1}{2}\|\mathbf{w}\|^2+\frac{1}{\nu n}\sum_i \xi_i-\rho\right) \tag{2.4}$$

$$\text{subject to} \quad \mathbf{w}^T\phi(\mathbf{x}_i)\geq \rho-\xi_i, \forall i=1,\dots,n$$
$$\xi_i\geq 0$$

where $\mathbf{w}$ is the normal vector of the hyperplane, ρ is the bias, $\nu\in(0,1]$ is a hyperparameter that decides the upper bound on the fraction of outliers in S, $\mathbf{x}_i$ are the data points, and $\phi(.)$ is a transformation function from the input space to the feature space $\mathcal{F}$. The above minimization problem can be solved using Lagrange multipliers $\alpha_i>0$ as a dual optimization leading to quadratic programming (QP) solution. Then the decision on the test sample $\mathbf{z}=[z_1,\dots,z_d]^T\in\mathbb{R}^d$ is evaluated using the OCSVM discriminant function f_{OS} as

$$f_{\mathrm{oc}}(\mathbf{z};\boldsymbol{\theta}_{\mathrm{m}})\equiv f_{\mathrm{os}}(\mathbf{z};\mathbf{x}^{\dagger},\alpha,\rho)=\mathrm{sign}\left(\left(\sum_{i=1}^{\mathrm{K}}\alpha_i K\left(\mathbf{x}_i^{\dagger},\mathbf{z}\right)\right)-\rho\right) \tag{2.5}$$

$$K(\mathbf{x},\mathbf{z})=\phi(\mathbf{x})^T\phi(\mathbf{z})=\mathbf{x}^T\mathbf{z} \tag{2.6}$$

$$K(\mathbf{x},\mathbf{z})=\phi(\mathbf{x})^T\phi(\mathbf{z})=\exp(-\gamma\|\mathbf{x}-\mathbf{z}\|^2) \tag{2.7}$$

where K denotes the number of support vectors, $\mathbf{x}_i^{\dagger}$ denotes the support vectors, γ is the width of the Gaussian curve, and the kernel function K is defined by $K(\mathbf{x},\mathbf{z})=\phi(\mathbf{x})^T\phi(\mathbf{z})$ which is referred to as the dot product in the feature space.

2.3.1.2 Support vector data description (SVDD)

SVDD is another particular variant of conventional two-class SVM aiming to formulate an optimal hypersphere by primarily learning from S containing only target data. The samples within the hypersphere are examples of the target, the non-target samples are outside of it

(Tax and Duin, 1999; Khazai *et al.*, 2012). The model of SVDD can be formulated by solving the following optimization problem:

$$\min_{R,\xi,\mathbf{a}} \left(R^2 + C \sum_i \xi_i \right) \tag{2.8}$$

$$\text{subject to} \quad \|\phi(\mathbf{x}_i) - \mathbf{a}\|^2 \leq R^2 + \xi_{i_1}, \forall i = 1, \dots, n$$

$$\xi_i \geq 0$$

After solving the above minimization problem through α_i, an unknown test pixel vector **z** can be predicted using the SVDD decision function f_{SD} of the following form

$$\begin{aligned} f_{OC}(\mathbf{z}; \boldsymbol{\theta}_\mathbf{m}) \equiv f_{SD}(\mathbf{z}; \mathbf{x}^\dagger, \alpha, R, \mathbf{a}) &= sign(R^2 - \|\phi(\mathbf{z}) - \mathbf{a}\|^2) \\ &= sign\left(R^2 - K(\mathbf{z},\mathbf{z}) + 2\sum_{i=1}^{\mathrm{K}} \alpha_i K\left(\mathbf{z}, \mathbf{x}_i^\dagger\right) \right. \\ &\quad \left. - \sum_{i=1}^{\mathrm{K}} \sum_{j=1}^{\mathrm{K}} \alpha_i \alpha_j K\left(\mathbf{x}_i^\dagger, \mathbf{x}_j^\dagger\right) \right) \end{aligned} \tag{2.9}$$

Where α_i, $\alpha_j \geq 0$ are the Lagrange multipliers, $\mathbf{x}_i^\dagger$, $\mathbf{x}_j^\dagger$ denotes a set of K support vectors, R denotes the radius of the hypersphere that encloses the whole training data, and **a** it's center.

2.3.2 Similarity matching methods

The spectral similarity matching methods (SMMs) consist of spectral information divergence (SID) and spectral correlation mapper (SCM).

2.3.2.1 Spectral information divergence (SID)

SID is a stochastic measure derived from information theory. It is used for measuring spectral similarity and discriminability between unknown test pixel spectra and target reference spectra (Chein-I Chang, 1999). SID views each pixel vector as a random variable and calculates the probabilistic behaviors between the reference pixel vector **x** and test pixel vector **z**. The probability vectors of **x** and **z** are given by $\boldsymbol{p} = \{p_i\}_{i=1}^{d}$ and $\boldsymbol{q} = \{q_i\}_{i=1}^{d}$, respectively, where p_i and q_i are given as follows:

$$p_i = \frac{x_i}{\sum_{k=1}^{d} x_k}, \qquad q_i = \frac{z_i}{\sum_{k=1}^{d} z_k} \tag{2.10}$$

The SID, ψ_{SID}, between **x** and $\boldsymbol{z}$ is given by

$$\psi_{SID}(\mathbf{z}, \mathbf{x}_i) = D(\mathbf{x}_i \parallel \mathbf{z}) + D(\mathbf{z} \parallel \mathbf{x}_i) \tag{2.11}$$

$$D(\mathbf{x}_i \parallel \mathbf{z}) = \sum_{l=1}^{d} p_l \, D_l(\mathbf{x}_i \parallel \mathbf{z}) \quad = \sum_{l=1}^{d} p_l \log_2\left(\frac{p_l}{q_l}\right) \tag{2.12}$$

$$D(\mathbf{z} \parallel \mathbf{x}_i) = \sum_{l=1}^{d} q_l \, D_l(\mathbf{z} \parallel \mathbf{x}_i) \quad = \sum_{i=1}^{d} q_i \log_2\left(\frac{q_l}{p_l}\right) \tag{2.13}$$

Where $D(\mathbf{x} \parallel \mathbf{z})$ and $D(\mathbf{z} \parallel \mathbf{x})$ are relative entropy of **x** with respect to **z** and **z** with respect to **x**, respectively, and are also known as Kullack-Leibler divergence or cross-entropy. The values of ψ_{SID} ranges from 0 to ∞. A value of $\psi_{SID} = 0$ indicates a perfect match with no divergence between the two spectra. The lower the ψ_{SID} value, the better the level of similarity between the reference spectra and unknown test spectra.

2.3.2.2 Spectral correlation mapper (SCM)

SCM is a derivative of Pearson's linear correlation coefficient. It is commonly used to measure the similarity between the reference pixel and unknown pixel (van der Meero and Bakker, 1997). The spectral matching performance of SCM is relatively higher than spectral angle mapper (SAM) because SCM can perceive a difference between positive and negative correlation. Unlike SAM, SCM is invariant to the linear transformation of spectra. The mathematical expression of SCM, ψ_{SCM}, for any **z** is given by

$$\psi_{SCM}(\mathbf{z}, \mathbf{x}) = \frac{\sum_{l=1}^{d}(z_l - \bar{\mathbf{z}})\,(x_l - \bar{\mathbf{x}})}{\sqrt[2]{[\sum_{l=1}^{d}(z_l - \bar{\mathbf{z}})^2][\sum_{i=1}^{d}(x_i - \bar{\mathbf{x}})^2]}}. \tag{2.14}$$

$$\bar{\mathbf{x}} = \frac{1}{d}\left(\sum_{l=1}^{d} x_l\right), \qquad \bar{\mathbf{z}} = \frac{1}{d}\left(\sum_{l=1}^{d} z_l\right) \tag{2.15}$$

Where $\bar{\mathbf{x}}$ and $\bar{\mathbf{z}}$ are the means of **x** and **z** pixel vectors, respectively. The values of the ψ_{SCM} ranges from -1 (i.e., no similarity) to $+1$ (i.e., perfect similarity).

2.3.3 Multi-class classification (MCC) methods without reject-option

Unlike OCC, multi-class classification (MCC) builds a classification model by learning from a training dataset containing samples from more than one class. The objective of an MCC model is to learn a classification model (f_{MC}) from S to label the unknown test samples as one of the labels from y. For a given test pixel vector $\mathbf{z}$, the output of the trained MCC model $f_{MC}(\mathbf{z}, \boldsymbol{\theta}_{\mathrm{m}})$ is an element of y. The assumption made by the classification algorithms in the MCC is that all classes are known during training, or the test classes and training classes are similar. This is a classic example of a closed-set recognition scenario where there is no consideration for unseen or UC samples. The supervised MCC methods without a reject option consist of a multi-class support vector machine (MSVM), random forest (RF), and a 1D deep convolution neural network (1D-DCNN).

2.3.3.1 Multi-class support vector machine (MSVM)

SVM is one of the most popular and widely used supervised algorithms for binary or two-class classification. It is a non-parametric method adopted from the machine learning paradigm and has been successfully applied to a plethora of pattern recognition problems across various real-world application domains. The objective of the SVM (binary case) is to find the maximum margin optimal hyperplane between the two classes i.e., $y \in \{-1, +1\}$ (Cortes and Vapnik, 1995). The main advantage of SVM is that the hyperplane is built using few training samples called support vectors. Moreover, a soft-margin solution in SVM accounts for non-separable data and noisy data by introducing ξ whose cost is controlled by tuning regularization parameter $C > 0$, which has the ability to generalize the classifier by relaxing the outliers in the S. The optimal hyperplane of SVM is obtained by solving the following primal optimization problem.

$$\begin{aligned} &\min_{\mathbf{w},b,\xi} L(\mathbf{w}, b) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i \\ &\text{subject to} \quad \mathrm{y}_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \forall i = 1, \dots, n \\ &\qquad\qquad\quad \xi_i \geq 0 \end{aligned} \tag{2.16}$$

where b represents the bias of the optimal separating hyperplane from the origin and L is a Lagrange function. The above primal minimization problem can be rewritten as a dual optimization through α_i leading to a QP solution. Then, the decision function of SVM f_{SVM} for any test vector $\mathbf{z} \in \mathbb{R}^d$ is given by

$$f_{SVM}(\mathbf{z}; \boldsymbol{\theta}_{\mathrm{m}}) \equiv f(\mathbf{z}; \mathbf{x}^{\dagger}, \alpha, b) = sign\left(\left(\sum_{i=1}^{\mathrm{K}} y_i \alpha_i K(\mathbf{x}_i^{\dagger}, \mathbf{z})\right) + b\right) \tag{2.17}$$

In many applications, it is effective and desired to classify more than two classes. In such cases, the idea of SVMs can be extended to solve MCC problems by constructing a group of binary classifiers (BCs). There are two different approaches for constructing MCC using BCs, such as the widely used one-against-one (OAO) technique known for its high reliability and accuracy and the one-against-all (OAA). The OAO method breaks down a k-class MCC problem into a P number of BCs ($\mathrm{P} > k, \forall\, k > 2$), where

$$\mathrm{P} = \binom{k}{2} = k_{C_2} = \frac{k(k-1)}{2} \tag{2.18}$$

Let $\{f_{SVM}^{1}, \dots, f_{SVM}^{\mathrm{P}}\}$ be the set of BCs in OAO and $f_{MSVM} \in \mathbb{R}^k, (k \in \mathbb{N})$ be the combinational function of P-set of BCs in OAO, where each classifier $f_{SVM}^{\mathrm{P}}(\mathbf{z}; \theta_{\mathrm{m}}^{\mathrm{P}}) \in \{-1, +1\}$ produces a binary decision. Subsequently, out of a total P number of decision values for a pixel vector, a class label is mapped to the pixel vector using the majority voting strategy by $f_{MSVM}(\mathbf{z})$. Mathematically, we can represent the decision function $f_{MSVM}(\mathbf{z}) \in y$ of MSVM using OAO technique for any test vector '$\mathbf{z}$' as

$$f_{MSVM}(\mathbf{z}; \boldsymbol{\theta}_{\mathrm{m}}) = majority\ vote\ \{f_{SVM}^{\mathrm{P}}(\mathbf{z}; \theta_{\mathrm{m}}^{i})\}_{i}^{\mathrm{P}}, \forall i = 1, \dots, \mathrm{P} \tag{2.19}$$

2.3.3.2 Random forest (RF)

RFs were first introduced in (Breiman, 2001) as an ensemble learning approach that is used for regression as well as classification. An ensemble approach is a technique aimed at improving accuracy by combining several models instead of a single model. In RF, several classifiers are trained to generate multiple hypotheses and their class predictions are then combined to produce a single decision through a non-trainable combination function like the majority voting rule (Ghamisi *et al.*, 2017). The main principle behind using an ensemble of methods is to encourage diversity among the classifiers to improve stability and prediction accuracy. Let $\{d_1(\mathbf{z}), \ldots, d_B(\mathbf{z})\}$ be the individual class predictions of B trees for the test pixel vector $\mathbf{z}$, then the decision function of RF is given by

$$f_{RF}^{B} = majority\ vote\{d_i(\mathbf{z})\}_1^B, \forall i = 1, \ldots, B. \\ f_{RF}^{B} \in y, d_i(\mathbf{z}) \in y \tag{2.20}$$

In general, RFs are considered to be a special case of decision trees but RFs grow many classification trees that are randomized to decorrelate their predictions. The techniques like bootstrap or bagging help to create training data by resampling the original data in a random fashion with replacement. This process of resampling training data makes RFs insensitive to the presence of noise in training data and prevents overfitting. There are two free parameters ($\boldsymbol{\theta}_f$) in the RFs training phase, namely the number of trees and the number of splits or prediction variables. These adjustable parameters can be tuned using grid search and cross-validation methods.

2.3.3.3 Deep convolution neural network (DCNN)

Deep learning (DL) techniques have seen a global trend towards skyrocketing increase in popularity and use across various scientific and engineering disciplines in recent years. The increasing popularity and usage of DL algorithms are mainly because of their ability to learn representative and discriminative features hierarchically employing various layers of abstraction. The word deep in DL usually means a depth of more than two hidden layers in a neural network (NN) architecture. There are different types of DL-based NNs depending upon the network architecture used, such as recurrent (cyclic) NN, convolutional NN, deep belief NNs, and more. However, deep convolution NNs (DCNNs) have attracted widespread attention in many vital applications. The DCNN architecture consists of various hidden components or layers like convolution layers, pooing layers, fully connected (dense) layers, and activation functions, as shown in Figure 2.3. A convolution layer produces feature maps using the following expression:

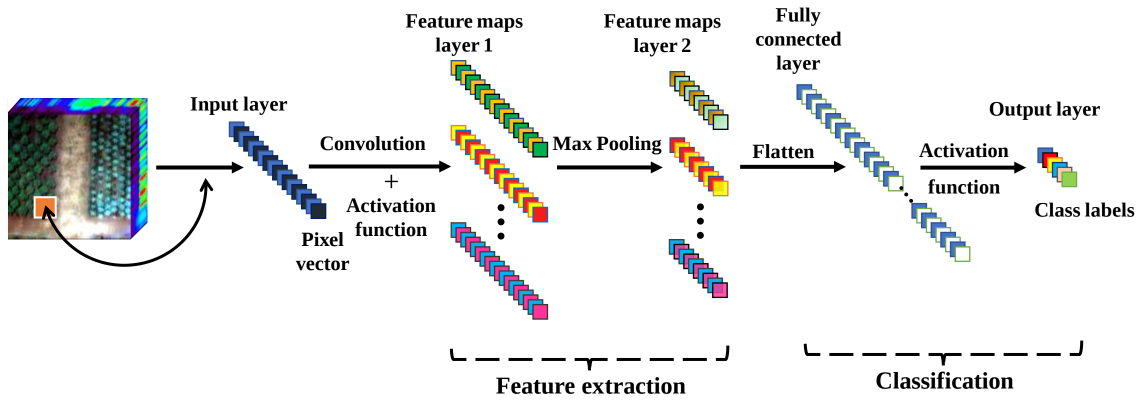


Figure 2.3: One-dimensional deep convolution neural network (ID-CNN) architecture.

$$\mathbf{x}_j^l = f\left(\left(\sum_i \mathbf{x}_i^{l-1} * \mathbf{k}_{ij}^l\right) + b_i^l\right), \forall i = 1, \dots, M^\dagger \tag{2.21}$$

where $\mathbf{x}_i^{l-1}$ is the i^{th} feature map of $(l-1)^{th}$ layer, $\mathbf{x}_j^l$ represents the j^{th} feature map of current i^{th} layer, $M^\dagger$ is the number of input feature maps, and $f(.)$ is a non-linear activation function. The trainable parameters $\mathbf{k}_{ij}^l$ and b_i^l represents the kernel or weights and bias in the convolution layer, respectively. The pooling layer is used to down sample the feature maps using either max or average rule. The fully connected layers stack the reduced features to perform classification.

2.3.4 Multi-class classification (MCC) methods with reject-option

The reject option-based MCC methods for open-set problems consist of open-set classifiers like multi-class pairwise SVM with Platt probability estimation (P-SVM) and two open-set nearest neighbor (OSNN) extensions: OSNN class verification (OSNN$^{\mathrm{CV}}$) and OSNN nearest neighbor distance ratio (OSNN$^{\mathrm{NNDR}}$).

2.3.4.1 Open set nearest neighbor cross verification (OSNN$^{\mathrm{CV}}$)

The OSNN$^{\mathrm{CV}}$ is a particular case of the nearest neighbor classifier, which aims at solving the multi-class open-set problems. It is based on the verification of class labels of the two nearest neighbors with respect to an unknown test pixel vector. The training stage of the OSNN^{CV} is similar to the nearest neighbor algorithm that stores all the available training samples S. In the classification stage, OSNN^{CV} finds two nearest neighbors from the test pixel vector $\mathbf{z}$ and labels it as unknown if both nearest neighbors have different labels; otherwise, the common label of the nearest neighbors is assigned to the test sample. Mathematically, this can be expressed as

$$f_{CV}(\mathbf{z}) = \begin{cases} y^{\dagger}, & if\ f(\boldsymbol{u}) = f(\mathbf{v}) \\ 0, & if f(\boldsymbol{u}) \neq f(\mathbf{v}) \end{cases} \tag{2.22}$$

Where class label 0 represents the class label for unseen class or UC samples, and $y^{\dagger} \subseteq y$ represents the label of the two nearest neighbors of **z,** i.e., $\mathbf{u}$ and $\mathbf{v}$.

2.3.4.2 Open set nearest neighbor based nearest neighbor distance ratio (OSNN^{NNDR})

Like OSNN^{CV}, the OSNN^{NNDR} classifier is another special variant of the nearest neighbor classifier for solving open-set problems. The open-set prediction performance of OSNN^{NNDR} is better than OSNN^{CV} because it considers the Euclidean distance between two different labeled nearest neighbors of test samples and the test sample. Unlike OSNN^{CV}, the training phase of OSNN^{NNDR} consists of tuning an optimum threshold from S for classification. For a given test sample $\mathbf{z}$, the OSNN^{NNDR} method obtains two nearest neighbors, i.e., $\mathbf{u}$ and $\mathbf{v}$, with different class labels, i.e., $f(\boldsymbol{u}) \neq f(\mathbf{v})$. Then we calculate the distance ratio D_r as

$$D_r = \frac{d(\mathbf{u}, \mathbf{z})}{d(\mathbf{v}, \mathbf{z})}, \tag{2.23}$$

where $d(.,.)$ represents the Euclidean distance between two-pixel vectors in the feature space. If D_r is less than or equal to the trainable threshold $T_r \in (0,1)$ then $\mathbf{z}$ is classified as the same label of $\mathbf{u}$. Otherwise, it is labeled as unknown or as UC. Mathematically, this can be expressed as

$$f_{DR}(\mathbf{z}) = \begin{cases} f(\boldsymbol{u}), & if\ D_r \leq T_r \\ 0, & if D_r > T_r \end{cases} \tag{2.24}$$

Where label 0 represents the label for unknown or UC samples, and f_{DR} represents the decision function for OSNN^{NNDR}.

2.3.4.3 Multi-class pairwise SVM with Platt probability estimation (P-SVM)

The P-SVM is a variant of MSVM which uses Platt probability estimation to produce probabilistic decision scores for classifying a test sample. The training phase of P-SVM follow the same procedure of MSVM, but it also includes Platt scaling to produce probability scores and then learning a thresholding probability score $T_p \in (0,1)$ for enabling the reject option. During the classification stage, for a given $\mathbf{z}$ the posterior probability score of i^{th} class (i.e., $P_s(y_i|\mathbf{z})$) using Platt scaling is given by,

$$P_s(y_i|\mathbf{z}) = \frac{1}{1+\exp\left(A_i f_{MSVM}(\mathbf{z}) + B_i\right)}, \forall i = 1, \dots, m \qquad (2.25)$$

where f_{MSVM} represents the prediction labels using MSVM, m represents the number of KCs and the two scalar parameters A_i and B_i are learned by the algorithm, i.e., model parameters. Then $\mathbf{z}$ is classified using the following decision rule

$$f_{PSVM}(\mathbf{z}) = \begin{cases} y_i, & if P_s(y_i|\mathbf{z}) \geq T_P \\ 0, & if P_s(y_i|\mathbf{z}) < T_P \end{cases} \qquad (2.26)$$

where y_i denotes the i^{th} class label that has the maximum probability score.

2.4 Classification Accuracy Assessment

The assessment of classification accuracy is an essential part of measuring the quality of information mining. Let us consider a confusion matrix $C^\dagger$ with elements $c_{ij}^\dagger$ where $i, j > 1$, and m represents the number of classes in the reference map. The following are the popular metrics, derived from the confusion matrix, used to quantify the accuracy of the output of any classification tasks.

- $Overall\ accuracy\ (OA) = \frac{\sum_i^m c_{ii}^\dagger}{\sum_{i,j}^m c_{ij}^\dagger} \times 100$ (2.27)
- $Producer's\ accuracy\ (PA_i) = \frac{c_{ii}^\dagger}{\sum_j^m c_{ij}^\dagger} \times 100$ (2.28)

- $Average\ accuracy\ (AA)\ = \frac{\sum_i^m c_{ii}^\dagger}{\sum_{i,j}^m PA_i} \times 100$ (2.29)

2.5 Real-Time Processing

A real-time processing system is required to process the inputs instantaneously (i.e., ideally) or near-instantaneously (i.e., practically) over time to say that the data processing is done in real-time or at runtime. If T_i is the time duration between two successive/consecutive inputs, then the real-time system has a time equal to or less than T_i to process and produce output. The restriction of the time specified for the computation of streaming inputs is known as the timing constraint. A processing system with T_{RT} denoting the time taken for processing inputs is said to be real-time processing if it satisfies the following timing constraint:

$$T_i \geq T_{RT} \tag{2.30}$$

An algorithm or a system is considered to be implementing real-time processing if the time taken by the system to process the inputs, say T_{RT}, is satisfying the timing constraint of the form shown in Equation (2.30). The value of time frame or time delay T_i is a vital attribute in real-time systems and is determined by the specific application or by imaging sensor scanning time.

2.6 FPGA Design Workflow

The current and emerging diverse use of image processing applications range from a wide spectrum of science and engineering streams to farming, industrial, planetary study, military, and so on. It is well known that most of these applications are associated with constraints like power usage, computation in real-time or near real-time, size, etc. Digital circuits like FPGAs, GPU, ASICs are popularly used to compensate for application-specific constraints than Traditional computing systems. Among the existing computing options, FPGAs have attracted a lot of attention due to their inherent on-the-fly reconfigurability, power usage, size, and cost-effectiveness. There are many methods

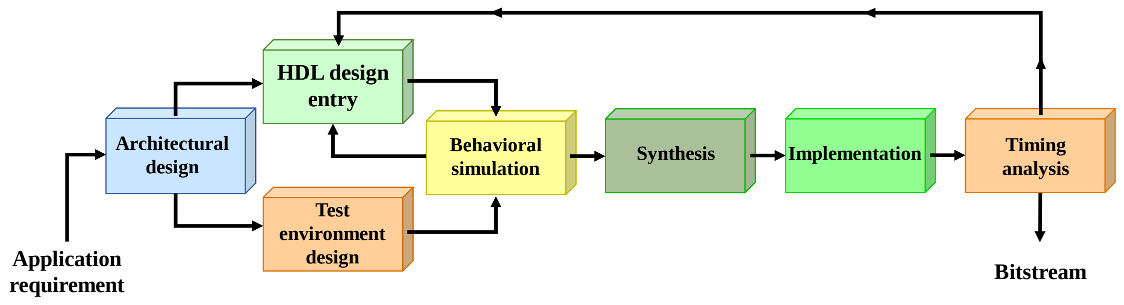


Figure 2.4: Illustration of a general workflow followed to design the FPGA architecture.

available to program an FPGA to the specification of an application. For example, the two most popular languages use hardware description language (HDL) style such as Verilog and very high-speed integrated circuit HDL(VHDL). However, the FPGA design flow typically comprises several steps, including design entry, synthesis, implementation, static timing analysis, and programing the FPGA with the generated bitstream file (see Figure 2.4). Based on the application specification, design entry is done using HDL or any tool that converts high-level script to HDL. A bitstream is generated after the design passes the behavioral simulation, synthesis, implementation, and timing analysis.

2.7 Literature Review

A systematic and comprehensive review of recent studies reported in the literature on real-time processing for image classification using HSIs/MSIs for real-time environments is discussed here from two different perspectives. One is from a computational perspective, and the other is from an algorithmic perspective. The computational perspective provides details about the relevant work reported on approaches focused on accelerating the MSI/HSI classification using FPGA as a hardware accelerator for achieving real-time processing. At the same time, the algorithmic perspective discusses the research studies reported on enhancing the robustness and classification performance of algorithms targeting the real-time environments with challenges like the presence of UCs, shadows, and varying illumination effects.

2.7.1 Computation perspective

In (Estlick *et al.*, 2001), a mapping of K-means unsupervised clustering-based image classification algorithm using both MSIs and HSIs to FPGA hardware is proposed. They have explored algorithmic level transforms and trade-offs such as distance measures and number representation formats for the K-means algorithm for designing reconfigurable FPGA hardware. The HSI dataset with 224 16-bit channels is from the AVIRIS sensor, and MSI with ten channels of 16-bit data is from a simulated multispectral thermal imager. The hardware implementation was on a peripheral component interconnect (PCI) board with three Xilinx Virtex1000 FPGAs written in VHDL. The experimental results indicate a two-fold speedup increase against the software implementation. In (Pingree, Scharenbroich and Werne, 2008), an on-board pixel-based image classification using FPGA co-processor design to implement a sea, water, ice, and land (SWIL) classifier is presented. They used Hyperion HSI with 11 and 30 arbitrarily selected bands out of 242 available bands in the image. The SWIL classifier uses a multiclass SVM method with linear and polynomial kernels. The FPGA co-design on the Xilinx Virtex-4FX60 FPGA on ML410 board is done using Impulse C tool which is a commercially available tool for converting C to HDL code. The obtained results indicate an overall hardware resource usage of less than 4% and a minimum accuracy difference between software and hardware implementations of the considered experiments.

In (Wang *et al.*, 2016), a scalable data flow engine (DFE) FPGA-based hardware accelerator for real-time on-board HSI classification is proposed. The accelerator is implemented on a Maxeler MAX4 DFE having Altera Stratix V FPGA and designed using java-based MAXJ language. The study used 6 out of 16 classes each from Indiana scene and Salinas valley scene HSIs with 9 out of 224 spectral bands that are recorded from AVIRIS sensor. The multiclass one-vs-one SVM algorithm with RBF kernel and Hamming distance as discriminant function is selected as the classification technique to implement offline training and on-board classification. The obtained accuracy results are satisfactory, FPGA fabric usage of less than 82%, and the significant speedup was achieved for FPGA over other accelerators indicating a plausible future for real-time on-board HSI classification. In (Basterretxea *et al.*, 2016), an extreme learning machine (ELM) based onboard learning and real-time classification using HSIs are presented. Two accelerators, namely, the Xilinx KC705 board with Kintex-7 FPGA and MicroZed board with Zynq-7020 programmable system on chip (PSoC), were considered in this study and are designed using custom made intellectual property (IP) cores. In this work, Kennedy space center HSI has 13 classes recorded from AVIRIS sensor with 176 bands in one case and reduced first 10 principal components in the other case were used for training and testing. The obtained results in terms of classification accuracy, latency, and throughput indicate feasibility of on-chip learning and classification tasks.

2.7.2 Algorithmic perspective

Some of the early and seminal works in remote sensing related to image classification even in the presence of uncertainties like UCs, shadows, and varying illumination effects are discussed here. In one of the early works, (Gorte and Gorte-Kroupnova, 1995) proposed a non-parametric image classification algorithm in case a UC is present in MSIs. Without knowing the prior probabilities of KCs and UCs, the described algorithm performs classification using the Bayesian decision rule and K-nearest neighbor (KNN) to estimate the maximum a posterior (MAP) probability of each class, including UC. The experiments were conducted on an RGB image and a three-band MSI. Similar work was conducted in (Mantero, Moser and Serpico, 2005), where a Bayesian algorithm with probability density function (PDF) estimation based on an SVM classifier with RBF kernel was used instead of k-NN. They have used two real MSIs and one synthetic MSI to perform image classification with different cases of UCs. The authors in (Jun and Ghosh, 2013) proposed a semi-supervised spatially adaptive mixture model (SESSAMM) to classify HSIs in the presence of UCs. Similar to previous works, SESSAMM uses a nonparametric Bayesian framework to handle mixture models with an unknown number of components. Two different HSIs were used to conduct the classification experiments with different UC in each trial. In (Mũnoz-Marí *et al.*, 2010), two semi-supervised OCSVM variants are proposed for OCC of different types of remote sensing data such as optical and radar. In (Condessa, Bioucas-Dias and Kovačević, 2016), a supervised HSI classification algorithm with rejection using contextual priors and MAP estimates was presented. The experiments were carried on two multi-sensor HSIs to demonstrate the classification with rejection for UCs.

The presence of shadows and varying illumination effects in images are two of the most challenging problems in real-world image-driven analysis. These problems are also severe with MSIs and HSIs because their presence will produce undesirable effects on image analysis algorithms. Over the years, many published scientific studies and articles in the literature indicate a growing research interest in shadow-related image analysis. In (Adler-Golden *et al.*, 2001), a physics-based shadow-insensitive detection or classification of materials using atmospherically corrected HSIs is presented. The atmospherically corrected HSI data is used to generate a simulated shadow spectrum signature of the chosen material and then used for shadow-invariant detection or classification problems. Later, they used spectral angle and distance for spectral-based matching between manually selected apparent endmember spectra and test examples. In (Dare, 2005), an unsupervised classification algorithm was used on a pan-sharpened four-band MSI and also on a single-band panchromatic image for shadow classification and found identical results for both the spaceborne images. And techniques like density slicing, thresholding, region encoding, and region filtering are used. In (Sanin, Sanderson and Lovell, 2012), a survey and comparative evaluation of recent techniques for shadow detection using traditional RGB images are presented. The taxonomy of recently published articles implies that using physics-based or texture-based features is the most used among other features. In (Qiao, Yuan and Li, 2017), a supervised shadow detection and classification algorithm using HSI of an urban setting is presented. The algorithm uses a sequence of steps like thresholding, spectral indices, feature extraction, spectral separability, SAM, maximum likelihood classifier, and SVM. In (Windrim *et al.*, 2018), a physics-based deep learning approach to shadow and illumination invariant representation of HSIs is presented. The method uses relit spectral angle stacked autoencoder-based unsupervised feature learning for mapping HSIs to shadow and illumination invariant images. Subsequently, a supervised classifier such as k-NN or SVM is used to label the invariant HSIs. In (Liu *et al.*, 2019), classification of HSIs by 2D-CNN based on shadow region enhancement through dynamic stochastic resonance is proposed.

2.8 Summary

This chapter presented a mathematical and theoretical background of data intake using MSIs/HSIs, processing, a brief description of various supervised classification methods, real-time processing, and FPGA design workflow. Further, a detailed survey of different studies related to MSI and HSI classification from both algorithmic and computation perspectives is briefly described.

CHAPTER 3

DESIGN AND DEVELOPMENT OF FPGA-BASED REAL-TIME IMAGE CLASSIFICATION

Prelude: This chapter presents and discusses a novel design and development of the hardware architecture for hyperspectral imagery classification using an FPGA as an onboard accelerator. A rapid FPGA prototyping scheme using a high-level architecture design tool is considered in designing the hardware architecture. The performance evaluation measures considered in our study are thematic classification accuracies derived from the error matrix and FPGA design metrics such as timing, speed, and logic capacity, and power usage.

3.1 Introduction

Over the past few decades, hyperspectral imaging techniques are increasingly used in various remote sensing application areas of science and engineering disciplines. Some application areas include monitoring and mapping applications of Earth system dynamics, medical, defense, and precision agriculture, etc (Clark *et al.*, 2003; Goetz, 2009; Madroñal *et al.*, 2017; Makki *et al.*, 2017). Hyperspectral imaging systems use specialized sensors to capture images having both spatial and spectral information of remote scenes for every pixel. They provide three-dimensional image data cube products, popularly known as HSIs, having two spatial dimensions and one spectral dimension (Eismann, 2012). The increasing interest in using HSIs as the primary choice of data in remote sensing analysis is because

This chapter is published as an article in *Geocarto International*, on 20th January 2020, with the title: "A real-time FPGA accelerated stream processing for hyperspectral image classification", and has a doi: 10.1080/10106049.2020.1713231. Authors: Dubacharla Gyaneshwar and Rama Rao Nidamanuri.

of their rich spectral content and spatial information of remotely sensed objects in a scene for each pixel (Manolakis, Marden and Shaw, 2003; Du and Nekovei, 2009). This rich spectral information enables accurate material discrimination with increased accuracy than MSIs. Recent technological advances in the field of remote sensing are the development of advanced imaging platforms and portable sensors.

In contrast to the spaceborne imaging platforms, airborne platforms such as aircraft and unmanned aerial vehicles (UAVs) have proven relatively effective in low operational costs. The state-of-the-art specialized imaging platforms and sensors can record high-spatial and spectral resolution imageries to cater to the requirements of a diverse range of remote sensing applications (Hagen and Kudenov, 2013; Yue *et al.*, 2017; Behmann *et al.*, 2018; Kemker, Salvaggio and Kanan, 2018). These advancements have fostered the way to broaden the current and future remote sensing application domains of HSI analysis (Khan *et al.*, 2018).

However, there are some practical challenges and issues involved in HSI analysis that are to be addressed to facilitate the application requirements of real-world tasks. One such challenging key aspect is that many current and future image analysis and image processing systems applications are time-critical because they usually operate in dynamic environments (Chang, 2016). Some examples of time-critical applications like disaster management, precision agriculture, health care, industrial inspection, surveillance, etc., require immediate assessment and response for (near) real-time decision making (Chang, Ren and Chiang, 2001; Salem and Kafatos, 2001; Gowen *et al.*, 2007; Sun, 2010). Unlike static or fixed environments, the information present in a testing scene in dynamic environments would change continuously, either periodically or aperiodically, over time. In some cases, testing environments can combine static or repeatable and dynamic or stream environments. So, it is imperative to have methodologies capable of performing the runtime or real-time data processing to mitigate the performance bottlenecks involved in analyzing large volumes of high-dimensional HSIs.

In general, real-time data processing deals with the arrival of a continuous stream of data inputs that needs to be analyzed in a short turnaround time to provide real or near-instantaneous outputs. The amount of time between two consecutive incoming data is the maximum available time to process the first input, produce output before the arrival of the second input, and so on. This runtime deadline of upper bound (i.e., maximum allowed) time to process inputs is known as timing constraint that is a vital attribute in the real-time processing systems (Chang, 2016). The timing constraints are defined based on the selected application, i.e., application-specific. Several research efforts have been directed towards HPC systems to address the computational requirements of various time-critical applications using HSIs, as reported in the literature. HPC systems such as FPGAs, GPUs, and cluster computing are popularly used in remote sensing missions (Plaza, 2008). Among the abovementioned HPC systems, FPGAs are advantageous in energy efficiency, reconfigurability, and portability (Fowers *et al.*, 2012; González *et al.*, 2013; Lopez *et al.*, 2013). Therefore, the use of FPGAs is more prevalent to achieve real-time processing for remote sensing applications, especially for HSIs analysis.

There are several HSI data analysis techniques such as classification, target detection, feature selection, and feature extraction. The objective of each one of the data analysis techniques is focused on providing the user-desired outputs using HSIs. However, classification is considered as one of the most prominent techniques of HSI analysis because it construes large quantities of information that has multiple usages, and it also includes subtle functionalities of detection, discrimination, identification, and quantification (Chang, Ren and Chiang, 2001; Chang, 2003; Ghamisi *et al.*, 2017). In recent years, there is a fair amount of research reported in the literature on HPC systems for remote sensing HSI analysis. Still, most of the studies are related to designing and implementing techniques such as unmixing, detection, and dimension reduction (Bioucas-Dias *et al.*, 2013). Yet, there is still a dearth of research on optimal design approaches of FPGA architectures and their implementations for HSI analysis techniques, more specifically for image classification.

Over the past few decades, there is an emerging trend in using machine learning techniques in remote sensing for image classification objectives. Among the existing machine learning-based image classification algorithms, SVM is one widely used supervised classification method. The wide usage of SVM is due to several benefits like their robustness to the noise, ability to provide models with sparse solutions for a reasonably limited amount of training data, and ease of usage in any given task (Cortes and Vapnik, 1995; Ghamisi *et al.*, 2017). The optimal sparse solution of the SVM, known as support vectors, is the use of a few training vectors to construct a hyperplane boundary (Camps-Valls *et al.*, 2004). Nevertheless, SVMs have computation complexity and linearly grow with the number of support vectors making it uncertain and complex to process HSIs using software implementation alone (Papadonikolakis and Bouganis, 2012). In such cases, high-performance reconfigurable computing systems like FPGAs are one of the potential solutions available in achieving the real-time processing of HSI classification.

However, there are several significant challenges present in designing FPGA-based hardware architecture for hyperspectral imagery classification. Some of the crucial challenges that are yet to be addressed in these areas are writing a comprehensive or full-scale hardware-specific code, debug, and design verification (Tessier, Pocek and DeHon, 2015). In particular, efficient strategies and frameworks with minimum usage complexity are needed to design FPGA architecture for HSI classification. In this regard, a high-level architecture design tool like XSG enables the automatic synthesis of HDL code (Feist, 2012; Xilinx User Guide, 2016). XSG tool operates at a high-level abstract model with libraries containing optimized predefined hardware-specific primitive digital signal processing (DSP) functions like addition, multiplication, accumulator, etc. Therefore, in this chapter, we employ XSG to design FPGA architecture that performs real-time hyperspectral image classification. The computation performance of the designed FPGA architecture is compared with the software-based implementation.

This chapter is organized into six sections. The following section presents the research problem statement, while section 3.3 briefly describes the datasets used in our analysis of this chapter. Section 3.4 presents the software and hardware implementation methodologies and frameworks used to investigate this chapter's goals. Section 3.5 describes and discusses the experimental results obtained. Finally, the last section presents the conclusions of this chapter.

3.2 Problem Statement

Consider an imaging spectroradiometer that takes T_{im} time to record spectral reflectance of scene surface to produce an HSI $\mathbf{X} \in \mathbb{R}^{m \times n \times d}$ with m rows, n columns, and d spectral channels. Then, let T_i be the time taken by the imaging sensor for recording a single-pixel vector, say i^{th} pixel vector $\mathbf{x}_i \in \mathbb{R}^d$ where $i = 1, \dots, N$ and $N = m \times n$. Therefore, T_i is the maximum time allowed to process a single pixel vector for achieving real-time or near real-time per-pixel based HSI classification. Mathematically, if T_{RT} is the time taken to process a single input test pixel vector and then to realize real-time processing of HSI classification, the timing constraint shown in Equation (2.30) should be satisfied.

This chapter presents a novel low-complexity strategy to design and develop FPGA architecture to realize onboard real-time HSI classification using supervised approaches. As mentioned before, a supervised learning approach has two phases, namely the training phase and the classification phase. Since the training phase has computational bottlenecks at run-time, it is better to perform offline training rather than online to avert the overheads of on-chip learning. After offline model training, the model parameters are used to design FPGA architectures to perform real-time pixel-based classification for a given HSI.

3.3 Datasets Used

We used four different sources of multi-platform and multi-sensor HSIs. One HSI each from AVIRIS-NG, reflective optics system imaging spectrometer (ROSIS), and two HSIs from Cubert UHD 185s sensor. The four outdoor images with different spatial and spectral resolutions are covering several land cover categories and sites. These HSI datasets were selected to form a complete and non-trivial experimental setup to assess the generalization of the methods. In order to measure the performance of algorithms quantitatively, we manually interpreted and collected the ground truth reference samples. True color composites and ground truth maps of the four images are shown in Figure 3.1.

Pavia University image: The airborne Pavia University HSI is provided by Prof. Paolo Gamba from the Telecommunications and Remote sensing laboratory (TRSL),

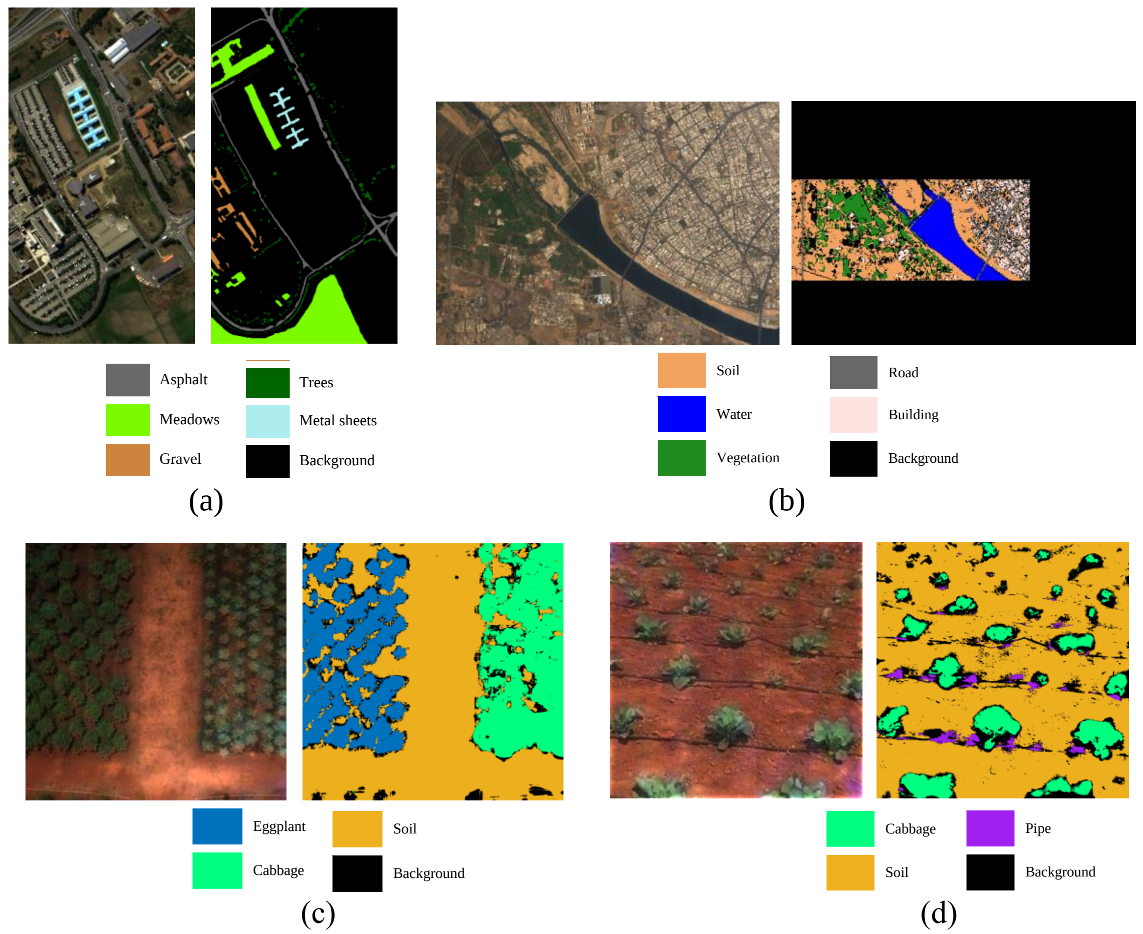


Figure 3.1: Three-band RGB display of hyperspectral data and their reference maps of: (a) Pavia University, (b) AVIRIS-NG, (c) Drone, and (d) Terrestrial. For each image, colored legend is also provided.

University of Pavia, Italy. The ROSIS dataset was acquired during a flight campaign by ROSIS-03 sensor, in July 2001, over the University of Pavia, Italy, under cloud-free sky conditions. This data comprising of 610×340 spatial pixels, each having a high-spatial resolution of 1.3 m and 103 spectral bands (after removing the water absorption and bad bands) were collected in the spectral range 0.43 to 0.86 μm with a nominal spectral resolution of 0.04 μm. We selected five heterogeneous urban classes out of nine information classes: asphalt, meadows, gravel, trees, and metal sheets. Figure 3.1(a) shows the true color composition and ground truth reference map for the Pavia University image.

AVIRIS-NG image: The airborne AVIRIS-NG HSI was acquired over the urban area of the Ahmedabad region, India, in January 2016. The subset image has a spatial dimension of 400×565 pixels with a fine spatial resolution of 4 m and has 351 spectral bands in the wavelength range 0.38 to 2.51 μm with a spectral resolution of 0.05 μm. The predominant land use categories in the study site are asphalt, building, soil, water, and vegetation. Figure 3.1(b) shows the true color composition image and ground truth reference map of the AVIRIS-NG image.

Drone image: The outdoor airborne-based platform Drone image was acquired by us on 5th April 2017. This image was recorded by the Cubert UHD 185s sensor mounted on Quadcopter-drone flying at an altitude of 20 m capturing the top view of agriculture crops grown in the fields of the University of Agricultural Sciences, Bengaluru, India. This image consists of 1000×1000 spatial pixels with a very high spatial resolution of 5 cm and has 139 spectral bands in the wavelength range 0.45 to 0.95 μm with a spectral resolution of 0.08 μm. Figure 3.1(c) displays the RGB image and ground truth reference map of the Drone image. Figure 3.1(c) shows that the drone image contains three dominant information classes: eggplant, cabbage, and soil.

Terrestrial image: The terrestrial HSI was acquired over the agriculture crops grown in the experimental fields of the University of Agricultural Sciences, Bengaluru, India, on 5th April 2017. This dataset was captured by the Cubert UHD 185s sensor mounted on the ground-based tripod capturing the top view of the cabbage crop. The Cu-T image has 1000×1000 spatial dimensions with an ultra-high spatial resolution of 2 mm, and a total of 139 spectral bands were collected in the spectral range 0.45 to 0.95 μm with a spectral resolution of 0.08 μm. The predominant LULC categories in the study site are cabbage crop, soil, and pipe. Figure 3.1(d) shows the RGB image and ground truth reference map of the Terrestrial image.

3.4 Methodology

This section briefly describes the FPGA-based classification framework in two steps: software implementation (offline mode for training) and hardware implementation (online mode for classification) (see Figure 3.2). A detailed discussion of each step is as follows.

3.4.1 Software implementation of MSVM training

As mentioned before, a MATLAB version of the training stage based on the LIBSVM library is realized to formulate a classification model before online classification (see Figure 3.2). To ensure the stability and performance of the classification model on out-of-sample data, the cost parameter of the MSVM classifier C is tuned by grid search technique using 10-fold cross-validation (CV). The search is conducted across the predefined parameter space $C = \{2, 10^0, 10^1, 10^3, 10^5, 10^7\}$. Each C value with 10-fold CV results in cross-validation accuracy (CVA) and its expression is given by CVA=$mean\{[OA_{\text{cv}}]_{\text{cv}=1}^{10}\}$. As shown in Figure 2.2, a final model is formulated from a set of candidate models by selecting an optimal hyperparameter (C_{BEST}) which gives the highest CVA (i.e., best CVA) among the group of candidate models. The considered CV scheme prevents the final model from overfitting the training data and generates an effective classification model (Chang and Lin, 2011). The trained classification model so obtained has P binary hyperplanes, each defined by the parameter set denoted $\theta_{dual} = \{\mathbf{x}_i, y, \alpha, b\}$, thereby introducing considerable memory requirement. This problem is addressed by transforming Equation (3.1) back to the primal version (Pingree, Scharenbroich and Werne, 2008; Madroñal *et al.*, 2017). The dual-form of a linear kernel classification model $f(\mathbf{z}; \theta_{dual})$ can be optimized in terms of input arguments by rewriting it in the primal form $f(\mathbf{z}; \theta_{primal})$ ($\theta_{primal} = \{\mathbf{w}, b\}$) using Equation (3.2) (see Equations (3.3) and (3.4)). As a result, the MSVM model performance is maximized by reducing the considerable amount of memory and computational resources required for a pixel vector-level classification.

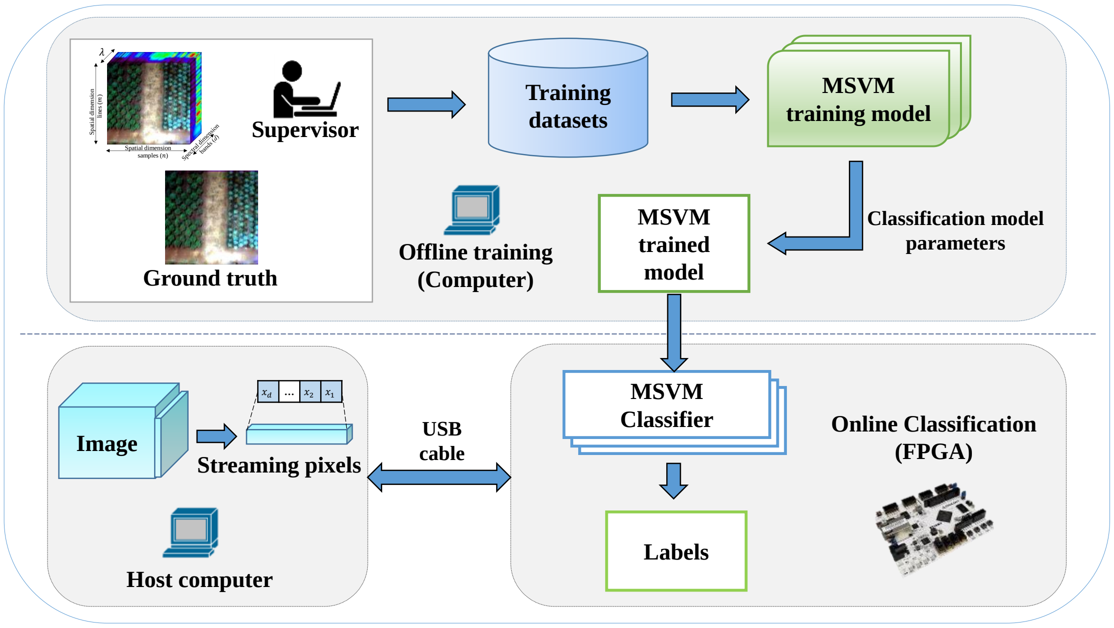


Figure 3.2: A schematic block diagram of the proposed classification framework. Note that offline training in the host computer and online classification using FPGA are shown above and below of horizontal dashed line, respectively.

$$f(\mathbf{z};\theta_{dual}) = sign\left(\left(\sum_{i=1}^{K} y_i \cdot \alpha_i \cdot \left(\mathbf{x}_i^{\mathrm{T}} \cdot \mathbf{z}\right)\right) + b\right) \tag{3.1}$$

$$\mathbf{w} = \sum_{i=1}^{K} \alpha_i \cdot y_i \cdot \mathbf{x}_i. \tag{3.2}$$

$$f(\mathbf{z}) = \left(\sum_{i=1}^{K} \alpha_i \cdot y_i \cdot \mathbf{x}_i\right)^{\mathrm{T}} \mathbf{z} + b \tag{3.3}$$

$$f(\mathbf{z};\mathbf{w},b) = \mathbf{w}^{\mathrm{T}}\mathbf{z} + b \tag{3.4}$$

3.4.2 Hardware implementation of MSVM classification

This subsection describes an FPGA-based hardware design for the MSVM algorithm using a system-level modeling approach for rapid prototyping. An abstract representation of a communication process between systems is shown in Figure 3.2. The top-level architecture

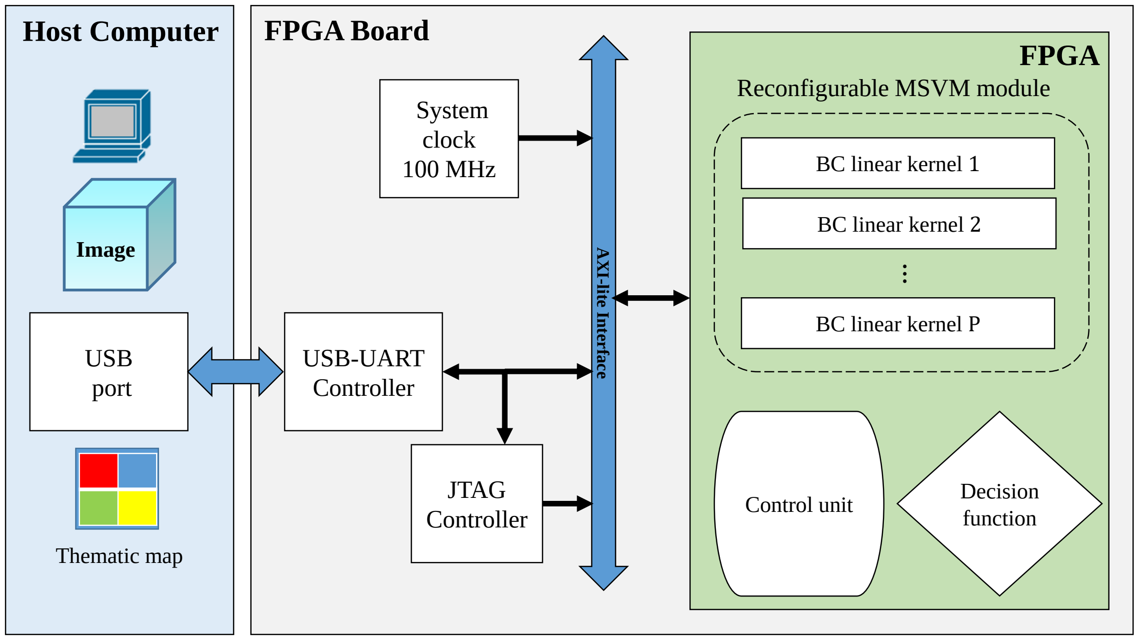


Figure 3.3: Top-level architecture of online-classification framework. The blue filled box represents host computer for streaming HSIs and receiving labels, the FPGA board consists of modules and the green box represents the reconfigurable hardware modules for implementing the MSVM algorithm.

of the proposed scheme consists mainly of two systems: a host computer and an FPGA board (see Figure 3.3). The host computer is responsible for HSI data streaming, receiving labels, and input/output (I/O) communications with FPGA. A typical FPGA board has two modules: a fixed functionality module and an FPGA (XC7A35T) chip. The fixed modules on the FPGA board used in the current study include a 100 MHz system clock, a universal serial bus (USB) - universal asynchronous receiver-transmitter (UART), and a joint test action group (JTAG) interface (that provides USB connectivity to FPGA board with a UART interface). The XC7A35T FPGA chip is a custom circuitry module that is used to implement the MSVM algorithm. The data communication between the host computer and the FPGA board is realized using the USB-UART serial interface. Unlike conventional I/O interfaces, which use standard interfaces like RS-232, PCI express for data streaming, we use a relatively flexible USB-UART bridge interface between host and device. The USB-UART interface offers a plug-and-play device connectivity solution (i.e., connection

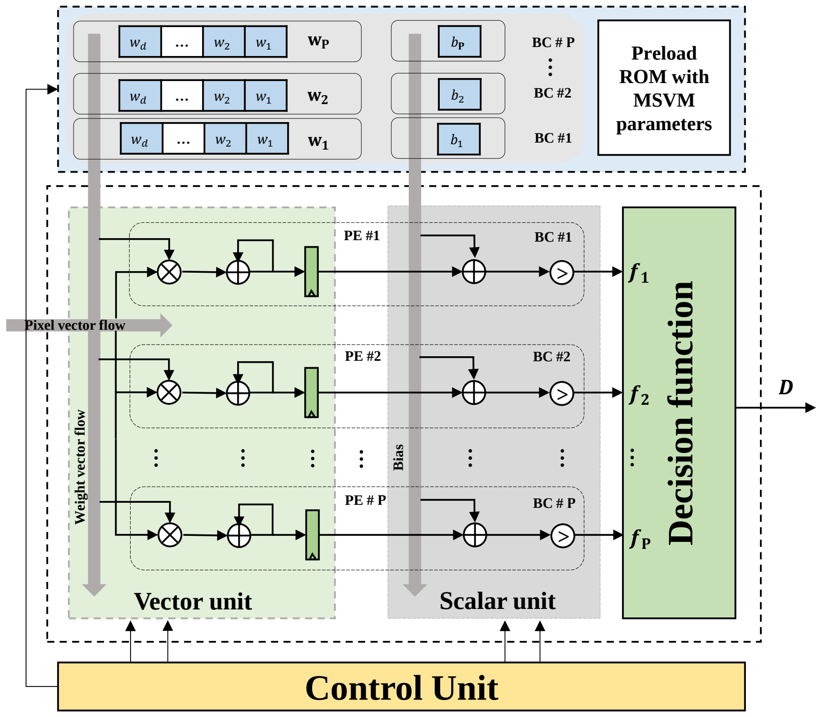


Figure 3.4: Streaming pipelining architecture of the MSVM algorithm, where P represents the number of BCs. Note that pixel vectors and parameters are streamed synchronously from left to right and top to bottom, respectively.

through USB port) for UART devices. The advanced extensible interface (AXI-lite) is used to link the modules on the FPGA board (see Figure 3.3).

The advantage of an offline training step before hardware architecture design is to have a basis for making a projection of memory usage and independent operations required for the online classification task. Under these considerations, we have exploited the inherent parallelism of MSVM for constructing an efficient and computationally coherent streaming architecture. The stream processing architecture performs operations synchronously without explicitly managing the memory allocations and yields high throughput. As mentioned earlier, a real-time processing approach that implements pixel vector-level streaming classification has been adopted to enhance our architecture's computing performance and make it compatible with the systems using BIP data format for image

Table 3.1: Entries for decision function D and their class labels for a 5-class classification problem

Class \ BC	1v2	1v3	1v4	1v5	2v3	2v4	2v5	3v4	3v5	4v5
Class-1	1	1	1	1	X	X	X	X	X	X
Class-2	-1	X	X	X	1	1	1	X	X	X
Class-3	X	-1	X	X	-1	X	X	1	1	X
Class-4	X	X	-1	X	X	-1	X	-1	X	1
Class-5	X	X	X	-1	X	X	-1	X	-1	-1

Note: X-represents either label '1' or '-1'

acquisition (Du and Nekovei, 2009). In the proposed technique, the pixel vector (**z**) is streamed simultaneously to all the BCs in series fashion i.e., pixel-by-pixel.

As shown in Figure 3.4, the proposed architecture consists of four main modules: the memory, the processing elements (PEs), the control unit, and the decision function module. First, the memory module comprises a group of read-only memories (ROMs) to store the MSVM model parameters obtained from offline training. Second, a homogenous network of PEs consisting of a vector unit (VU) and a scalar unit (SU) are responsible for computing vector and scalar operations in each BCs. Each PE has its data and instruction memory to store partial results, thereby facilitating stream processing and pipeline parallelism (see Figure 3.4). Third, a control unit constitutes the deterministic finite state machine (FSM) to transmit essential control signals like address lines, reset (RST) and enable (EN) to synchronize the internal processing flow. Finally, a decision function D is implemented using the conditional constructs designed in the M-Code block from the XSG library (Feist, 2012; Xilinx User Guide, 2016). The possible outcomes of each BC in a 5-class MSVM classifier and their corresponding class labels (C1 to C5) are shown in Table 3.1. Similarly, the BC outcomes given in Table 3.1 are also valid for k<5 MSVM classifiers associated with class labels (C1 to Ck). The reconfigurable module implementing a BC kernel consists of five modules: the first module is a vector multiplication; the second is an adder; the third and fourth modules are storage components (such as ROMs) of weight vector and bias, and the last module is a comparator which is responsible for binary decision f.

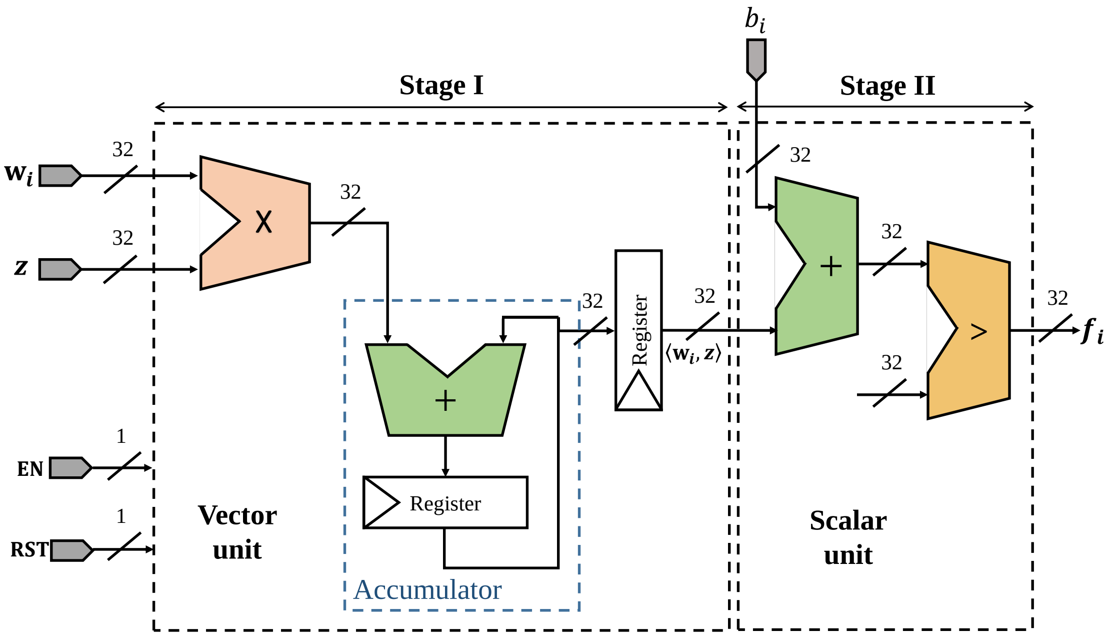


Figure 3.5: Processing element (PE) structure for i^{th} BC to compute a decision function f.

As shown in Figure 3.5, the BC linear kernel or PE structure comprises two processing stages in classifying a pixel vector to a binary label. The first stage of the BC linear kernel consists of VU, which performs dot products of weight vector **w** and test pixel vector **z**. A second stage includes a SU, which is responsible for the addition of bias and scalar value generated from a VU, and finally, the value is pipelined to the comparator. Mathematically, the dot product or inner product can be divided into a series of operations to exploit the hardware parallelism in streaming architecture. The first operation is based on Hadamard product ($\mathbf{w} \circ \mathbf{z}$) which is an element-wise multiplication of two vectors given by

$$\begin{aligned} \langle \mathbf{w}, \mathbf{z} \rangle &= \mathbf{w}^{\mathrm{T}}\mathbf{z} = \mathrm{w}_1\,\mathrm{z}_1 + \cdots + \mathrm{w}_d\,\mathrm{z}_d \qquad (3.5) \\ &= \sum_{i=1}^{d} \mathrm{w}_i\,\mathrm{z}_i = \sum_{i=1}^{d} [\mathrm{w} \circ \mathrm{z}]_i \\ [\mathrm{w} \circ \mathrm{z}]_i &= [\mathrm{w}]_i[\mathrm{z}]_i \;\forall\; 1 \le i \le d. \end{aligned}$$

The next operation is adding the Hadamard products using the accumulator block from XSG, as shown in Figure 3.5. Another important aspect of our hardware architecture is the use of pipelined arithmetic modules from the XSG library by adding pipeline registers to

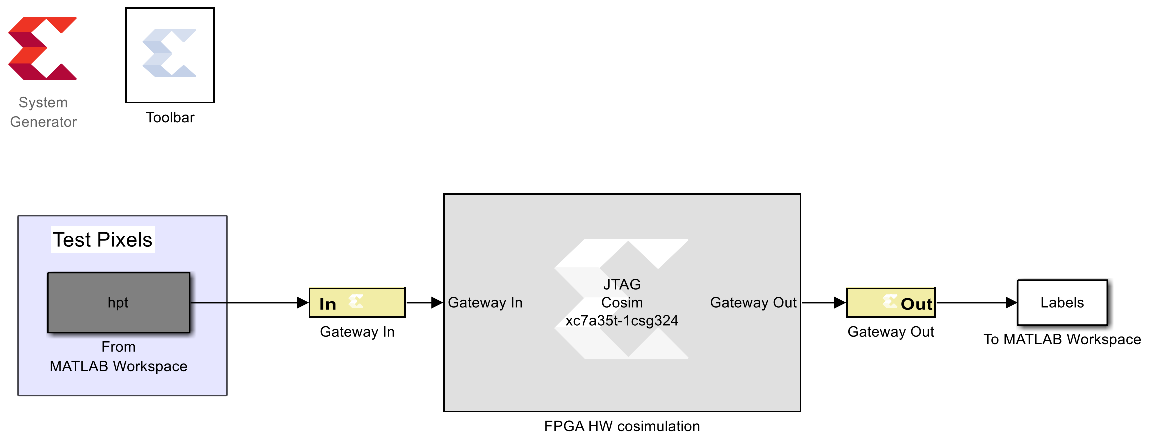


Figure 3.6: JTAG based HW/SW co-simulation for verifying FPGA-based hardware logic

the outputs in each PEs. These overhead registers are added to shorten the critical paths in the designed circuitry based on timing closure. Even though these registers initially increase the latency, they help process the data in the pipelined parallelism fashion for a given stream, which increases the throughput of the design. In the second stage of PE, SU is used to add the scalar values produced from the VU and bias.

For any given input $\mathbf{z}$, the streaming sequence is $\{\mathrm{z}_1, \ldots, \mathrm{z}_d\}$ and the corresponding sequence of weights (as a stream of weights) are $\{\mathrm{w}_1, \ldots, \mathrm{w}_d\}$ in respective BCs. A VU computes the inner product as a sequence defined by $\sum_{i=1}^{d} \mathrm{w}_i z_i$. The SU performs addition of the output produced from a VU and bias of respective BC as $\left(\sum_{i=1}^{d} \mathrm{w}_i \mathrm{z}_i\right) + b$. Then the overall result is converted by the comparator to a binary output $f_{\mathrm{P}} \in \{-1, +1\}$. Each BC provides a binary value to decision function D and which further maps to set y. Several offline MATLAB-Simulink simulations were conducted to deduce the optimum number of bit representations required for our FPGA-based MSVM implementations. Accordingly, the MSVM architecture is configured with a 32-bit signed fixed-point data format with one sign bit, 24 fractional bits, and seven integer bits to minimize the overflow generated in calculating the decision function (see Figure 3.5). With these considerations, we have performed timing closure in XSG to meet the timing requirements. After successfully satisfying all timing constraints, the maximum operating frequency of the design F_{max} is calculated (Zhang, 2017).

The testing of the proposed FPGA design is achieved by using the hardware-in-loop (HIL) approach in MATLAB/Simulink via XSG, which is used for I/O data communication with FPGA using JTAG based co-simulation interface, as shown in Figure 3.6. The XSG provides a HIL simulation that enables rapid prototyping by evaluating the proposed design in a real-time environment. Finally, we have verified the hardware and software functionality of the design using JTAG based HW/SW co-simulation (Feist, 2012; Xilinx User Guide, 2016).

3.5 Experimental Results and Discussion

The hardware architecture described in Section 3.3 has been implemented on an Artix-7 35T FPGA Arty Evaluation kit using XSG to the specifications of the MSVM algorithm. The 35T board provides features common to many embedded processing systems, including 256 MB DDR3L SDRAM memory, 210 I/O ports, other expansion interfaces. The XC7A35T FPGA has a total of 5200 logic slices, 20800 look-up tables (LUTs), and 41600 slice flip-flops. In addition, the FPGA also includes some heterogeneous resources such as 90 DSP slices and 1800 Kbits block RAMs (BRAMs).

3.5.1 MSVM classification accuracy assessment

In order to assess the classification performance of the MSVM algorithm mapped on FPGA, all four hyperspectral imageries were classified using an FPGA board, as shown in Figure 3.7. Note that the obtained classified images [see Figure 3.7(a), 3.7(b), 3.7(c), 3.7(d)] illustrate that the MSVM classifier has also mapped the untrained classes to one of the trained classes which are closer to the labels from y. The thematic accuracy of the implemented hardware-based MSVM classifier is validated using the PA of each class ($\mathrm{C}i$, $i = 1, \dots, k$) (see Equation (3.6)) and OA (see Equation (3.7)) derived from the error matrix or confusion matrix. Overall results in Table 3.2 indicate a successful estimation of class

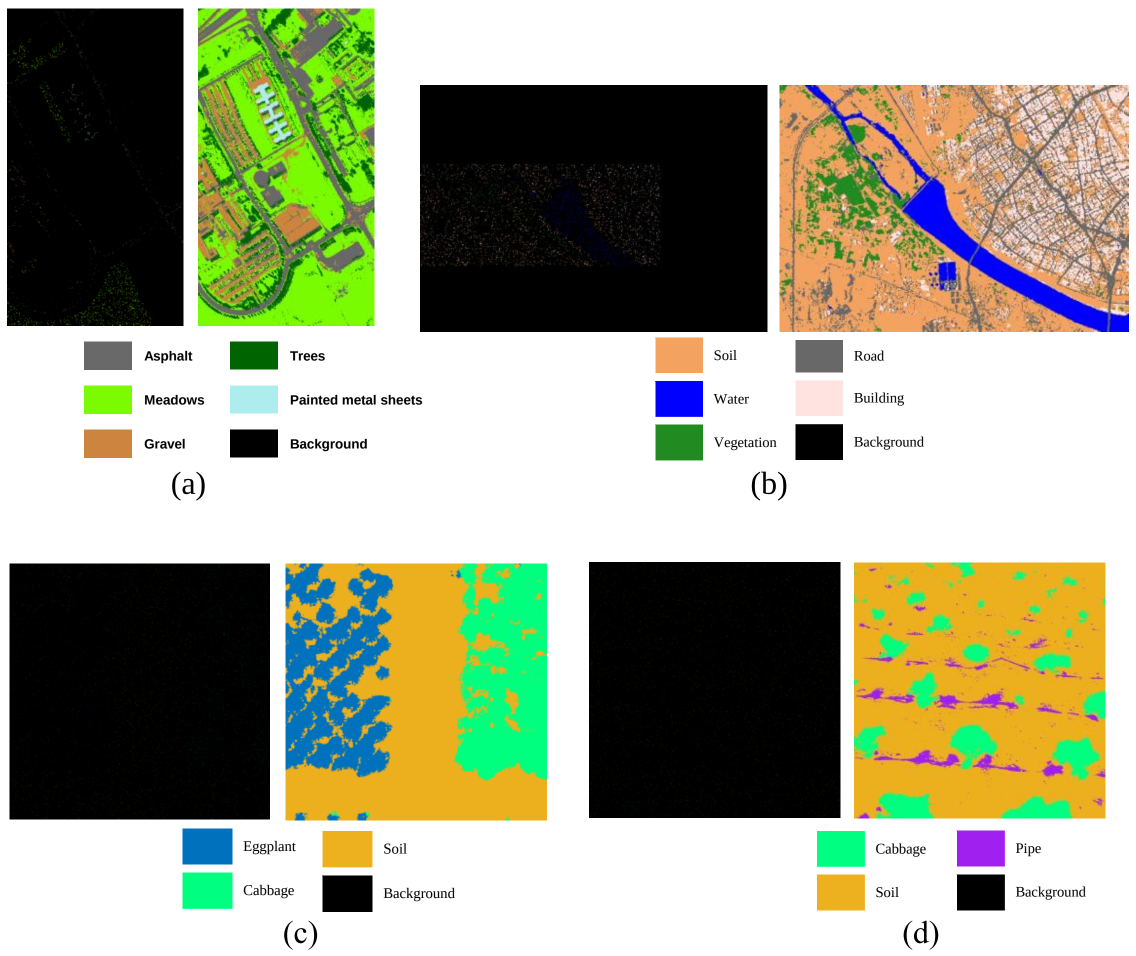


Figure 3.7: Training set reference maps for learning and obtained classified images of (a) Pavia University, (b) AVIRIS-NG, (c) Drone, and (d) Terrestrial.

labels using the proposed implementation with a minimum statistical confidence interval at speed commensurate with image acquisition rates.

$$PA = \frac{Number\ of\ correctly\ classified\ pixels\ of\ class\ \mathrm{Ci}}{Number\ of\ reference\ pixels\ of\ class\ \mathrm{Ci}} \tag{3.6}$$

$$OA = \frac{Total\ number\ of\ correctly\ classified\ pixels\ of\ all\ classes}{Total\ number\ of\ reference\ pixels\ of\ all\ classes} \tag{3.7}$$

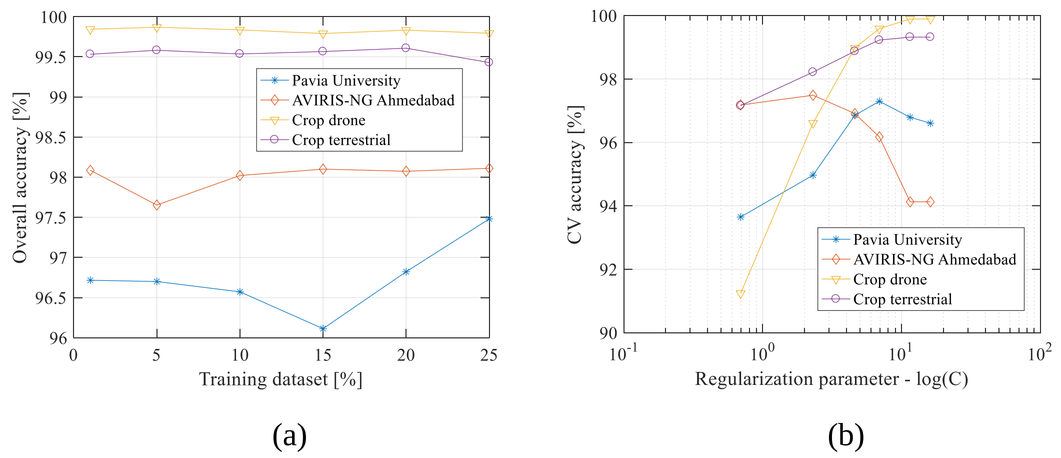


Figure 3.8: (a) OAs of MSVM over different percentages of training dataset size for the datasets considered; (b) CVAs obtained in relation to six different values of C parameters for four different datasets.

Table 3.2: Class-wise classification accuracy obtained using proposed FPGA-based MSVM implementation

	Pavia University		Ahmedabad		Drone		Terrestrial	
	Classes	**PA (%)**	**Classes**	**PA (%)**	**Classes**	**PA (%)**	**Classes**	**PA (%)**
Class-wise accuracy	Asphalt	98.85	Building	93.05	Eggplant	99.84	Cabbage	98.64
	Meadows	98.03	Road	93.05	Cabbage	99.76	Soil	99.61
	Gravel	91.67	Soil	99.25	Soil	99.91	Pipe	91.76
	Trees	91.17	Vegetation	99.43	C4	-	C4	-
	Metal sheets	99.68	Water	99.92	C5	-	C5	-
	OA (%)	**97.12**	**OA (%)**	**98.09**	**OA (%)**	**99.86**	**OA (%)**	**99.33**

Note: PA-Producer's accuracy; OA-Overall accuracy

3.5.2 Sensitivity analysis of OA

The OAs of the classified HSIs listed in Table 3.2 are similar to the MATLAB-based software implementation and are acceptable for many classification applications. However, the accuracy performance may vary due to various factors, as listed below.

3.5.2.1 Training set size

Figure 3.8(a) shows that the OAs of AVIRIS-NG and Cubert imageries are marginally stable after 10% of training data size, except for Pavia University. This trend is due to the consideration of large ground truth reference samples of AVIRIS-NG and Cubert imageries [see Figures 3.1(b), 3.7(b), 3.1(c), and 3.7(c)] when compared to Pavia University HSI.

3.5.2.2 C-parameter tuning

In Figure 3.8(b), the CVA increased significantly due to a smaller-margin hyperplane formulated for larger values of $C = \{2, 10^0, 10^1, 10^3, 10^5, 10^7\}$ and then it became insensitive for further increase in C (see Sections 2.2.1 and 3.4.1). The optimal C controls the tradeoff between large margin and small training error. Therefore, the grid searching mechanism is beneficial in selecting the optimal C for generalizing a classifier to out-of-sample data compared to the single choice of C parameter, as the outliers in the data may vary.

3.5.2.3 Normalization

A pre-processing step such as normalization is used to enhance and generalize the performance of a classifier in terms of classification accuracy. In the current methodology, we have considered the min-max normalization technique to scale the HSI data between [0,1], indicating spectral reflectance values ranging from 0 (minimum) to 1(maximum). The data normalization is performed offline in MATLAB on both training and testing datasets of HSIs. The normalized HSI data is used for offline training to obtain a model and then use the trained model to predict the testing dataset online (e.g., FPGA). The min-max normalization of data helps in controlling the overflow errors in arithmetic operations like multiplication and accumulation.

3.5.2.4. Number representation

In this paper, the hardware implementations are designed for fixed-point operations in which the number representation is crucial for determining the results (see Section 3.4.2). In many cases, the selection of number representation plays a significant role and may even result in errors because of the results quantization effects and overflow truncations.

Table 3.3: Summary of resource utilization for the proposed FPGA-based implementation of the MSVM algorithm

Components	**Available**	**Pavia University**	**Ahmedabad**	**Drone**	**Terrestrial**
		Utilization	**Utilization**	**Utilization**	**Utilization**
BRAMs	50	7 (14%)	7 (14%)	**3.5 (7%)**	**3.5 (7%)**
DSPs	90	40 (44.44%)	40 (44.44%)	**12 (13.33%)**	**12 (13.33%)**
LUTs	20800	1874 (9%)	1875 (9.01%)	**1185 (5.69%)**	**1185 (5.69%)**
Registers	41600	2056 (4.94%)	2058 (4.95%)	**1370 (3.29%)**	**1370 (3.29%)**
Maximum operating frequency (MHz)		104.61	103.41	**194.43**	**194.43**

3.5.3 Real time performance evaluation of FPGA based MSVM implementation

The performance evaluation of a real-time computing platform typically includes an estimation of its resource usage, throughput, latency, and processing time (Basterretxea *et al.*, 2016). As mentioned before, the FPGA design was implemented on the Artix-7 FPGA 35T Evaluation Kit. The XC7A35T-1CSG324 programmable fabric is used to design MSVM under several constraints like achieving expected operating frequency with optimum utilization of the hardware resources. As shown in Table 3.3, four different implementations are compared based on their hardware fabric usage in our experiment. For the ROSIS and AVIRIS-NG imagery, we constructed 10 BCs in parallel to form the 5-class MSVM. For the drone-mounted and ground-based Cubert implementations, the number is 3 to create a 3-class classifier. Moreover, according to the number of spectral bands and the classes trained, different resources are used by the four implementations. As a result, the percentage of hardware utilization increases as spectral bands and classes increase. As shown in Table 3.3, images acquired using the Cubert sensor consumed similar resource units. The maximum operating frequency of each implementation for the considered imagery is also given in Table 3.3. The results demonstrate that all four implementations can operate with a maximum frequency (F_{max}) of more than 100 MHz. It is important to emphasize that the performance of a computing system is directly related to the operating

Table 3.4: Real-time performance of the proposed MSVM design for the considered hyperspectral imageries

Hyperspectral images	**Latency**	**Throughput**		**Time**			**On-chip power (Watts)**
	Time (μs/pixel vectors)	**Pixel vectors/s**	**Data rate (MB/s)**	**MATLAB (s)**	**FPGA (s)**	**Speedup**	
Pavia University	1.09	909,652	140	7.74	0.20	38.7	0.156
Ahmedabad	3.51	284,876	174	10.95	0.77	14.2	0.156
Drone	**0.77**	**1,287,615**	**268**	**20.29**	**0.71**	**28.6**	**0.096**
Terrestrial	**0.77**	**1,287,615**	**268**	**20.29**	**0.71**	**28.6**	**0.096**

clock frequency, more operations per second are computed for a higher clock frequency which results in higher performance.

We designed a pipeline-based architectural framework to achieve low-latency and high-throughput rates, which are essential parameters for real-time system realization. The results are shown in Table 3.4. The proposed design leads to a latency of about $(d + 12)$ clock cycles for a d-dimensional pixel vector. It achieves a per-pixel vector classification or per-pixel vector processing time of 1.09 μs and 3.51 μs (shown in Table 3.4), against the per-pixel-vector scanning time of 31.5 μs, 15.6 μs for ROSIS and AVIRIS-NG instruments to achieve real-time performance, respectively (Zebin Wu *et al.*, 2015; Thorpe *et al.*, 2016). As a result, the proposed FPGA-based streaming pipeline architecture of MSVM performs classification strictly in real-time. Since the spectral images acquired using the Cubert sensor have a post-processing stage such as image fusion (Yue *et al.*, 2017), the obtained results in Table 3.4 offer a compelling computing performance for drone and terrestrial-based imagery. As shown in Table 3.4, the metric for throughput is expressed as the pixel vector per second and data rate. The results shown in Table 3.4 indicate that the throughput and latency performance primarily depend on the number of spectral channels and the BCs.

In real-time classification based applications, short predicting time is essential (Chang, Ren and Chiang, 2001; Du and Nekovei, 2009). When a k-class classification problem has a large k, the one-against-one method has to calculate P decision function f values for each datum or pixel vector, rendering sizeable computational complexity, which leads to considerable predicting time. Considering this, we have shortened the predicting time by P folds by implementing all BCs in parallel. An interesting feature of the proposed design is that it can be scaled without significantly increasing the delay. Although the current Arty 35T FPGA platform is operating at 100 MHz clock, which is approximately 24 times lesser than the operating frequency of the computer, it still presents a higher performance (at least ten times more) than the same block implemented in software as shown in Table 3.4.

Real-time processing requires computation time equal to or less than the scan time of the sensor or the time at which the data arrives at the input of the processing system. For example, the time required to acquire the considered scene for the ROSIS sensor (i.e., 610×340 pixels collected over the Pavia University area) is about 6.5s, and the AVIRIS-NG sensor is about 3.5s (for 400×565 samples acquired over Ahmedabad city). These sensor-specific scanning times introduce a computing time required for the ROSIS sensor scene in less than 6.5s and the AVIRIS-NG sensor scene in less than 3.5s to achieve real-time performance. Thus, our design has classified approximately in 0.20s and 0.77s for ROSIS and AVIRIS-NG, respectively. As shown in Table 3.4, for all the considered hyperspectral imagery, the FPGA implementation can process and classify in real-time specifications and offers a significant speedup compared to the MATLAB-based LIBSVM software version. The processing time depends on the ratio of clock cycles and F_{max} which are achieved for the synthesized design in the XSG tool. The power consumed by the on-chip FPGA device (XC7A35T) is estimated using the Xilinx Vivado tool for each imagery (see Table 3.4). Results indicate that power usage mainly depends on the number of BCs in the MSVM.

3.6 Chapter Conclusions

This chapter introduced a low-complexity real-time FPGA design and implementation of the MSVM algorithm using a model-based rapid prototyping design approach. In particular, we have presented a comprehensive analysis of both the learning and classification stages, and experiments were performed on four different hyperspectral datasets acquired from diverse platforms. The experimental results, conducted on a newer generation of low-power Artix-7 Arty 35T board, report a maximum operating frequency of more than 100 MHz for our proposed design. Finally, the reported results indicate an apparent increase in speedup over the software version and satisfy strict real-time constraints for the imaging sensors considered in this experiment. As future work, we will develop and integrate the pre-processing stage and evaluate the processing times. The present study provided the scope and capabilities of FPGAs in providing real-time solutions for airborne and ground-based ultra-high resolution hyperspectral imagery at the native spectral resolution.

CHAPTER 4

SUPERVISED CASCADED CLASSIFIER SYSTEM (SC^2S) FOR OPEN-SET IMAGE CLASSIFICATION IN REAL-WORLD ENVIRONMENTS

Prelude: This chapter proposes a new supervised classification algorithm specifically designed to provide robust and stable accuracy performance in both static and dynamic environments in real-world tasks. The performance of the proposed algorithm is compared with other widely used existing state-of-the-art algorithms with and without reject-option. This chapter also presents an efficient strategy to design an experimental setup consisting of different real-world classification scenarios to evaluate multispectral and hyperspectral imagery classification performance. Experiments carried out on four multi-sensor, and multi-platform images show that the proposed technique successfully offers superior performance even in cases of a large number of UCs.

4.1 Introduction

Classification is one of the standard techniques of analysis to mine the information present in remotely sensed optical images such as color images, MSIs and HSIs. It is a widely researched and used technique because of its potential to provide thematic information to a broad spectrum of practical applications like environmental study (Liu and Han, 2017), weather analysis (Piñeros, Ritchie and Tyo, 2011), resource exploration (Adep, shetty and

---

This chapter is published as an article in *IEEE Geoscience and Remote Sensing Letters*, vol. 18, no. 3, pp. 421-425, March 2021, doi: 10.1109/LGRS.2020.2980186, with the title: "A Novel Supervised Cascaded Classifier System (SC^2S) for Robust Remote Sensing Image Classification". Authors: Dubacharla Gyaneshwar and Rama Rao Nidamanuri.

Ramesh, 2017), LULC change investigation, machine vision, etc (Khan *et al.*, 2018). Several different classification techniques are available in the literature that has proven successful in diverse fields, particularly in remote sensing (Mather and Tso, 2016; Ghamisi *et al.*, 2017). Some of the notable classification techniques such as SVM, convolution neural network (CNN), and more are adopted from prominent paradigms like machine learning, deep learning, signal, and information theory. Yet, there is still a more significant requirement for consistent and trustworthy classification performance in real-world environments, especially for the HSIs. The inconsistent and sometimes unreliable performance of classification is due to the presence of various uncertainties in a scene of studies like incomplete a priori knowledge about the test site, inevitable presence of spectral classes, noise, etc. Because of these uncertainties, most of the existing techniques are known to produce substantial omission or commission errors in the classification (Muzzolini, Yang and Pierson, 1998; Mantero, Moser and Serpico, 2005; Jiang *et al.*, 2018). The solution to this ubiquitous problem continues to be a challenge that needs to be addressed to deploy the classification methods in real-world application tasks.

The omission or commission errors produced by the classification techniques are mainly due to uncertainties like OoD or UC samples in the scene being observed. Examples of UCs include untrained or unseen samples during training, undesired spectral classes, and within-class samples with substantial variation in surface reflectance due to varying illumination, shadow, etc (Foody and Atkinson, 2002; De Rosa, Mensink and Caputo, 2016; Júnior *et al.*, 2017). The presence of UCs in images causes the classifier to produce undetected false-known or false-positive errors during classification and often yields high-confidence class prediction results even for random noise. For example, we train a classifier with an urban feature training set and anticipate future test instances during classification only from urban built-up settings. But in the real world, this is not the scenario; there also exists a scenario where many UCs are present in the test data. In such cases, the trained classification models are susceptible to produce errors for UCs by labeling them as one of the ID pre-trained information class labels, i.e., KCs. This case is due to the forced assignment problem of classifiers that arises from the assumption made during the training stage, i.e., the training and testing instances will be from the same ID. And this scenario of a classification problem where the training and test instance is from the same ID is a closed-set classification problem.

On the contrary, an open-set classification problem has different training and test instances. In fact, a practical, real-world classification environment is often an open-set setting. Therefore, researchers have a great desire to develop classification algorithms capable of performing stable in any given application environment.

To deal with the problem of UCs, we propose a novel classification algorithm named supervised cascaded classifier system (SC^{2}S) that is developed by joining two supervised learning approaches in a cascade fashion. The two interconnected stages in SC^{2}S are the supervised novelty classifier (SNC) and the rule-based supervised classifier (RSC). The SNC is the first stage in SC^{2}S responsible for grouping the incoming test instances as UC if there are no reliable training samples in the training stage. Otherwise, the test instance is given a KC label. In the second stage of SC^{2}S, RSC uses a decision rule-set to label the test instance as UC or as one of the KC labels. Stage-II (RSC) rule-set is a function of the predicted label from Stage-I (SNC). In such a manner, the proposed cascading arrangement of diverse supervised classification models in the SC^{2}S algorithm unifies novelty detection and classification objectives. Thus, SC^{2}S can discriminate the KCs and UCs with increased accuracy and minimize the false-positive errors of overconfident predictions by alleviating the forced assignment problem.

A unique class label (i.e., referred to as class-0) is introduced in SC^{2}S to group all the UC test instances during classification. The class-0 label represents the availability of reject-option for UCs in SC^{2}S for not allowing the UC instances for supervised classification in RSC. So, this reject-option for UC instances in classification is particularly advantageous in open-set classification tasks to ensure reliability, safety, and security rather than producing inaccurate class predictions (Júnior *et al.*, 2017). There are several research attempts made in the past to tackle the UC problems for classification using MSIs/HSIs, but the reported methods are complex and may produce suboptimal results if the settings are not correctly configured (Mantero, Moser and Serpico, 2005; Condessa, Bioucas-Dias and Kovačević, 2016). But, the proposed SC^{2}S method is simple, easy to use, and can be enhanced to user desired specifications for providing superior classification performance for different LULC classes.

4.2 Problem Statement

A complete MSI/HSI cube with d-spectral channels can be represented as an array of N pixel vectors $(\mathbf{x}_i \in \mathbb{R}^d, i = 1, \dots, N)$ known as a data matrix, say $X = [\mathbf{x}_1\ \mathbf{x}_2 \dots \mathbf{x}_N] \in \mathbb{R}^{d \times N}$. For each pixel vector, there exists a corresponding discrete class label y that belong to the set of all possible classes ${y_i}^{\Psi} \in \{1, \dots, c\}$ (where c is the actual number of information classes) present in the given scene. However, knowing the full extent of information classes in the test site can be difficult, expensive, and is not always possible (Foody and Atkinson, 2002; Bendale and Boult, 2015; Júnior *et al.*, 2017). In such cases, information about only a subset of classes or classes of interest is only available.

Let ${y_i}^{\Omega} \in \{1, \dots, m\}$ be a set of m KCs $(m < c)$ and ${y_i}^{\mho} = \{y \in {y_i}^{\Psi} \middle| y \notin {y_i}^{\Omega}\}$ be a set of $(c - m)$ UCs which are the absolute complement of ${y_i}^{\Omega}$ (i.e., ${y_i}^{\Omega} \cup {y_i}^{\mho} = \emptyset$), then we can define ${y_i}^{\Psi} = {y_i}^{\Omega} \cup {y_i}^{\mho}$. We assume that the classifier is trained with a suitable set of representative training data that belong to the KCs. Our proposed SC^{2}S method aims to reduce the false-known predictions by classifying each pixel vector to one of the KCs (${y_i}^{\Omega}$), otherwise as a UC (${y_i}^{\mho}$).

4.3 Datasets Used

We used four different sources of multi-platform and multi-sensor remote sensing images. One HSI each from AVIRIS, Cubert UHD 185s, ROSIS-3, and one MSI from Sentinel-2A (S2A) sensor. The three outdoor images with different spatial and spectral resolutions are covering several land cover categories and sites. These remote sensing datasets were chosen to form a complete and non-trivial experimental setup to assess the generality of the methods. In order to measure the performance of SC^{2}S algorithms quantitatively, we manually interpreted and collected the ground truth reference samples for Cubert and S2A

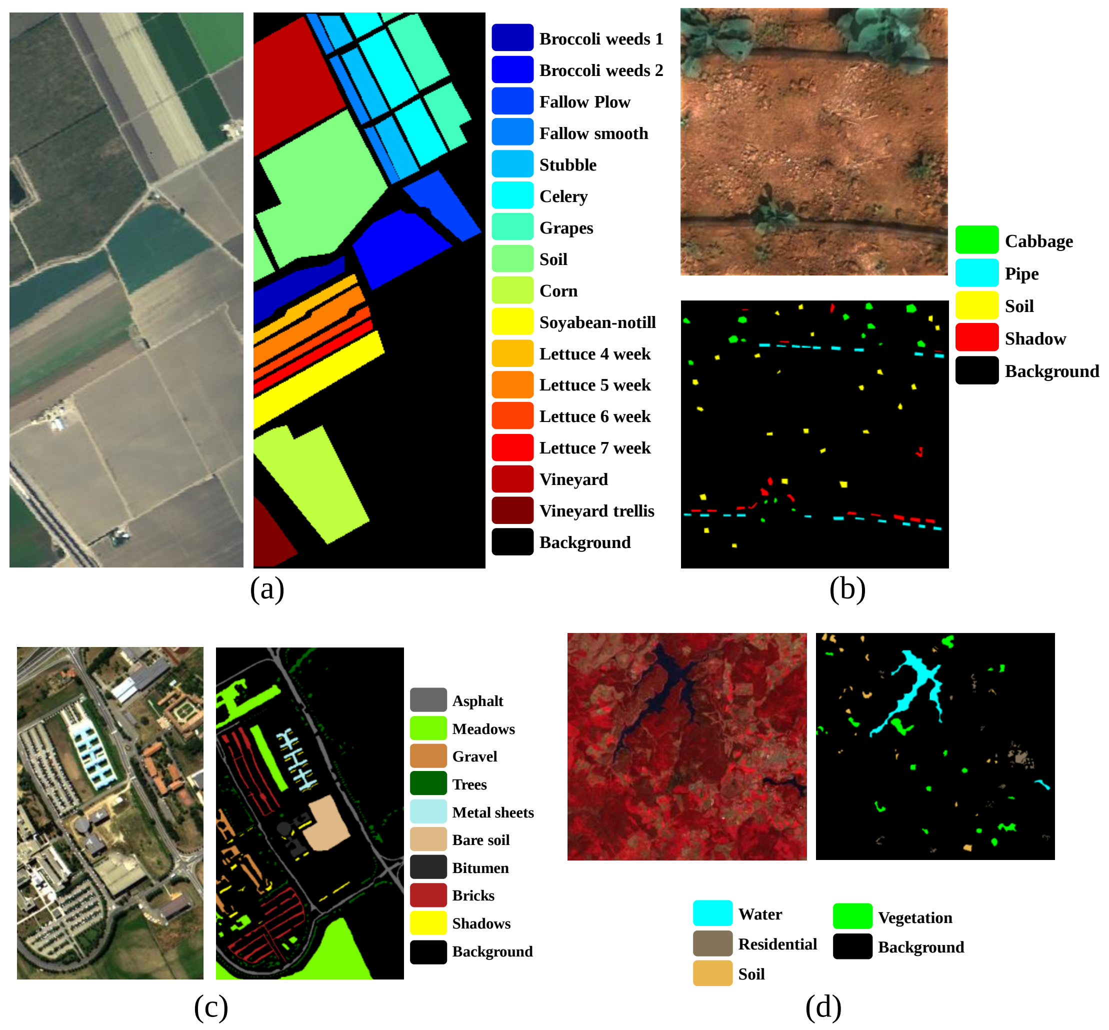


Figure 4.1: (a) RGB display of Salinas HSI and its ground truth reference map; (b) RGB display of Cubert HSI and its ground truth reference map; (c) RGB display of PU HSI and its ground truth reference map (d) FCC display of S2A MSI and its ground truth reference map.

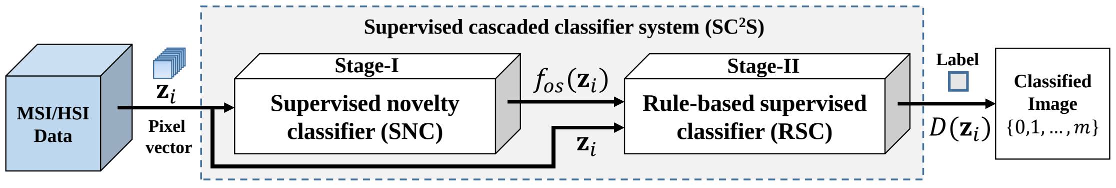
images. Three-band image composites and ground truth maps of the three images are shown in Figure 4.1.

Salinas image: The airborne Salinas HSI was acquired by an AVIRIS sensor flown at low altitude over the city of Greenfield in the Salinas Valley in California, United States of America (USA) on 9th October 1998. This data comprising of 512 × 217 spatial pixels, each having a fine spatial resolution of 3.7 m and 204 spectral bands (after discarding 20 water absorption bands), was collected in the spectral range of 0.36 to 2.5 μm with a nominal spectral resolution of 0.04 μm. The image includes 16 information classes of vegetables, vineyard fields, bare soils, and ground. A total number of 54129 reference ground truth pixels are available for evaluation. Figure 4.1(a) shows the true color composition and ground truth reference map for the Salinas HSI image.

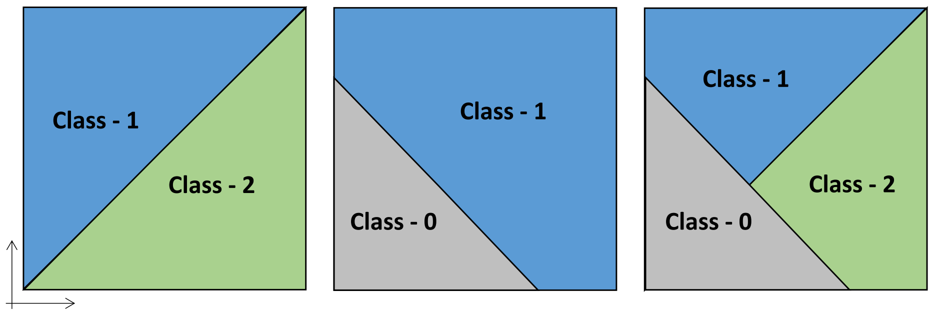
Cubert image: The terrestrial Cubert HSI was acquired over the agriculture crops grown in the experimental plots in the University of Agricultural Sciences, Bengaluru, India, on 5th April 2017. This dataset was captured by the Cubert UHD 185s sensor mounted on the ground-based tripod capturing the top view of the cabbage crop. The Cubert image has 1000×1000 spatial dimensions with an ultra-high spatial resolution of 2 mm, and a total of 139 spectral bands were collected in the spectral range 0.45 to 0.95 μm with a spectral resolution of 0.08 μm. The predominant LULC categories in the study site are cabbage crop, soil, pipe, and shadow. Figure 4.1(b) shows the RGB image and ground truth reference map of the Terrestrial image.

Pavia University (PU) image: The airborne PU HSI is provided by Prof. Paolo Gamba from the TRSL, University of Pavia, Italy. The ROSIS dataset was acquired during a flight campaign by ROSIS-03 sensor, in July 2001, over the University of Pavia, Italy, under cloud-free sky conditions. This data comprising of 610×340 spatial pixels, each having a high-spatial resolution of 1.3 m and 103 spectral bands (after removing the water absorption and bad bands) were collected in the spectral range 0.43 to 0.86 μm with a nominal spectral resolution of 0.04 μm. There are 42776 ground-truth reference samples available for nine information LULC types. Figure 4.1(c) shows the true color composition image and ground truth reference map for the PU image.

S2A image: The spaceborne S2A image was acquired by the Sentinel-2A satellite sensor on 20th March 2018. This image is covering the suburban area of the Mudumalai region, India. This image consists of 711×742 spatial pixels with a spatial resolution of 10 m and has ten spectral bands in the wavelength range 0.04 to 2.19 μm with a varying spectral resolution. Figure 4.2(d) displays the false-color composite (FCC) of the S2A



(a)



(b)

Figure 4.2: (a) Top-level architecture diagram of the proposed pixel-level SC^{2}S method for the i^{th} test pixel vector $\mathbf{z}_i$. (b) Concept diagram of feature space $\mathcal{F}$ partitioning into decision regions by (Left) SVM, (Middle) OCSVM, and (Right) SC^{2}S (OCSVM+SVM) for a 2-KC classification problem.

image and its ground truth reference map. Figure 4.1(d) shows that the S2A MSI contains four dominant information classes: water, residential, soil, and vegetation.

4.4 Methodology

This section introduces and discusses in detail the mechanism of the proposed SC^{2}S algorithm.

4.4.1 Supervised cascaded classifier system (SC^{2}S)

The SC^{2}S approach combines the idea of a novelty classifier, e.g., one-class SVM (OCSVM), with the rule-based supervised classifier and is aimed at minimizing the false-known predictions (see in Figure 4.2 and Figure 4.3). Our key idea behind cascading different types of predictive models is to achieve a diversity of classification models to boost the overall accuracy. In addition, we have introduced an optimum reject decision rule

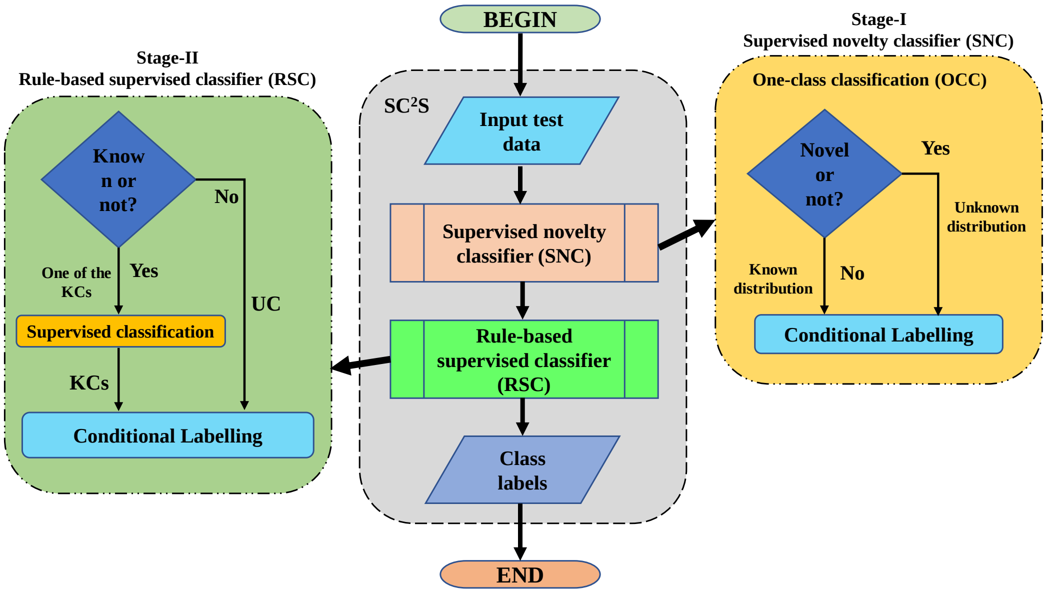

BEGIN
SC²S
Input test data
Supervised novelty classifier (SNC)
Rule-based supervised classifier (RSC)
Class labels
END
Stage-I
Supervised novelty classifier (SNC)
One-class classification (OCC)
Novel or not?
Yes
Unknown distribution
Known distribution
No
Conditional Labelling
Stage-II
Rule-based supervised classifier (RSC)
Known or not?
No
One of the KCs
Yes
UC
Supervised classification
KCs
Conditional Labelling

Figure 4.3: A detailed illustration of general workflow of SC^{2}S algorithmic framework and its two sub-stages using flowchart diagram.

and an extra class label ${y_i}^{\mho} \in \{0\}$ for UC pixel vectors which are sufficiently different from the training pattern. Therefore, in this paper, we precisely define the set of all possible classes as ${y_i}^{\Psi} \in \{0,1,\dots,m\}$ in any given scene. Thus, the SC^{2}S feature space is partitioned into $m+1$ regions for m-KCs, as illustrated in Figure 4.2(b). The abovementioned reasons make the SC^{2}S classifier reliable and robust among other classification algorithms when the classes are not well sampled during the training stage.

The SC^{2}S framework can be generalized to the problem of determining ID or KC and OoD or UC samples. The SC^{2}S method has two steps to using the classifier, namely, training and classification. During training, a straightforward learning scheme is proposed to construct two optimal separating hyperplanes of SC^{2}S (see Figure 4.2(b)) from a given training set $S = \{(\mathbf{x}_i, {y_i}^{\Omega})\}_{i=1}^{n}$, one from the supervised novelty classifier and another from the supervised classifier. In the classification step, a two-stage classification approach for prediction is used.

4.4.1.1. Stage I: SNC

The designed SC^{2}S method uses a supervised novelty classifier as the base classifier for mapping ID and OoD samples (see Figure 4.2(a) and Figure 4.3). Supervised novelty classifier like OCSVM is one of the widely used techniques for anomaly detection, mainly due to their low implementation complexity and computation time (Mũnoz-Marí *et al.*, 2010). Our system uses the OCSVM method to detect novel or anomalous future test data by sorting them as inliers (i.e., ID) or outliers (i.e., OoD).

The key idea in our OCSVM training approach is to consider all the available training data as one-categorical class referred to as inliers, or KCs say ${y_i}^{\Omega} = \{C_i, .., C_j\} \in \{+1\}$ (where $i \neq j$) to be a positive class. In OCSVM classification, inliers are labeled '+1', while outliers or UCs are labeled '-1' (Chang and Lin, 2011). The learning rate and accuracy of the OCSVM method are defined by two hyperparameters: $\nu \in (0,1]$ (upper bound on the fraction of outliers in S), and γ (width of the Gaussian curve). The optimum values of hyperparameters are derived from the annotated training data by performing a grid search using CV. The obtained optimal hyperparameter values are then used to formulate the classification model of OCSVM (Schölkopf *et al.*, 2000). For any given test pixel vector **z**, a supervised novelty classifier produces a decision value $f_{OS} \in \{-1, +1\}$ using the discriminant function (f_{OS}).

4.4.1.2 Stage II: RSC

In the second stage of SC^{2}S, we use a rule-based supervised classifier that uses a decision rule for mapping. Unlike Stage-I, the training procedure in Stage-II follows conventional supervised learning. As shown in Figure 4.2(a), Stage-I is cascaded with Stage-II, building an interconnected system, where the output of Stage-I is the input to Stage-II. There are two inputs to the rule-based supervised classifier, input test pixel vector ($\mathbf{z}$) and the classified label (f_{OS}) from OCSVM (Stage-I). The output of the rule-based supervised classifier D is based on a rule-set for class prediction. The rule-set states that, if f_{OS} is equal to '1', then the pixel vector $\mathbf{z}$ is accepted for classification using supervised classifier otherwise rejected for classification and labeled as '0'. For any given test pixel vector **z**, a rule-based supervised classifier with a discriminant function $D \in y^{\Psi}$ produces a decision value using the optimum decision rule of the following form:

$$\begin{aligned}&\underline{Rule}:\\ &Assign\ D(\mathbf{z}) \rightarrow f_S(\mathbf{z};\boldsymbol{\theta}), \quad if f_{OS}(\mathbf{z};\mathbf{x},\boldsymbol{\alpha},b) = 1 \\ &Otherwise\ D(\mathbf{z}) \rightarrow 0,\end{aligned} \tag{4.1}$$

where "0" represent anomaly class in test data and $f_S(\mathbf{z};\boldsymbol{\theta}) \in y^{\Omega}$ is the discriminant function of a supervised classifier with a set of n-input parameters $\boldsymbol{\theta} = \{\theta_1, .., \theta_n\}$.

4.5 Design of Experimental Setup for Various Real-World Classification Case Studies

In this section, we describe an experimental setup designed exclusively to assess the effectiveness of the proposed SC^{2}S algorithm. The experimental setup consists of three different case studies, each with several test cases of classification scenarios of a varying number of KCs and UCs. In each case study, there are two disjoint sets. Namely, KCs set of size or cardinality m, and UCs set of size $(c - m)$. The information classes in KCs set are used for training the model, and then we use the trained model to perform class predictions on datasets that include samples of all class examples (i.e., KCs and UCs). A brief description of each case study is given as follows.

1. The first case study uses two HSIs, the Salinas dataset consists of an agricultural setting, and the PU dataset has information classes with urban features. As shown in Table 4.1, there are eight test case classification scenarios. Each test case has only two information classes in KCs set, and the remaining classes are considered as UCs. We used SVM and SC^{2}S methods with linear and RBF kernels. As a final result, report the prediction results in terms of OA of only KCs (OA$_{KC}$), OA of both KCs and UCs (OA$_{mix}$), and misclassification rate for each.
2. In the second case study, we use three datasets, namely, Salinas (HSI), Cubert (HSI), and S2A (MSI). As shown in Table 4.3, there are five, three, and two test cases using Salinas, Cubert, and S2A datasets, respectively. There are more than or equal to two information classes in the KCs set in each of the ten test cases, and the

Table 4.1: Summary of the list of eight open-set experimental test cases used in case study I. Each test case contains a specific set of Training classes or KCs and UCs relative to Salinas and PU datasets.

Datasets	Test cases	KCs	UCs	# KC pixels	# UC pixels
Salinas (HSI)	C1	{1,2}	{3,..,16}	5735	48394
	C2	{3,5}	{1,2,4,6,..,16}	4651	49478
	C3	{9,10}	{1,..,8,11,..,16}	9481	44648
	C4	{7,15}	{1,..,6,8,..,14,16}	10847	43282
PU (HSI)	C5	{1,2}	{3,…,9}	25280	17496
	C6	{3,8}	{1,2,4,…,7,9}	5781	36995
	C7	{1,7}	{2,..,6,8,9}	7961	34815
	C8	{4,5}	{1,2,3,6,..,9}	4409	38367

selection of class type is made subjectively. In this case study, we use seven classifiers that belong to one of the two categories of classification methods, such as classifiers without reject-option and with-reject option. As a final result, report OA and AA of each of seven classifiers for each test case.

3. In the third case study, we use a comprehensive evaluation scheme in which the size of the KCs set is preset. Still, the type of information class is selected arbitrarily such that no bias occurs in class type selection. This case study consists of eight different test case scenarios designed using three datasets with an increasing number of KCs (or decreasing number of UCs). A typical use of each test case involves the following three steps. First, select a set of m out of c classes at random and perform model training. Second, use the trained model for class prediction on the dataset that includes all class examples. Finally, repeat steps 1 and 2 for ten random trials for each test case. OA, AA, FNR, and FPR are adopted to quantify the classification performance for the evaluation metrics. The metrics are estimated from the confusion matrix for each realization, and as a final result, the mean and standard deviation (SD) over ten realizations are reported.

4.6 Results and Discussion

This section shows the empirical effectiveness of the SC^{2}S method over the existing classifiers such as RF, one-against-one based SVM, OSNN^{CV}, OSNN^{NNDR}, and P-SVM.

The input data used in the experiments are preprocessed using min-max normalization over the interval [0,1]. We have performed training and classification of all the considered classifiers in the MATLAB environment. During training, only 10 percent of the whole reference data has been randomly selected for training. For the RF classifier, the number of trees is set to 30. The number of splits is set to $(n-1)$, where n denotes the total number of training samples (Ghamisi *et al.*, 2017). In terms of the SVM, P-SVM, and OCSVM, the linear kernel used only in case study I and RBF kernel is used in case studies I, II and III.

The hyperplane parameters $\boldsymbol{\theta} = \{C, \gamma, \nu\}$ (C is regularization parameter) have been searched in the ranges of $C = \{10^{-1}, \dots, 10^{5}\}$ in powers of 10 steps, $\gamma = \{1,2, \dots, 50\}$ and $\nu = \{10^{-2}, \dots, 10^{-1}\}$ in the step of 0.01 using grid search. For a detailed explanation of the classifiers, see chapter 2. The selection of $T_r = \{0, 0.01, \dots, 0.99\}$The selection of optimal thresholds $T_r = \{0, 0.01, \dots, 0.99\}$ (Farfan-Escobedo, Enciso-Rodas and Vargas-Munoz, 2017; Júnior *et al.*, 2017), $T_p = \{0, .51, 0.52, \dots, 1\}$ (Platt, 1999; Scheirer, Jain and Boult, 2014) and model parameters are tuned during the training stage using grid search based on a 10-fold CV. The prediction performance of the considered classifiers has been investigated and compared by performing class prediction on the data comprising samples from both trained and untrained classes. The objective is to measure the number of false-known predictions produced by a classifier. The classification accuracies obtained by different classifiers are reported in Tables 4.2, 4.3, 4.4, and 4.5. The results reveal interesting observations about the incorrect predictions often produced by the classifiers.

4.6.1 Performance evaluation of the list of classifiers using case study I

The quantitative OA results of SVM and SC^{2}S methods for eight test cases (C1 to C8) are shown in Table 4.2. From a detailed analysis of Table 4.2, we notice that, in the majority of the test cases of closed-set setting, our method obtained comparable OA$_{KC}$ and misclassification rates with SVM. But, the proposed method significantly outperformed SVM more consistently for all the test cases of open-set settings. Unlike the proposed method, SVM showed erratic behaviors, where the misclassification rate is very high for

Table 4.2: OA (in %) and misclassification rate (in %) results to evaluate the prediction performance of conventional SVM and SC^{2}S classifier with linear and RBF kernels for the specific open-set test case scenarios in case study I. The best accuracy and error rate in each row are shown in bold.

Test Cases	**Kernel**	**Prediction without UCs (closed-set setting)**				**Prediction with UCs (open-set setting)**					
		OA$_{KC}$ (%)		**Misclassification rate (%)**		**OA$_{mix}$ (%)**		**Misclassification rate (%)**		**# UC classified**	
		SVM	**SC^{2}S**	**SVM**	**SC^{2}S**	**SVM**	**SC^{2}S**	**SVM**	**SC^{2}S**	**SVM**	**SC^{2}S**
C1	**L**	99.89	99.69	0.11	0.31	10.58	**84.24**	89.42	**15.76**	0	**39893**
	R	99.86	99.51	0.14	0.49	10.58	**88.40**	89.42	**11.60**	0	**42145**
C2	**L**	99.26	99.20	0.74	0.80	17.39	**28.56**	82.61	**71.44**	0	**6065**
	R	99.98	99.29	0.02	0.71	20.04	**26.14**	79.96	**73.86**	0	**3379**
C3	**L**	99.26	99.20	0.74	0.80	17.39	**28.56**	82.61	**71.44**	0	**6065**
	R	99.26	98.46	0.74	1.54	17.36	**48.96**	82.64	**51.04**	0	**17169**
C4	**L**	99.89	99.82	0.11	0.18	20.04	**22.92**	79.96	**77.08**	0	**1600**
	R	99.98	99.29	0.02	0.71	20.04	**26.14**	79.96	**73.86**	0	**3379**
C5	**L**	99.87	99.75	0.13	0.25	59.02	**61.28**	40.98	**38.72**	0	**1025**
	R	99.83	98.96	0.17	1.04	59	**60.86**	41	**39.14**	0	**1016**
C6	**L**	87.16	86.96	12.84	13.04	11.7	**24.7**	88.30	**75.30**	0	**5568**
	R	87.16	86.58	12.84	13.42	11.78	**34.60**	88.22	**65.40**	0	**9794**
C7	**L**	90.35	90.28	9.65	9.72	16.4	**19.11**	83.60	**80.89**	0	**1055**
	R	90.35	88.98	9.65	11.02	16.82	**19.11**	83.18	**80.89**	0	**1092**
C8	**L**	99.93	99.82	0.07	0.18	10.30	**21.58**	89.70	**78.42**	0	**4833**
	R	99.93	99.23	0.07	0.77	10.30	**33.58**	89.70	**66.42**	0	**9990**

Note: **L** and **R** represents linear and RBF kernel, respectively.

open-set scenarios compared to closed-set scenarios. There is a minimum of 0.06% and a maximum of 0.31% accuracy difference between linear-SVM and linear-SC^{2}S classifier predictions for KCs (OA$_{KC}$) alone.

In contrast, it is about 2.26% and 73.66% for the mixed set (OA$_{mix}$) that presents an actual scenario consisting of both KCs and UCs. Similarly, in the case of accuracy difference between RBF-SVM and RBF-SC^{2}S predictions, there is a minimum and maximum of 0.35% and 1.37% for KCs set; 1.86% and 77.82% for the mixed set, respectively. The obtained results indicate that the proposed linear-SC^{2}S method yields an OA approximately equal to linear-SVM for KC cases and significantly outperforms linear-SVM for mixed cases. Interestingly, there is no substantial difference between linear-SVM and RBF-SVM with regard to OA is concerned. However, based on the experimental

Table 4.3: OA (in %) and AA (in %) results obtained for ten open-set test cases (in case study II) by the list of classifiers having without reject-option and with reject-option. The best accuracies in each row are shown in bold.

Datasets	Test case	Prediction for the available reference data (both trained and untrained classes) : Accuracy in [%]													
		Classifiers without reject-option				Classifiers with reject-option									
		RF		SVM		OSNN^{CV}		OSNN^{NNDR}		P-SVM		SC^{2}S (OCSVM+RF)		SC^{2}S (OCSVM+SVM)	
		OA	AA	OA	AA	OA	AA	OA	AA	OA	AA	OA	AA	OA	AA
Salinas	C1	10.55	66.36	10.57	66.41	10.50	65.85	14.24	67.05	58.62	81.72	99.77	98.19	**99.78**	**98.22**
	C2	26.41	82.96	26.41	82.02	49.15	87.70	61.16	90.41	83.77	95.47	95.37	97.41	**95.37**	**97.44**
	C3	50.49	88.89	50.97	90.30	57.07	89.19	78.10	92.84	66.45	92.38	92.34	95.83	**92.81**	**97.23**
	C4	8.99	77.92	9.11	79	22.66	79.24	26.97	81.62	45.82	84.61	78.61	92.71	**78.70**	**93.49**
Cubert	C5	31.87	79.37	31.93	79.74	34.02	79.40	33.46	79.47	34.12	80	71.94	90.14	**72**	**90.49**
	C6	51.45	66.58	51.52	66.67	51.90	66.93	52.31	67.21	82.18	87.52	88.21	91.40	**88.28**	**91.50**
	C7	49.93	66.59	49.94	66.60	50.48	66.94	51.92	67.93	70.68	80.38	82.42	87.60	**82.42**	**87.60**
	C8	76.08	74.41	76.61	74.91	76.16	74.49	76.53	74.88	74.67	73.05	91.29	90.81	**91.87**	**91.35**
S2A	C9	45.35	65.47	45.69	65.95	97.55	97.55	99.01	98.55	94.68	93.23	98.90	98.39	**99.24**	**98.87**
	C10	77.23	66.44	77.37	66.53	77.60	66.67	80	70.47	84.75	77.48	92.02	89.38	**92.02**	**89.38**

Note: The test cases are C1={1,2}, C2={1,2,6,7,14}, C3={1,2,6,7,9,…,14}, C4={11,12,13,14}, C5={3,4,5,8}, C6={1,3}, C7={1,2}, C8={1,2,3}, C9={2,3}, and C10={1,3}.

results, the RBF-SC^{2}S offers superior performance yielding an OA much higher than linear-SVM, linear-SC^{2}S, and RBF-SVM for the mixed set (OA$_{mix}$).

4.6.2 Performance evaluation of the list of classifiers using case study II

In case study-II, the prediction performance of the seven different classifiers has been investigated and compared for ten test cases of open-set scenarios. The classification accuracies obtained by seven classifiers are reported in Table 4.3. The obtained results reveal interesting observations about the incorrect predictions often produced by the classifiers. It can be observed in Table 4.3 that there is a considerable difference between OA and AA results consistently for all cases. This significant difference can be ascribed due to the presence of unbalanced samples in the datasets. For Salinas data, SC^{2}S (OCSVM+SVM) produced the maximum OA of 99.78% and AA of 98.22%, compared to the SC^{2}S (OCSVM+RF) method, there was more than 0.01% improvement in both OA and AA. Similarly, for Cubert and S2A data, SC^{2}S (OCSVM+RF) and SC^{2}S (OCSVM+SVM) produced the highest accuracies than RF, SVM, OSNN^{CV}, OSNN^{NNDR}, and P-SVM. This

Table 4.4: The mean and SD of OA (in %) results over ten realizations are presented for eight open-set scenarios (in case study III) by the list of classifiers having without reject-option and with reject-option. The best results in each row are shown in bold.

Datasets	Test cases	Classification scenarios		Prediction for the available reference data (both trained (or KCs) and untrained classes (or UCs))						
				Classifiers without reject option		Classifiers with reject option				
				RF	SVM	OSNN^{CV}	OSNN^{NNDR}	P-SVM	SC^{2}S (OCSVM+RF)	SC^{2}S (OCSVM+SVM)
				OA [%]	OA [%]	OA [%]	OA [%]	OA [%]	OA [%]	OA [%]
		KCs	UCs	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Salinas	**C1**	2	14	12.84 (5.62)	12.88 (5.62)	22.02 (8.15)	50.60 (25.41)	53.90 (31.88)	76 (14.35)	**76.03 (14.33)**
	C2	5	11	29.15 (7.74)	30.34 (9.01)	38.86 (7.78)	39.33 (13.09)	51.47 (14.69)	68.31 (10.55)	**69.49 (10.39)**
	C3	8	8	46.46 (7.93)	48.20 (8.44)	53.56 (5.74)	51 (7.54)	54.68 (6.78)	62.98 (4.30)	**64.69 (4.05)**
	C4	12	4	61.59 (5.85)	73.37 (6.51)	70.58 (5.29)	70.74 (4.84)	75.55 (6.21)	69.29 (5.62)	**80.92 (6.95)**
Cubert	**C5**	2	2	49.65 (1.30)	49.99 (1.37)	50.83 (1.51)	60.21 (10.86)	73.86 (14.91)	84.77 (5.80)	**85.11 (6.21)**
	C6	3	1	74.06 (1.45)	75.65 (1.40)	75.74 (1.42)	77.04 (1.58)	86.55 (7.37)	91.13 (3.37)	**92.72 (4.04)**
S2A	**C7**	2	2	43.95 (14.37)	44.08 (14.17)	44.85 (13.63)	65.91 (27.73)	85.72 (22.14)	91.80 (4.25)	**91.93 (4.45)**
	C8	3	1	68.32 (20.96)	70.08 (19.36)	69.85 (19.36)	70.68 (20.23)	84.63 (17.13)	93.06 (3.49)	**94.87 (5.09)**

trend is particularly due to the inclusion of a reject option and an extra-label. The higher the OA and AA will be, the lower the false-known errors.

Interestingly, there is no substantial difference between RF and SVM regarding OA and AA. The comparatively lower accuracy for the C5 can be ascribed to the presence of substantially sparse areas of vegetable crops with soil backgrounds. However, based on the experimental results, the SC^{2}S using SVM as supervised classifiers indicates superior performance yielding an OA and AA substantially higher than the accuracies among all the seven classifiers (see Table 4.3).

4.6.3 Performance evaluation of the list of classifiers using case study III

Tables 4.4 and 4.5 summarize the results of seven different classification algorithms in terms of mean and SD of accuracies over ten trials. A high mean of OA and AA values indicate low false prediction errors. A low SD says that the actual accuracy tends to be close to the mean of OA and AA. The classifiers without rejection, such as RF and SVM, produced the lowest mean (SD) of OA and AA confirming the high false prediction errors

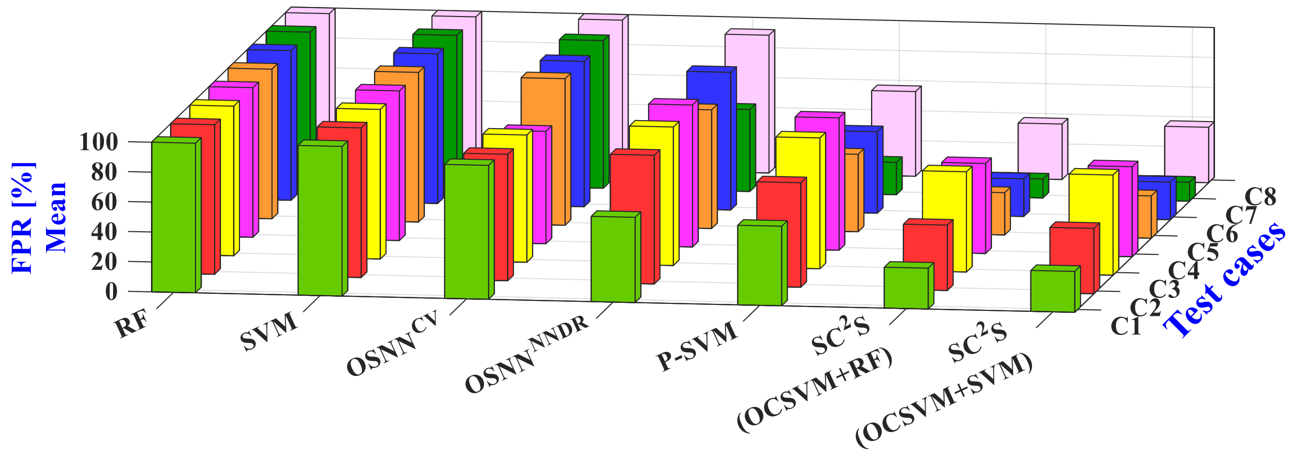
Table 4.5: The mean and SD of AA (in %) results over ten realizations are presented for eight open-set scenarios (in case study III) by the list of classifiers having without reject-option and with reject-option. The best results in each row are shown in bold.

Datasets	Test cases	Classification scenarios		Prediction for the available reference data (both trained (or KCs) and untrained classes (or UCs))						
				Classifiers without reject option		Classifiers with reject option				
				RF	SVM	OSNN^{CV}	OSNN^{NNDR}	P-SVM	SC^{2}S (OCSVM+RF)	SC^{2}S (OCSVM+SVM)
				AA [%]	AA [%]	AA [%]	AA [%]	AA [%]	AA [%]	AA [%]
		KCs	UCs	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
Salinas	C1	2	14	66.39 (0.44)	66.38 (0.40)	71.34 (6.55)	83.27 (12.37)	80.67 (9.19)	90.42 (7.19)	**90.45** **(7.19)**
	C2	5	11	81.87 (2.13)	82.55 (1.53)	82.96 (4.04)	86.49 (5.85)	86.68 (3.88)	89.35 (3.99)	**90.03** **(3.61)**
	C3	8	8	85.56 (2.61)	86.74 (1.91)	85.34 (4.72)	85.31 (4.54)	88.78 (3.70)	88.91 (2.93)	**90.08** **(2.56)**
	C4	12	4	88.62 (1.52)	90.25 (1.25)	87.03 (2.97)	87.85 (2.95)	90.42 (1.64)	91.03 (1.47)	**92.65** **(1.47)**
Cubert	C5	2	2	66.24 (0.28)	66.63 (0.02)	67.75 (0.73)	73.72 (7.23)	73.59 (5.52)	91.72 (3.32)	**92.11** **(3.52)**
	C6	3	1	74.51 (0.22)	74.93 (0.06)	75.62 (1.18)	80.53 (6.86)	87.62 (3.36)	88.38 (7.08)	**88.81** **(7.29)**
S2A	C7	2	2	66.53 (0.41)	66.62 (0.09)	67.63 (1.94)	77.58 (13.05)	79.36 (13.18)	92.98 (8.23)	**93.08** **(8.31)**
	C8	3	1	74.51 (0.42)	74.87 (0.09)	76.05 (2.53)	83.21 (9.76)	88.08 (9.66)	87.85 (8.46)	**88.21** **(8.66)**

in all test cases. In this case of Salinas data, SC^{2}S (OCSVM+SVM) produced the highest mean OA of 80.092% and AA of 91.80% for C5 when compared with SC^{2}S (OCSVM+RF) there is more than 0.01% improvement. A similar trend can be observed for Cubert and S2A data, where the SC^{2}S algorithm produced the highest mean OA and AA than RF, SVM, OSNN^{CV}, OSNN^{NNDR}, and P-SVM.

To assess the error rates of all experiments in this case study, 3D bar plots illustrating the statistical measures like mean FNR (%) and mean FPR (%) are shown in Figure 4.4. Note, the FNR and FPR are estimated as two-class classification problems, i.e., KC or UC. It can be observed that RF and SVM resulted in 0% FNR and 100% FPR for all the cases, which means that all of the untrained UC samples are confidently classified incorrectly as one of the KCs. Interestingly, there is no substantial difference between RF and SVM concerning OA, AA, FPR, and FNR. Contrarily, the classifiers with-reject option (i.e., OSNN^{CV}, OSNN^{NNDR}, P-SVM, and SC^{2}S) are relatively able to identify more UC samples with less than 10% FNR and less than 100% FPR. This increasing trend is mainly due to the inclusion of the reject option with an extra-label. Moreover, the FNR of SC^{2}S can yield

(a)



(b)

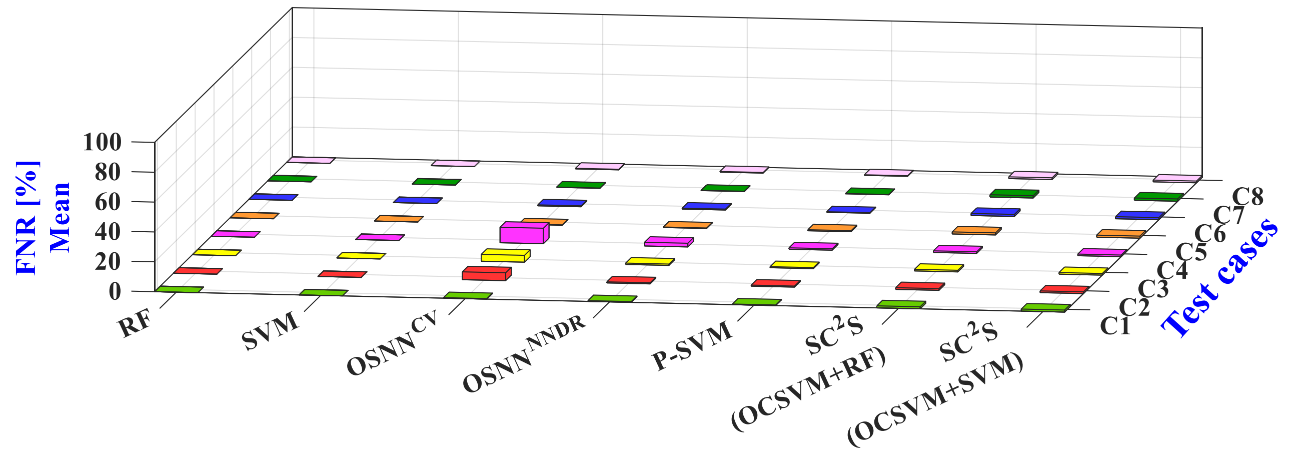


Figure 4.4: 3D bar plot visualization of obtained mean percentages of (a) FPR and (b) FNR results by seven different classification algorithms for eight experimental open-set test cases in case study III.

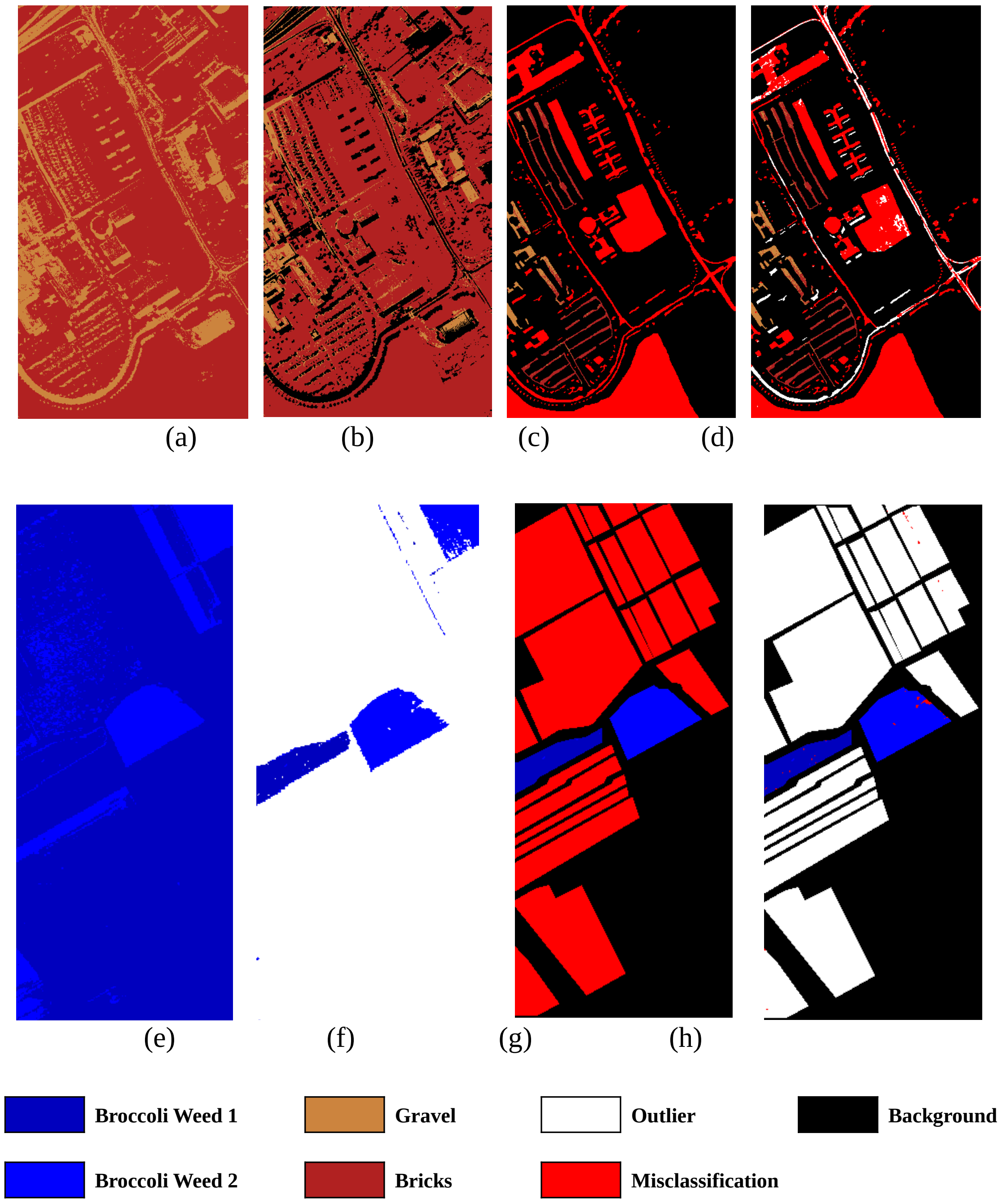


Figure 4.5: Classification maps obtained for specific test case by SVM in (a) and (e); (b) and (f) by SC^{2}S (OCSVM+SVM) prediction for full PU image (above) and Salinas image (below). A detailed representation of classified maps to assess false-known errors (in red) are obtained by SVM (c) and (g); (d) and (h) by SC^{2}S prediction only for available ground-truth reference data.

0% if the model learning is done on the representative training set of KCs. In all the test cases, training set S was selected at random. However, based on the obtained results, SC^{2}S using SVM as a supervised method indicates superior performance that yielded a higher OA (>64.50%), AA (>89.50%), lower FPR (<67%), and relatively low FNR (<1.6%) among the considered seven classification techniques (see Table 4.4, Table 4.5, and Figure 4.4).

For illustrative purposes, Figure 4.5 shows the classification maps obtained by SVM and SC^{2}S (OCSVM+SVM) classifiers for the specific realization of test cases C3 for PU in case study I and C1 for Salinas data in the case study I or II or III. In addition, Figures 4.5 (c), (d), (f), and (g) provide better insights into the confident errors produced by the classifiers, especially for SVM. In this context, the training set of C1 included 574 pixel vectors of two KCs (i.e., broccoli weeds 1 and broccoli weeds 2), whereas the testing set included 54129 (5735 of KCs and 48394 of UCs) samples of all 16 classes. The OA, AA, FPR, and FNR results of SVM are 10.58%, 66.60%, 100%, and 0%, while SC^{2}S yielded 99.82%, 99.05%, 0.02%, and 1.48%, respectively. Overall, the classification maps shown in Figure 4.5 indicate the positive effect of classification having a reject-option for minimizing the false positives for UCs.

Unlike (Mantero, Moser and Serpico, 2005; Jun and Ghosh, 2013; Condessa, Bioucas-Dias and Kovačević, 2016; Júnior *et al.*, 2017), we have proposed a fully supervised non-parametric multi-class classifier system that is simple, easy to integrate and implement classification of remote sensing data in the presence of UCs. In practical applications, a pattern recognition system should avoid false predictions (Chow, 1970). Based on these observations, we can conclude that the results across the ten cases are consistent with the SC^{2}S algorithm in providing reliable and robust thematic accuracy

4.7 Chapter Conclusions

In this paper, we proposed a novel classification algorithm called the SC^{2}S for the multi-class classification even in the presence of many UCs. This study has evaluated the importance and advantage of including a class label for grouping anomalous instances that have a significant deviation from training distribution. Comparing the performance of SC^{2}S against classifiers with and without reject option indicates that the proposed SC^{2}S method effectively provides trustworthy classification results with significantly lower error rates. The proposed SC^{2}S method has potential applications in the classification of remote sensing imagery from which fewer information classes are to be retrieved while rejecting the presence of a large number of spectral classes or UCs.

CHAPTER 5

SHADOW AND ILLUMINATION INVARIANT CLASSIFICATION USING MULTISPECTRAL AND HYPERSPECTRAL DATA

Prelude: In this chapter, a set of new categories of supervised image classification algorithms are proposed for shadow and illumination invariant class prediction using spectral information from MSIs or HSIs. In the first set, we present a novel version of two information-theoretic spectral similarity measures. The second set consists of a two-stage system called shadow and illumination invariant classifier system (SI^{2}CS). In order to evaluate the proposed three different algorithms, we designed an experimental setup consisting of several classification scenarios with a varying range of shadow and illumination complexities. The overall performance is measured in terms of classification accuracies and thematic classification maps obtained from several experimental scenarios.

5.1 Introduction

Optical images are one of the prime sources of remote sensing data for various applications across a range of scientific and engineering fields (Eismann, 2012; Taneja, Ballan and Pollefeys, 2015). Examples of such applications include, but are not limited to, environment monitoring, weather forecasts, hydrological management, military, and planetary exploration (Sabins, 1999; Clark *et al.*, 2003; Khan *et al.*, 2018). Optical remote sensing technology is continuously advancing to meet the current and future application

This chapter will be published as an article and currently is in review in *Pattern recognition*, with the title: “Shadow and Illumination Invariant Classification for High Resolution Images”. Authors: Dubacharla Gyaneshwar and Rama Rao Nidamanuri.

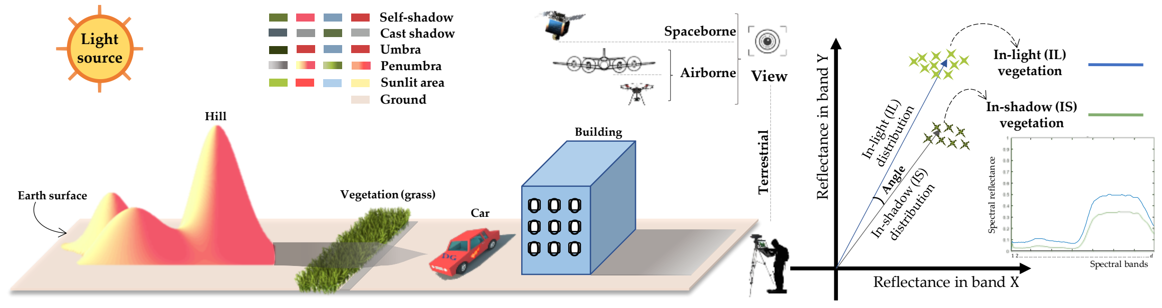


Figure 5.1: Illustrating different types of shadows. The feature space plot in the right represents the 2-D scatter plot and spectral reflectance curves of the vegetation reflectance in the area exposed to light and dark regions where illumination from a light source is blocked.

requirements. The technological advances are ranging from observation practices to producing high-quality images with different resolutions. For example, the interest to non-invasively map materials from a remote location has resulted in spectral imaging techniques (Clark *et al.*, 2003; Akhtar and Mian, 2018). Unlike traditional imaging, spectral imaging systems produce images like MSIs and HSIs with rich spectral information along with spatial data. Both MSIs and HSIs are produced by passive sensors that rely on the source of scene illumination and incoming radiation, similar to other traditional optical images. Because of the higher level of spectral detail than MSIs, HSIs provide the advantage of accurately characterizing a wide range of materials from the surface reflectance using techniques from domains like machine learning, information theory, and so on. However, both MSIs and HSIs have merits and demerits based on the application selected.

Despite the success achieved by image analysis (including MSI/HSI) in many remote sensing applications, there still exists a number of significant challenges and uncertainties that can hinder the success (Dubacharla and Nidamanuri, 2021). For example, the information in the imagery data collected by the optical sensors is influenced by uncertainties from adverse environmental effects such as shadows and changing lighting conditions. Due to these unavoidable uncertainties in any given non-trivial target scenes, the performance of analysis techniques like classification, detection, etc., will be unstable and may cause serious problems (Adler-Golden *et al.*, 2001; Sanin, Sanderson and Lovell, 2012; Rüfenacht, Fredembach and Süsstrunk, 2014). Hence, there is a pressing need to address these challenges and issues, which require careful consideration and analysis.

A shadow, shade, or an unlit area is a natural phenomenon that has always existed. As shown in Figure 5.1, they are commonly observed optical events in any given scene depending on the sensor view angle, illumination, and scene geometry. Whereas, the presence of different lighting conditions in the location of study can be caused by changes in the ambient lighting environment, for example, due to rapid differences in the weather conditions such as cloudy, sunny, and shade. A shadow can be grouped into one of the two categories, namely, cast shadow (CS) and self-shadow (SS). A CS is an object's adjacent shadow in the direction of the light source, and a SS is an unilluminated area of the object itself. In addition, there are mainly two different regions in a shadow, such as an umbra and a penumbra (Dare, 2005). The dark region of a shadow is referred to as umbra, and the transition region around the umbra is referred to as penumbra (see Figure 5.1).

In many practical image-driven applications, shadows are a severe problem because they limit the information mining and image interpretation tasks by occluding the target objects or study scene. For example, a conventional supervised classification model is built using a training set consisting of in-light (IL) KC samples that are exposed to a sufficient lighting condition. However, the trained classification model deployed in real-world application environments often faces test cases arising either from IL-KC distribution or UC distribution that may or may not be exposed to bright or dim illumination. The UC distribution means samples belonging to out-of-KC training distribution, unseen class samples during training, and KCs either exposed to bright or dim light (Júnior *et al.*, 2017; Dubacharla and Nidamanuri, 2021). As a result, in such shadow and changing light conditions, most classifiers fail to classify the shaded or in-shadow (IS) KC examples and produce errors (i.e., false negatives). This erroneous result is more common in MSI/HSI analysis due to spatial pixel value and spectral reflectance curve distortions. Thus, the development of new algorithms capable of performing shadow and illumination invariant detection and classification is critical for accurate remote sensing image analysis because their presence skews the image information, especially for MSIs/HSIs.

This chapter presents new categories of supervised algorithms for shadow and illumination invariant image classification using MSIs and HSIs. The first category consists of two information-theoretic spectral similarity measures, namely SCM and SID. These similarity measures use an adaptive threshold search technique to perform invariant class prediction. The second category has a novel two-stage cascade classifier system named shadow and illumination invariant classifier system (SI^{2}CS). The SI^{2}CS is developed by combining two diverse supervised models: 1) OCC and 2) rule-based similarity matching (RSM). In the first stage of SI^{2}CS, OCC correctly maps the training distribution samples of KC (i.e., IL-KC samples) as one group and unseen UC samples as another group. The UC group consists of IS-KCs like SS-KCs (i.e., unlit portions of the KCs) and CS-KCs (shadows cast by a UC or KC object on the KC). In the second stage, RSM uses a decision rule to allow a test input to SMM or not. In this study, we use SCM and SID as SMM, but the flexibility of SI^{2}CS is that other matching techniques can also be utilized as SMM in RSM. The idea behind using SMM is that we can discriminate an IS-KC sample from a UC sample. The reason for this is because a minimum of 50% spectral signature matches between a typical IL-KC sample and an IS-KC sample (Zhai *et al.*, 2019). Figure 5.1 shows this similarity trend between different vegetation spectra, i.e., IL vegetation and IS vegetation spectral signatures. So, we can exploit this similarity match between IL-KC and IS-KC for invariant class prediction. The cascading architecture in SI^{2}CS is designed to unify the decisions of two different models. In this way, the current chapter presents four algorithms of two different categories, i.e., two proposed versions of SMMs and SI^{2}CS with different SMM.

The previous studies in the literature have mainly dealt with intending to perform either shadow detection or removal and classification by primarily using physical or texture properties in the image (Adler-Golden *et al.*, 2001; Dare, 2005; Sanin, Sanderson and Lovell, 2012). But very few studies were reported for shadow-invariant detection or classification using only spectral information from HSIs/MSIs. Unlike the previous studies, our approaches are quite simple and easy to implement and jointly aim at performing shadow and illumination invariant class predictions. In particular, the advantage of our methods is the use of the same training set required for a supervised approach and performs invariant class prediction with better accuracy. This study is the first of a kind that presents shadow and illumination invariant algorithms to handle the complex shadow regions and lighting settings in MSIs as well as in HSIs.

This chapter is organized into six sections. The following section presents the research problem statement, while section 5.3 briefly describes the datasets used in our analysis of this chapter. Section 5.4 presents the three proposed methodologies and frameworks used to investigate this chapter's objectives. Section 5.5 describes and discusses the experimental results obtained. Finally, the last section presents the conclusions of this chapter.

5.2 Problem Statement

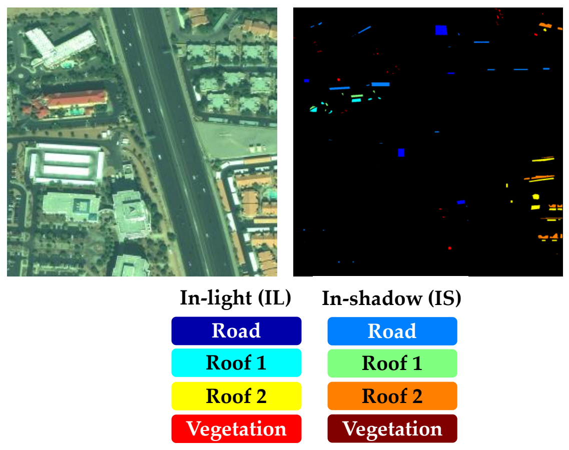
Let $\mathbf{X} \in \mathbb{R}^{a \times b \times d}$ represent an MSI/HSI with a rows or lines, b columns or samples, and d spectral channels. Let $\mathbf{x}_i \in \mathbb{R}^d$ (where $i = 1, \dots, (a \times b)$) be the i^{th} pixel vector in $\mathbf{X}$, y_i be the corresponding KC label where $y_i \subseteq y^{\Psi}$ and y^{Ψ} is the set of c number of actual information classes in a given image or scene, say $y^{\Psi} = \{1, \dots, c\}$. Then the objective of a supervised classification approach is to map each pixel vector $\mathbf{x}_i$ to corresponding y_i, say $f\colon \mathbf{x} \to y$ where f is the decision function learned during training. However, in practice, collecting the full extent of actual information classes in any given scene of study before classification is quite difficult, costly. It may not be possible in some cases. In such cases, we have information on the classes of interest referred to as KCs or IL-KCs. So, the incomplete knowledge of the actual classes during training would lead to prediction errors because of the forced assignment problem in multiclass classification. Thus, in these circumstances, OCC is a suitable method because it does not suffer from the forced assignment problem and results in relatively low errors.

Yet, as previously mentioned, a real-world classification task has uncertainties other than UCs, namely, shadows and varying illumination effects. The existing methods are sensitive to IS-KC regions and produce omission errors. Examples of IS-KC samples are CS-KC, SS-KC, KC umbra, and KC penumbra. Let $y^{\Omega} \in \mathbb{R}^{m}$, $y^{\mho} \in \mathbb{R}^{(c-m)}$, and $y^{\Phi} \in \mathbb{R}^{m}$ denote the class labels of IL-KC, UCs, and IS-KC, respectively. Then, we can define the set of all possible class labels in any given target scene based on the spectra as $y^{\Psi} = y^{\Omega} \cup y^{\mho} \cup y^{\Phi}$ and $y^{\Omega} \cap y^{\mho} \cap y^{\Phi} = \emptyset$. Due to the evident spectral signature mismatch between the IL-KC (y^{Ω}) and IS-KC (y^{Φ}) spectra, most classification methods fail to map them as a single class even though both belong to the same target class material. On the contrary, our methods are capable of grouping y^{Ω} and y^{Φ} together, i.e., $y^{\Phi} = y^{\Omega} = \{1\}$ and $y^{\mho} = \{0\}$ for an OCC problem. We assume that the classifier is trained with a representative and diverse set of training set $S = \{\mathbf{x}_i, y_i^{\Omega}\}_{i=1}^{n}$ samples that belong to the IL-KC set, i.e., $y_i^{\Omega} \in \{1\}$. Then our proposed algorithms aim to reduce errors, especially omission errors or false-negatives, for candidate shadow pixels by classifying each pixel vector as UC ($y^{\mho}$), otherwise as KC (y^{Ω}).

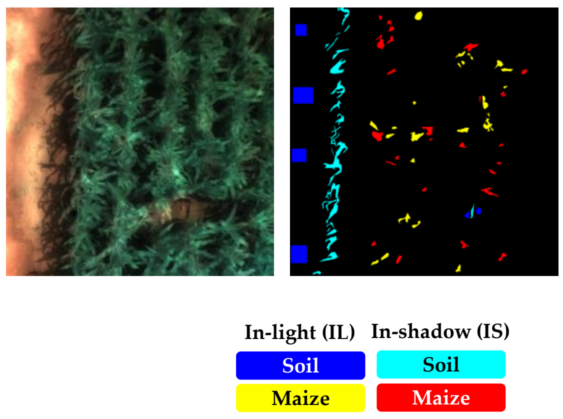
5.3 Datasets Used

We used six different sources of multi-platform and multi-sensor spectral images (one MSI from Worldview-3 (WV-3) spaceborne platform, two HSIs from Cubert UHD 185s with an airborne and a terrestrial platform, one HSI each from a terrestrial platform equipped with Nuance FX, Specim-IQ, and Visible sensors). The six indoor-outdoor images with different spatial and spectral resolutions cover several land cover categories and sites with varying complexities of shadows and illumination effects. The six imagery datasets were selected to form a complete and non-trivial experimental setup to assess the generalization of the methods. To measure the performance of algorithms quantitatively, we manually interpreted and collected the ground truth reference samples for IL and IS regions. True color composites and ground truth maps of the six images are shown in Figure 5.2.

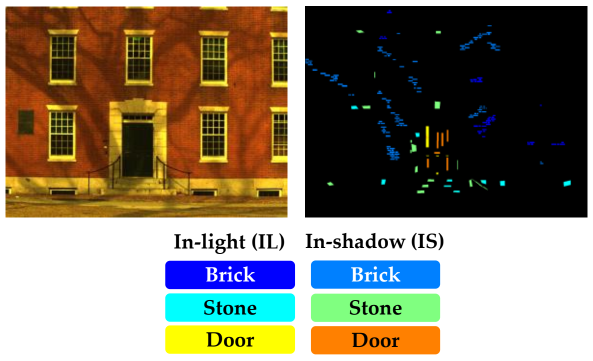
WV-3 image: The spaceborne WV-3 MSI is from the DigitalGlobe and distributed by the SpaceNet (SpaceNet on Amazon Web Services (AWS)., 2021). The WV-3 dataset



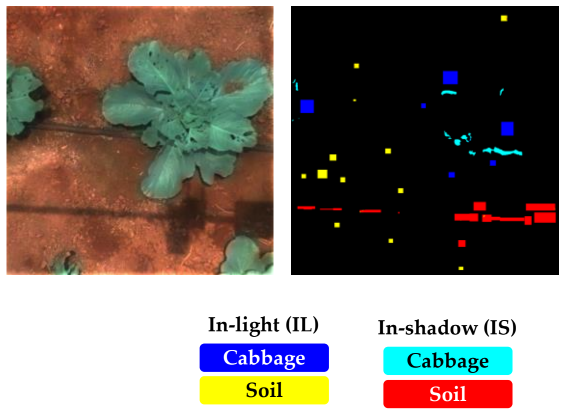
(a)



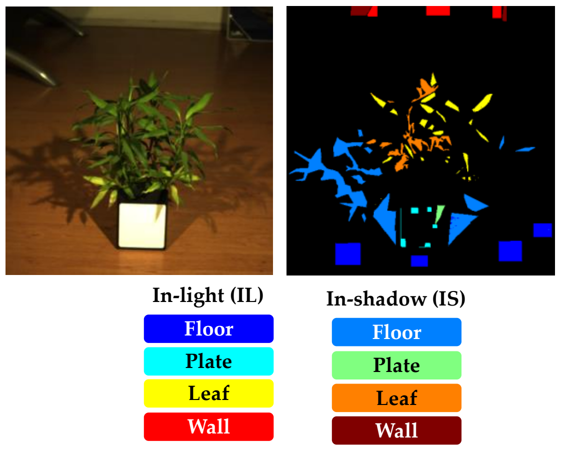
(b)



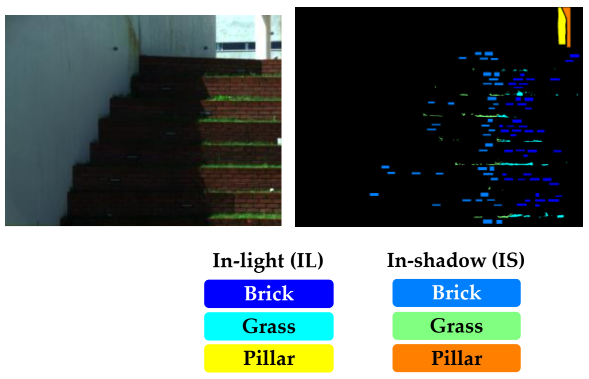
(c)



(d)



(e)



(f)

Figure 5.2: Three band true color composite images with manually labeled IL and IS ground truth samples of the dataset used for the experiments. (a) Spaceborne WV-3 MSI. (b) Airborne Cu-D HSI. (c) Terrestrial Harvard HSI. (d) Terrestrial Cu-T HSI. (e) Indoor Specim HSI. (f) Terrestrial GS HSI.

was acquired, on 15th October 2015, over the city of Las Vegas, United States of America (USA), under cloud-free sky conditions. WV-3 is pan-sharpened satellite imagery comprising of 1300×1300 spatial pixels, each having a very high-spatial-resolution of 30 cm, and eight spectral bands were collected in the spectral range 0.4 to 1.04 μm with a varying spectral resolution of 0.04 to 0.68 μm. The number of ground truth reference pixels for IS and IL are 11485 and 12601, respectively. The dominant LULC types in the heterogeneous urban landscape study site include impervious surfaces, trees, and shadows. So, there are four information classes and eight spectral classes in WV-3 MSI. Figure 5.2(a) shows the overlay of the ground truth map on the true color composition image, where the shadow strength in WV-3 is medium and its size is moderate.

Cubert drone (Cu-D) image: The drone-based airborne Cu-D HSI was acquired over the agriculture crops grown in the experimental fields of the University of Agricultural Sciences, Bengaluru, India, on 5th April 2017. This dataset was captured by the Cubert UHD 185s sensor mounted on the quadcopter drone flying at the height of 20 m. The image has a spatial dimension of 900×900 pixels with a very high spatial resolution of 5 cm and has 137 spectral bands in the wavelength range 0.45 to 0.95 μm with a spectral resolution of 0.08 μm. The number of ground truth reference pixels for IS and IL are 14558 and 8074, respectively. The predominant land use categories in the study site are maize crop, soil, CS, and SS. So, there are a total of two information classes and four spectral classes in Cu-D HSI. Figure 5.2(b) shows the overlay of the ground truth map on the true color composition image where the shadow strength and size in Cu-D are moderate.

Harvard image: The outdoor ground-based platform Harvard is from the Harvard HSI database acquired in daylight conditions in 2011 (Chakrabarti and Zickler, 2011). The high-resolution Harvard dataset captured a front view main scene of one of the residential houses at Harvard University, where a massive shadow of a tree with its branch is cast on the house front view. This image consists of 1392×1040 spatial pixels with a very high spatial resolution of 2.5 cm and has 31 spectral bands in the wavelength range 0.42 to 0.72 μm with a spectral resolution of 0.01 μm. The number of ground truth reference pixels for IS and IL in Harvard image are 20617 and 9212, respectively. Figure 5.2(c) displays the overlay of the ground truth map on the RGB image. The total number of information classes and spectral classes in Harvard MSI are three and six, respectively. As shown in Figure 5.2(c), the Harvard image contains three dominant information classes such as bricks, stone, a door, and, besides, a large shadow region cast by a tree on the house.

Cubert terrestrial (Cu-T) image: The terrestrial-based Cu-T HSI was acquired over the agriculture crops grown in the experimental fields of the University of Agricultural Sciences, Bengaluru, India, on 5th April 2017. This dataset was captured by the Cubert UHD 185s sensor mounted on the ground-based tripod capturing the top view of the crop. The Cu-T image has 1000×1000 spatial dimensions with an ultra-high spatial resolution of 2 mm, and a total of 139 spectral bands were collected in the spectral range 0.45 to 0.95 μm with a spectral resolution of 0.08 μm. The number of ground truth reference pixels for IS and IL in Cu-T image are 20059 and 13473, respectively. The predominant LULC categories in the study site are cabbage crop, soil, CS, and SS. This image consists of two information classes and four spectral classes. Figure 5.2(d) shows the overlay of the ground truth map on the true color composition image where the shadow strength and size in Cu-D are moderate.

Specim image: The indoor Specim HSI was collected, on 14th March 2018, in an indoor environment scene by a portable snapshot Specim-IQ camera mounted on a tripod. This terrestrial image consists of 512×512 pixels with a very high spatial resolution of 1 cm and has 204 spectral bands in the spectral range of 0.4 to 1 μm with a spectral resolution of 0.03 μm. The number of ground truth reference pixels for IS and IL in Specim image are 14551 and 1050, respectively. The dataset has predominantly four classes: polyvinyl chloride-based floor, white reference plate, leaves, wall, and large shadows that are spatially connected with objects. There are four information classes and eight spectral classes in this HSI. As shown in Figure 5.2(e), the Specim dataset describes the best indoor test case scenario for shadows with high shadow strength, size, and different shades caused by multiple light sources.

Gualtar step (GS) image: The outdoor GS HSI captured the natural scene of residential brick stairs in the Gualtar campus, University of Minho, Portugal, on 21st May 2003 (Nascimento, Amano and Foster, 2016). The image has 1024×1344 spatial pixels with a very-high nominal spatial resolution of 1 cm and has 33 spectral bands in the wavelength range 0.4 to 0.72 μm with a spectral resolution of 0.01 μm. The number of ground truth reference pixels for IS and IL in the GS image are 21849 and 25374, respectively. In this dataset, there are three information classes and six spectral classes. This study site consists of three predominant LULC categories or information classes: bricks, grass, and pillars (see Figure 5.2(f)). In addition, three IS regions of each information class with high shadow strength and considerable large shadow region size.

5.4 Methodology

In the following, we describe the two new sets of shadow and illumination invariant image classification algorithms for MSIs and HSIs. The first set consists of two proposed versions of SMMs, namely SID and SCM. An adaptive threshold selection procedure is presented for both SID and SCM to estimate a rejection threshold for invariant predictions. The second set consists of a two-stage interconnected system, called SI^{2}CS, developed by cascading OCC and RSM. A detailed explanation of each proposed method is described as follows.

5.4.1 Proposed SMMs

This subsection briefly describes the two proposed variants of SMMs for shadow and illumination invariant image classification methods. The two considered SMMs are SCM as a deterministic measure and SID as a stochastic measure. A detailed description of the SMMs is presented in chapter 2. Below we provide a detailed description of the proposed version of SID and SCM.

5.4.1.1 Proposed SID

SID, is introduced by (Chein-I Chang, 1999), is a supervised classification method based on probabilistic measures adopted from information theory. It is used for measuring the discrepancy of probabilistic behaviors between test pixel spectral $(\mathbf{z})$ and target reference spectra $(\mathbf{x})$. In SID, each input is viewed as a random variable, and probabilistic behaviors are calculated to estimate spectral variability. Unlike classical SMMs, the proposed SID uses a set of reference pixel vectors $\mathbf{x}^{\dagger} = \{\mathbf{x}_i\}_{i=1}^{n}$ from training set S. For each test pixel vector $\mathbf{z}$, we calculate a set of SID scores between $\mathbf{z}$ and every reference pixel vector from the set $\mathbf{x}^{\dagger}$. For example, the i^{th} SID score, say f_{SID}^{i}, is computed between $\mathbf{z}$ and i^{th} reference pixel vector $\mathbf{x}_i$ from $\mathbf{x}^{\dagger}$. As a final SID score, we add up the n SID scores estimated using f_{SID}. If the final SID score, say F_{SID}, for $\mathbf{z}$ is greater than zero then we found a match otherwise no-match. Under these assumptions, we can now define the proposed version of the SID (F_{SID}) for a given $\mathbf{z}$ using the following expression:

$$F_{SID}(\mathbf{z}; \mathbf{x}^{\dagger}, \delta_{SID}) = I\left(\left\{\sum_{i=1}^{n} f_{SID}^{i}(\mathbf{z}; \mathbf{x}_i, \psi_{SID}, \delta_{SID})\right\} > 0\right), \tag{5.1}$$

$$f_{SID}^{i}(\mathbf{z}; \mathbf{x}_i, \psi_{SID}, \delta_{SID}) = I\big(\delta_{SID} \leq \psi_{SID}(\mathbf{z}, \mathbf{x}_i)\big), \tag{5.2}$$

Where ψ_{SID} denotes the SID between two spectra (refer to Section 2.3.2.1) and δ_{SID} is the threshold parameter of SID. The values of ψ_{SID} can range from 0 to ∞. A value of $\psi_{SID} = 0$ shows a perfect match or correlation with zero divergence between two spectra. The lower the ψ_{SID} value, the better the level of match between the $\mathbf{x}^{\dagger}$ and $\mathbf{z}$. $I(\cdot)$ is the unit step function defined by the equation of the following form:

$$I(A) \text{ is assigned to} \begin{cases} 1, & if \;\; A \; is \; true \\ 0, & otherwise, \end{cases} \tag{5.3}$$

A key step in our version of the SID method is the selection of the optimal $\delta_{SID} \subseteq \psi_{SID}$ value using S. The best parameter δ_{SID} is determined by 5-fold CV with parameter tuning using a linear grid search technique with a grid space of 100 evenly spaced values over the interval $[\delta_{min}, \delta_{max}]$.

Algorithm 1. Pseudocode for threshold selection of the SMM

```
1. Calculate SMM matrix using $S = \{\mathbf{x}_i, y_i\}_{i=1}^{n}$
   for $i = 1 \ldots n$ do
       for $j = 1 \ldots n$ do
           $h'_{i,j} = \psi_{SMM}(\mathbf{x}_i, \mathbf{x}_j)$                        ⊳ $h'$ = SMM matrix
       end
   end
2. Compute $\delta_{min}$ or $\delta_{max}$ using $h'$ to create 100 evenly spaced values over the interval $\boldsymbol{\delta} \in [\delta_{min}, \delta_{max}]$ for
   defining an optimal threshold $\delta_{SMM}$.
3. The threshold tuning procedure in SMM is as follows:
   partition $S$ into 5 equally sized parts $S_1, S_2, \ldots, S_5$
   for $\Delta = 1 \ldots 100$ do
       for $k = 1 \ldots 5$ do                                                     ⊳ 5-fold CV
           partition $S_k$ into $S^{train} = \{(\mathbf{x}'_l, y_l)\}_{l=1}^{n1}$ and $S^{test} = \{(\mathbf{x}''_l, y_l)\}_{l=1}^{n2}$
           for $j = 1 \ldots n2$ do
               $H_j = F_{SMM}(\mathbf{x}''_j; \mathbf{x}^{\dagger}, \boldsymbol{\delta}(\Delta))$                  ⊳ $\mathbf{x}^{\dagger} = \{\mathbf{x}''_l\}_{l=1}^{n1}$
           end
           $A'_k = \left(\sum_{j=1}^{n1} H_j\right)/n2$                                ⊳ $A'$ = Accuracy scores
       end
       $A'' = \left(\sum_{i=1}^{k} A'_k\right)/k$                                    ⊳ $A''$ = CVA score
       if $(A'' < 1)$ & $(A'' >= bA'')$ & (CONDITION) then
           $bA'' = A''$                                              ⊳ CONDITION for SID: $\boldsymbol{\delta}(\Delta) > 0$
           $\delta_{SMM} = \boldsymbol{\delta}(\Delta)$                                  ⊳ CONDITION for SCM: $\boldsymbol{\delta}(\Delta) < 1$
       end
       if $(A'' <= 1)$ & $(A'' >= bA'')$ & (CONDITION) then
           $b'A'' = A''$
           $\delta'_{SMM} = \boldsymbol{\delta}(\Delta)$
       end
       if $(\Delta == 100)$ & $(bA'' <= 1)$ & $(b'A''! = 0)$ then
           $bA'' = b'A''$
           $\delta_{SMM} = \delta'_{SMM}$
       end
       if $(\Delta == 100)$ & $(bA'' == 0)$ then
           $\delta_{SMM} = \delta$                                    ⊳ $\delta = 0$ for SID and $\delta = 1$ for SCM
       end
   end
```

Before the tuning procedure, we first need to define the possible interval of $\delta_{SID} \in [\delta_{min}, \delta_{max}]$. The minimum value that δ_{SID} can take is $\delta_{min} = 0$, while the maximum value needs to be properly selected to achieve a good generalization and maintain the balance between under-fitting and over-fitting of δ_{max}. The selection of δ_{max} is also critical because it will directly affect the final classification result. In this regard, we compute a SID matrix, say $h' \in \mathbb{R}^{n \times n}$, whose elements represent $\psi_{SID}(\mathbf{x}_i, \mathbf{x}_j)\ \forall\ 1 \leq i \leq n, 1 \leq j \leq n$, between all pair-wise pixel vectors of S. Next, we calculate a single threshold value T from h' using multi-level Otsu's method (Otsu, 1979) to compute $\delta_{max} = T$ by means of the equation

$$T = \underset{T}{\operatorname{argmin}} \left(\sum_{i=T1}^{T} p(i)(i - \mu_1)^2 + \sum_{i=T+1}^{T2} p(i)(i - \mu_2)^2 \right), \tag{5.4}$$

where $p(i)$ represents the probability of the ratio value i in h', $\mu_1 = \sum_{i=T1}^{T} p(i)/\, w_1$ and $\mu_1 = \sum_{i=T+1}^{Tm} p(i)/w_2$, $T1$ and $T2$ indicate the minimum and maximum values in an array h', $w_1 = \sum_{i=T1}^{T} p(i)$ and $w_1 = \sum_{i=T+1}^{Tm} p(i)$.

Finally, using a set of *if-then* conditional statements, each value out of 100 evenly spaced grid points from the interval $[\delta_{min}, \delta_{max}]$ are verified to find the optimal δ_{SID} that produces the highest CVA (i.e., bA''). The detailed steps are included in Algorithm 1, where replace SMM with SID. After performing the SID training to select the optimal δ_{SID}, we use Equation (5.1) to perform pixel-wise shadow and illumination invariant classification for any **z**.

5.4.1.2 Proposed SCM

Contrary to SID, SCM measures the similarity between two spectra using Pearson's linear correlation coefficient (van der Meero and Bakker, 1997). The training process of SCM to select optimal threshold is similar to SID. For each **z**, the $\{f_{SCM}\}_{i=1}^{n}$ returns n scores computed between $\mathbf{x}^{\dagger} = \{\mathbf{x}_i\}_{i=1}^{n}$ and $\mathbf{z}$, as a final score, we add all the n SCM scores. Hence, the expression of SCM F_{SCM} for any **z** is given by

$$F_{SCM}(\mathbf{z}; \mathbf{x}^{\dagger}, \delta_{SCM}) = I\left(\left\{\sum_{i=1}^{n} f_{SCM}^{i}(\mathbf{z}; \mathbf{x}_i, \psi_{SCM}, \delta_{SCM})\right\} > 0\right), \tag{5.5}$$

$$f_{SCM}^{i}(\mathbf{z}; \mathbf{x}_i, \psi_{SCM}, \delta_{SCM}) = I\big(\delta_{SCM} \geq \psi_{SCM}(\mathbf{z}, \mathbf{x}_i)\big), \tag{5.6}$$

Where ψ_{SCM} represent the SCM (refer to Section 2.3.2.2) and δ_{SCM} is the optimal threshold parameter of SCM. The values of the ψ_{SCM} can range from -1 (i.e., no similarity) to $+1$ (i.e., perfect similarity). δ_{SCM} can take values in the bounded interval $[\delta_{min}, \delta_{max}]$. The interval values of δ_{SCM}, i.e., δ_{min} and δ_{max}, are set depending on the threshold T. The single threshold T is calculated using multi-level Otsu's method from SCM matrix $h' = \psi_{SCM}(\mathbf{x}_i, \mathbf{x}_j)\ \forall\ 1 \leq i, j \leq n$ (see Equation (5.4)). If $T \geq 0.9$, we set $\delta_{min} = 0.9$ and $\delta_{max} = T$ to determine the δ_{SCM} because ψ_{SCM} values between $[-1, 0.9)$ indicate a strong negative and weak positive correlation. If $T < 0.9$, the results may be suboptimal for datasets with

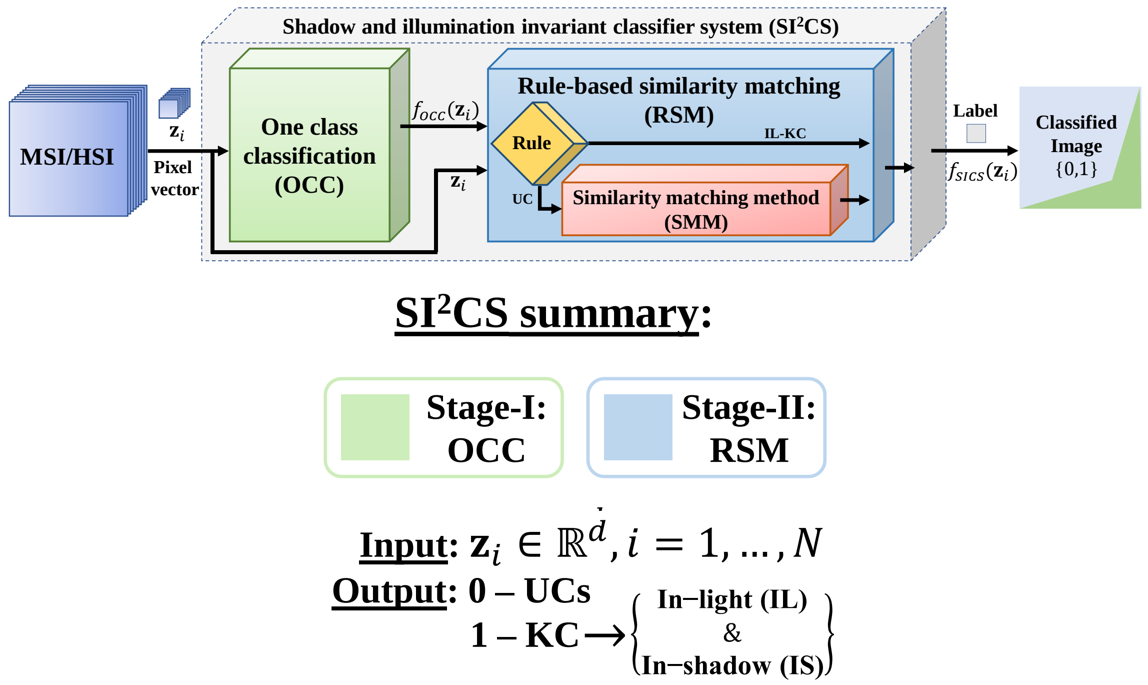

Shadow and illumination invariant classifier system (SI²CS)
MSI/HSI
$\mathbf{z}_i$
Pixel vector
One class classification (OCC)
$f_{occ}(\mathbf{z}_i)$
$\mathbf{z}_i$
Rule-based similarity matching (RSM)
Rule
IL-KC
UC
Similarity matching method (SMM)
Label
$f_{SICS}(\mathbf{z}_i)$
Classified Image {0,1}
SI²CS summary:
Stage-I: OCC
Stage-II: RSM
Input: $\mathbf{z}_i \in \mathbb{R}^d, i = 1, \ldots, N$
Output: 0 – UCs
1 – KC ⟶ In−light (IL) & In−shadow (IS)

Figure 5.3: Architecture diagram of the proposed pixel-level shadow and illumination invariant classification using SI²CS method for the i^{th} test pixel vector $\mathbf{z}_i$.

low interclass variances cases. A detailed procedure for SCM threshold selection is given in Algorithm 1, where substitute SMM with SCM. After the training phase, we now have the SCM threshold δ_{SCM} that is used to perform pixel-wise shadow and illumination invariant classification for any test instance $\mathbf{z}$ using Equation (5.5).

5.4.2 Proposed SI²CS

This subsection describes the two main functional components of the third proposed method, i.e., SI²CS – OCC and RSM (see Figure 5.3). The top-level architecture of the SI²CS follows a two-stage cascade architecture in terms of input-output relationship where the output of OCC is fed as one of the inputs to RSM and whose output is the final result of SI²CS. The decision function of SI²CS is $f_{SICS} \in \{0,1\}$, where 1 denotes the KC (including the IS-KC and high IL-KC), and 0 denotes the UCs. The SI²CS algorithm is made up of two different phases, namely, the training phase (i.e., construction of the classifier) and the classification phase (i.e., usage of the classifier). The training phase of SI^{2}CS follows a traditional supervised modeling approach. Therefore, we independently build both the OCC 'f_{OCC}' and SMM 'F_{SMM}' (using Algorithm 1) models using the available training set S. In this phase, we tune the classification model parameters of OCC and determine the optimal threshold of SMM. After completing the training phase of the SI^{2}CS, we now have the n-list of free parameter values $\boldsymbol{\theta} = \{\theta_1, \dots, \theta_n\}$ of the model which is required for performing classification tasks. In the classification phase of the SI^{2}CS algorithm, we predict the label of the unknown test pixel vector using the unified framework of the trained classification models, as shown in Figure 5.3. A detailed description of the two components of the SI^{2}CS method follows.

5.4.2.1 Stage-I: OCC

The first stage of the SI^{2}CS algorithm performs OCC to characterize an input test pixel vector. In this study, we adopt one-class SVM (OCSVM) (refer to Section 2.3.1.1) and support vector data description (SVDD) (refer to Section 2.3.1.2) as OCC techniques in the SI^{2}CS algorithm. As above mentioned, the OCC model training is done independently using S to formulate an optimal decision boundary, i.e., f_{OCC}. In the classification stage, the trained OCC model $f_{OCC} \in \{-1, +1\}$ is used to perform the binary class predictions where -1 denotes UC group and $+1$ indicates IL-KC group.

5.4.2.2 Stage-II: RSM

In the second stage of the SI^{2}CS method, RSM uses a decision rule to make shadow and illumination invariant class predictions. As shown in Figure 5.3, the RSM has two subcomponents, namely the decision rule and SMM. As mentioned earlier, the learning phase of RSM finds the optimal threshold δ_{SMM} using Algorithm 1. In the classification phase of RSM, we make use of a decision rule-set that states when to accept $\mathbf{z}$ for classification using SMM. If $f_{OCC} = +1$, then $\mathbf{z}$ is rejected for SMM classification and labeled as the positive class. Otherwise, it is classified using SMM. As shown in Figure 5.3, RSM has two inputs, namely, $\mathbf{z}$, and the label $f_{OCC}(\mathbf{z})$ from OCC (Stage-I). The output of the RSM $f_{SICS} \in \{0,1\}$ is the final classification result of the SI^{2}CS algorithm. The decision rule of SMM for training and predictions used in RSM, Stage-II of SI^{2}CS, is as follows.

Rule:

$$\begin{aligned}&Assign \quad f_{SICS}(\mathbf{z}) \rightarrow +1 \quad if \quad f_{OCC}(\mathbf{z};\boldsymbol{\theta}) = +1;\\&Otherwise \;\; Df_{SICS}(\mathbf{z}) \rightarrow F_{SMM}(\mathbf{z};\mathbf{x}^{\dagger},\delta_{SMM}).\end{aligned} \tag{5.7}$$

In the above-given decision rule, replace F_{SMM} by F_{SCM} if the SCM method is used otherwise F_{SID} if the SID method is used.

5.5 Experimental Results and Discussion

In this section, we provide a brief description of the experimental setup and metrics used to evaluate the classifiers. Further, to quantify and validate the performance of the classifiers using the obtained classification results, a detailed discussion of the results is also described in this section.

5.5.1 Experimental design of various real-world classification scenarios

An experimental setup is designed to test the classifier performance on remote sensing imagery containing shadows and varying illumination effects. The setup is made up of different types of test case classification scenarios designed from indoor-outdoor recorded imageries consisting of complex shadow regions. Our experiments split the total classes in each dataset into two groups: 1) IL classes, 2) IS classes. Each test case scenario contains three disjoint sets: KC set, UCs set, and KC shadow set. A dataset with i IL and i IS classes has i test cases or experiments, wherein in each case, we consider one out of i IL classes as KC and the remaining IL-KCs as UCs. For example, the Specim HSI dataset has four IL classes, and four IS classes have four test case scenarios. A typical use of each test case or an experiment for a given dataset with m IL-KCs can be summarized in three steps below

1. Select one out of m IL information classes as KC for a given dataset and perform the training on dataset S that is sampled only from IL-KC set, i.e., y^{Ω}.
2. Use the trained classification model for predicting the data that includes samples from all class examples, i.e., IL-KC set, UCs set, and IS-KC set.

3. Repeat step 1 and step 2 until each of m IL-KCs is selected five times. For each experiment of $5m$ trails, calculate the confusion matrix and all required accuracy metrics. As a final result, estimate the mean and SD of the performance evaluation metrics across all $5m$ trials or realizations.

5.5.2 Evaluation metrics

To systematically evaluate the classification accuracy performance of various algorithms, we used four different metrics derived from the confusion matrix. The four objective evaluation metrics are OA, AA, FPR, and FNR. The OA and AA are the ratios taken into account for correct classification, whereas the FPR and FNR are the ratios for incorrect classification count. Hence the OA and AA are expected to be higher, while FNR and FPR are expected to be relatively lower for efficient classification. The quantitative classification results calculated using the four metrics are presented in terms of two statistics to obtain representative accuracy and error estimates. First, as the mean of each metric across all m realizations, and secondly, as the empirical SD of each metric about the mean across all the m realizations. In addition to the four classification accuracy measures, one more metric, we measured the detection performance of IS-KC pixels by calculating the KC shadow detection rate (η), and it is defined as follows:

$$\eta = \frac{\text{Number of reference KC shadow pixels detected}}{\text{Total number of reference KC shadow pixels}}. \tag{5.8}$$

Therefore, the expected value and the SD of OA, AA, FPR, FNR, and η across all the experimental realizations are calculated to be the final and conclusive results.

5.5.3 Results

In this subsection, quantitative and qualitative results are presented for the six considered datasets using OCSVM, SVDD, SID, SCM, and four variants of SI^{2}CS. We ran all the classification experiments on a computer with an Intel Xeon processor 3.6 GHz CPU with 64 GB RAM and using MATLAB tool. All the datasets used in the current study are

Table 5.1: Results for all six datasets and eight classifiers containing traditional and shadow invariant methods. The mean and SD of OA (in %), AA (in %), FNR (in %), and FPR (in %) are presented for five realizations of each IL information class relative to a dataset. The best result for each metric relative to a dataset is shown in bold.

Datasets	Metrics	Traditional methods		Proposed shadow-invariant classification algorithms					
		OCSVM	SVDD	SID	SCM	SI^2CS (OCSVM+ SID)	SI^2CS (SVDD+SID)	SI^2CS (OCSVM+SCM)	SI^2CS (SVDD+SCM)
		Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
WV-3	OA	87.44 (8.8)	81.88 (12.48)	95.72 (5.73)	93.2 (8.43)	95.72 (5.73)	95.72 (5.73)	93.29 (8.41)	93.23 (8.4)
	AA	75.2 (2.92)	63.84 (3.61)	94.33 (5.98)	84.2 (10.06)	94.33 (5.98)	94.33 (5.98)	84.63 (9.88)	84.27 (10.04)
	FNR	15.34 (11.45)	20.1 (14.32)	3.32 (4.65)	8.21 (10.61)	8.1 (10.58)	3.32 (4.65)	3.31 (4.65)	8.18 (10.58)
	FPR	0.05 (0.11)	0 (0)	8.85 (20.24)	0 (0)	0.05 (0.11)	8.87 (20.23)	8.85 (20.24)	0 (0)
Cu-D	OA	67.47 (1.21)	59.06 (4)	96.44 (1.45)	87.4 (6.82)	98.02 (2.08)	96.61 (1.61)	87.55 (6.75)	87.4 (6.82)
	AA	67.07 (3.3)	58.85 (1.8)	96.55 (1)	88.82 (6.07)	98.24 (1.84)	96.74 (1.17)	88.98 (5.98)	88.82 (6.07)
	FNR	39.67 (4.15)	45.18 (5.77)	6.95 (3.38)	6.55 (6.75)	6.21 (6.91)	4.16 (4.38)	6.66 (3.65)	6.55 (6.75)
	FPR	0 (0)	0 (0)	0 (0)	15.04 (15.89)	14.99 (15.84)	0 (0)	0 (0)	15.04 (15.89)
Harvard	OA	76.87 (13.25)	71.9 (14.96)	98.32 (1.22)	98.26 (1.46)	98.44 (1.24)	98.34 (1.22)	98.35 (1.47)	98.27 (1.45)
	AA	66.47 (3.26)	58.81 (2.39)	95.89 (4.63)	95.54 (5.59)	96.02 (4.69)	95.91 (4.63)	95.69 (5.52)	95.58 (5.55)
	FNR	26.22 (15.54)	29.85 (16.14)	2.44 (1.6)	2.27 (1.45)	2.12 (1.45)	2.23 (1.51)	2.4 (1.58)	2.26 (1.44)
	FPR	0 (0)	0 (0)	0.02 (0.03)	0.43 (0.95)	0.43 (0.95)	0.02 (0.03)	0.02 (0.03)	0.43 (0.95)
Cu-T	OA	70.65 (16.89)	59.5 (14.29)	89.11 (10.67)	90.01 (11.27)	89.25 (10.59)	89.11 (10.67)	90.14 (11.38)	90.03 (11.29)
	AA	72.83 (10.69)	60.58 (5.4)	90.85 (8.57)	91.73 (9.21)	90.98 (8.51)	90.86 (8.57)	91.89 (9.35)	91.76 (9.24)
	FNR	35.82 (18.77)	44.34 (14.39)	18.08 (17.66)	16.55 (17.91)	16.35 (18.09)	17.91 (17.58)	18.07 (17.66)	16.52 (17.95)
	FPR	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)
Specim	OA	84.92 (16.86)	79.38 (21.63)	88.88 (13.51)	89.4 (12.98)	89.37 (12.92)	88.93 (13.53)	89.75 (12.96)	89.51 (13.03)
	AA	74.78 (5.81)	62.38 (5.18)	81.91 (6.01)	81.49 (7.51)	82.71 (5.84)	82.13 (6.13)	82.85 (6.93)	82.02 (7.25)
	FNR	18.38 (21.71)	22.14 (23.58)	15.11 (19.79)	14.54 (19.47)	14.21 (19.5)	14.71 (19.37)	15.05 (19.82)	14.44 (19.53)
	FPR	0 (0)	0 (0)	0.29 (0.54)	1.4 (3.21)	1.3 (3.16)	0.28 (0.52)	0.29 (0.53)	1.35 (3.18)
GS	OA	81.81 (12.78)	75.66 (18.29)	91.99 (4.12)	94.83 (1.26)	92.05 (4.11)	89.71 (6.47)	94.88 (1.2)	92.37 (6.73)
	AA	76.77 (4.15)	64.25 (1.49)	86.24 (3.28)	87.7 (6.97)	86.37 (3.17)	85.19 (5.13)	87.82 (6.86)	86.39 (8.53)
	FNR	21.8 (20.35)	28.07 (22.8)	13.43 (10.97)	7.79 (2.23)	13.24 (10.56)	7.85 (2.01)	7.75 (2.24)	13.36 (10.94)
	FPR	9 (23.51)	0 (0.01)	0 (0)	0 (0)	8.4 (22.01)	8.53 (22.39)	0 (0)	0 (0)

normalized over the interval $[0,1]$. A special program is developed using the MATLAB-based software environment to train all eight algorithms using the same set of training samples that belong to the selected IL-KC and calculate results for objective evaluation. For OCSVM and SVDD, the RBF kernel provided by the MATLAB-LIBSVM tool is considered and used (Chang and Lin, 2011). In this study, only 5% of the available ground truth reference data is used to train the algorithms, and the selection of the training set is made random. During OCSVM training, the model hyperparameters have been traced in the ranges of $v = \{0.01, 0.02, \dots, 0.1\}$ and $\gamma = 2^{\beta}$, $\beta = \{-4, -3, \dots, 4\}$. In SVDD training, the parameter C was searched over the interval $(1/n, 1]$ with an equal grid spacing of 0.1, and γ is searched over the same grid space which is used in OCSVM. For Stage-I in SI^{2}CS variants, we use a similar tuning procedure that is used for learning OCSVM and SVDD models. During the learning phase of any given algorithm, the adaptive selection scheme of free parameters, hyperparameters, and the optimal threshold is performed using five-fold cross-validation and grid searching techniques. And in the classification phase, all the experimental results are calculated on the test data consisting of all class examples of IL-KC, IS-KC, and UCs.

As mentioned earlier, although it is difficult to generate the ground truth for shadow regions in real-world scenes, we manually collected the candidate reference pixels of shadows for quantitative measurements of classification accuracies. Table 5.1 displays the results obtained for quantitative and qualitative comparison of the performance of all eight algorithms using the six datasets in terms of mean (SD) of OA, AA, FPR, and FNR. The results show that there is considerable variability among the classifiers in terms of OA, AA, FNR, and FPR, thus indicating the suitability of combination schemes of OCC and SMMs for forming the SI^{2}CS.

The variability of OA and FNR are less significant than AA and FPR between the classical methods and the proposed methods for the WV-3 MSI dataset. There is a 13.8%, 30.5%, 16.8%, and 8.9% resulted from the difference between the maximum and minimum OA, AA, FNR, and FPR in the WV-3 MSI, whereas it is about 39%, 39.4%, 41%, and 15% for the Cu-D HSI. Among the remaining four HSIs, the Harvard image has

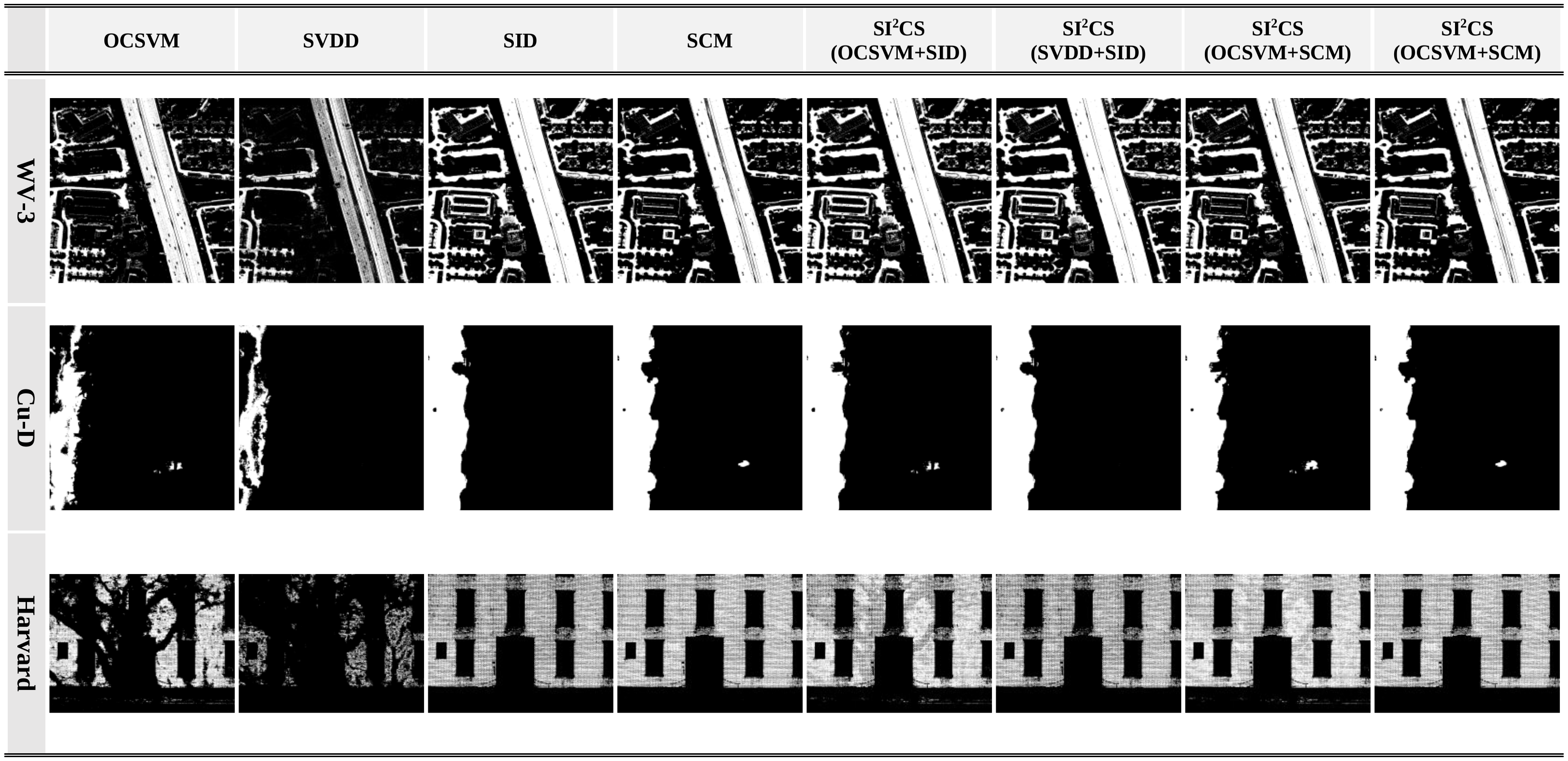


Figure: 5.4 Qualitative assessment of the eight considered classifer predictions on WV-3, CuD, and Harvard datasets. Visual comparison of classified images obtained for the specific test cases of each of the three datasets (top to bottom) using the eight classification methods (left to right). The white color represents the label prediction for the selected test cases of target KC that include IL and IS pixels and black represents the label for UCs.

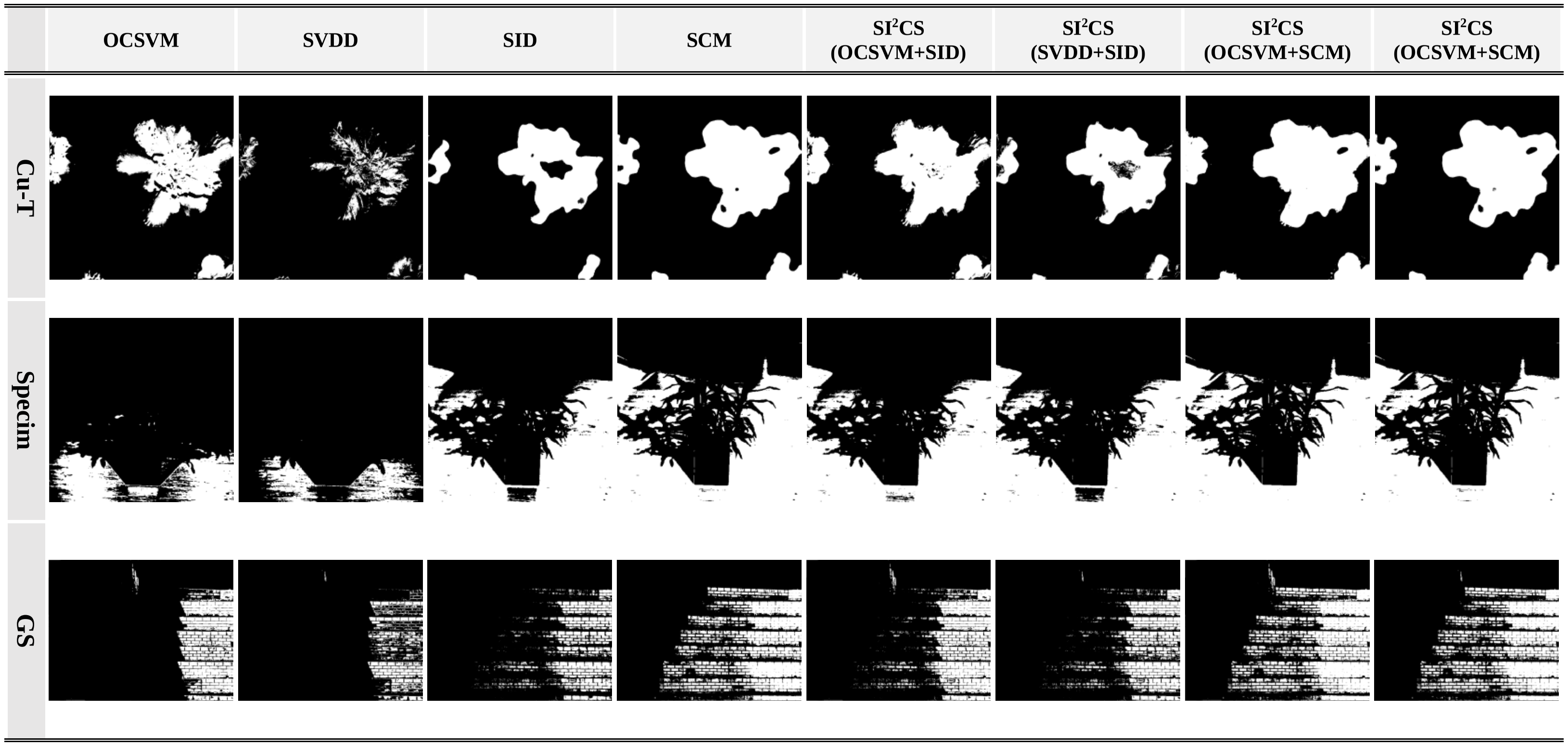


Figure: 5.5 Qualitative assessment of the eight considered classifer predictions on Cu-T, Specim, and GS datasets. Visual comparison of classified images obtained for the specific test cases of each of the three datasets (top to bottom) using the eight classification methods (left to right). The white color represents the label prediction for the selected test cases of target KC that include IL and IS pixels and black represents the label for UCs.

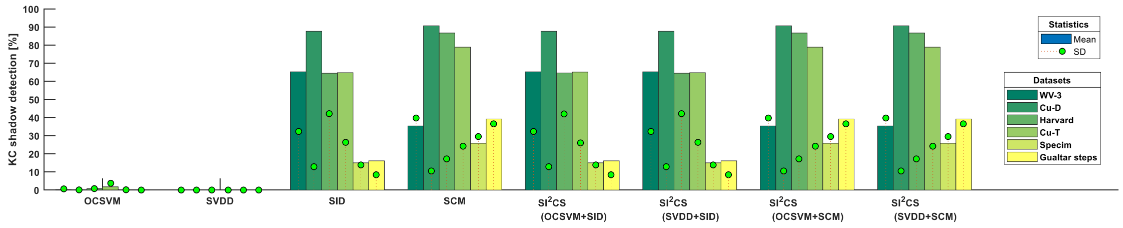


Figure 5.6: Comparison of KC shadow detection results obtained by different methods using the considered datasets. The performance measure is the mean (using bar) and SD (using green circular dot) of η.

the highest mean classification accuracies of about 98.4% OA and 96% AA. And the worst mean FNR of about 45% is for Cu-D using SVDD, and 15% mean FPR is for GS using SCM resulted among the four HSIs. However, Harvard and Cu-D datasets have the highest mean OA and AA of about 98.4% and 98.2%, respectively, among the six datasets. The best mean FNR of 2.1% resulted in Harvard, whereas the lowest mean FPR for all six datasets. SI^{2}CS has resulted in the highest classification accuracies and lowest FNR and FPR out of eight methods with respect to the classification techniques. The combination of selected classifiers in the SI^{2}CS has resulted in a 4.2% to 30.5% improvement with the OCSVM+SID combination scheme and 6.7% to 21.4% of improvement with the OCSVM+SCM against individual methods. Similarly, for the SI^{2}CS with the SVDD+SID and SVDD+SCM combination schemes, there is about a 9.2% to 37.5% and 11.1% to 30.5% increase in accuracy. It can be noted that the varying SD is due to the change in training samples that is influencing the performance. Stage-I can reliably classify IL-KC samples, while Stage-II can perform the shadow and illumination invariant classification on UC pixels labeled by Stage-I.

As shown in Figures 5.4 and 5.5, the obtained classification maps show the qualitative results of the proposed algorithms and the compared traditional algorithms for the selected test case realization of each dataset. In all the maps shown in Figures 5.4 and 5.5, OCSVM and SVDD resulted in poor thematic accuracy for shadows and OoD KC pixels. Visual inspection of Figures 5.4 and 5.5 reveal that the proposed SI^{2}CS algorithms produced smoother and accurate classification maps than OCSVM, SVDD, SID, and SCM. The

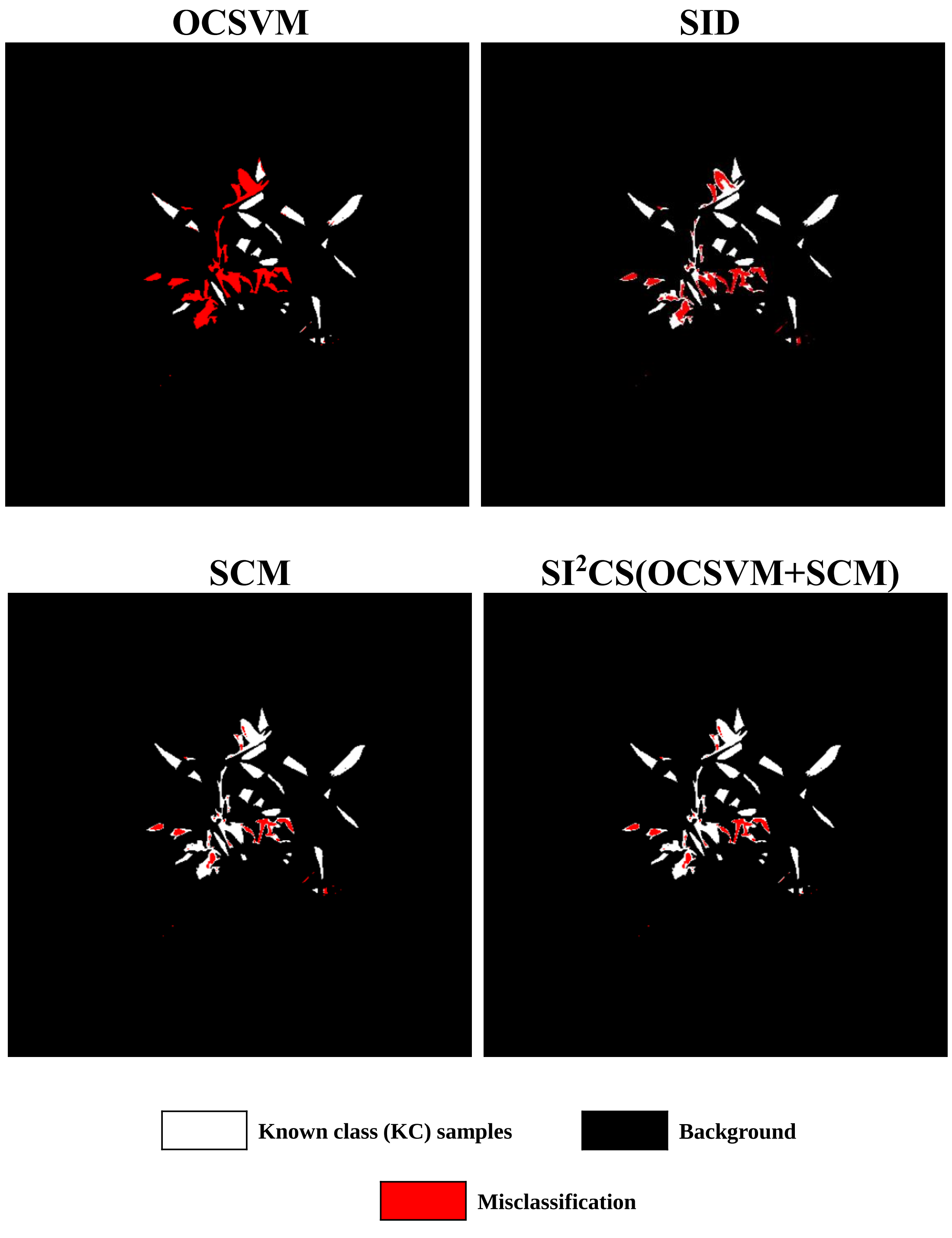


Figure 5.7: Visualization of the classification errors (in red) obtained for Specim HSI using selected classifiers for the specific realization of leaf target KC for available ground truth of IL and IS pixel vectors.

efficiency of KC shadow detection rate (η) of the eight classifiers is measured, and summary statistics of η results are graphically presented using a bar chart plot in Figure 5.6. The visual inspection of Figure 5.6 confirms that the mean and SD of η of SVDD is the first minimum, and OCSVM is the second minimum among the considered methods. The mean (SD) of η is equal for all the proposed algorithms. The best η estimates are obtained for the Cu-D image using the proposed algorithms. For illustrative purposes, Figure 5.7 displays the classification maps obtained using OCSVM, SID, SCM, and SI^{2}CS (OCSVM+SCM) classifiers for the leaf target class in Specim dataset. The classification maps in Figure 5.7 provide better insights into the errors produced by the classifiers for IS-KC pixels. In this specific realization, the training set included 130 pixels of IL leaf KC, whereas the testing set included 24647 (2592 IL-KC, 19096 UCs, and 2959 IS-KC).

The percentage results (OA, AA, FNR, FPR, η) of OCSVM, SID, SCM, and SI^{2}CS shown in Figure 5.7 are (87.8, 72.9, 0, 13.5, 0.5), (96.8, 92.8, 0, 3.96, 34.3), (97.1, 93.6, 0, 3.55, 77.7), and (97.2, 93.9, 0, 3.92, 77.7), respectively. Overall, the proposed framework SI^{2}CS with the two combination schemes of OCSVM+SCM and OCSVM+SID presented substantially higher accuracy performance for shadow and illumination invariant classification.

5.5.4 Discussion

The obtained results show that the proposed algorithms offer potential solutions to the shadow and biased lighting-related problems in classification tasks. However, several factors can hinder or limit the accuracy performance for shadow and illumination invariant classification. For instance, the ever-present within-class variability in shadow regions is one of the significant factors that challenge performance. Variability with respect to shadow reflectance of the target class is a probable source of uncertainty in classification. Although the proposed algorithms are specifically designed to account for the shadow variability, the success mainly depends on the training set used for the model learning. That is to say, the diverse collection of IL target class training samples better the performance of the algorithm in minimizing the omission errors for shadows in shadow and illumination invariant classification. If the training samples are not diverse, then the SI^{2}CS performance is better than the SMMs and the classical methods. This difference is because the SI^{2}CS combines the advantages of the OCC and SMM. Some other factors that challenged the performance of shadow invariant classification are dimensionality of data, mixed pixels, reflective surface, and interclass variance.

The task of performing a shadow invariant classification of MSIs is relatively difficult than HSIs using spectral features alone due to the following reasons. First and foremost is its lower dimensionality (e.g., $d = 8$) than HSIs, making it insufficient for accurate similarity matching between IL and IS spectra. Second, MSIs, unlike HSIs, are produced by sampling at discrete wavelengths with varying spectral resolutions for each channel or band. For these reasons, the interclass variance is relatively low for target and non-target materials in MSIs than HSIs. This performance trend between MSI and HSI in terms of metrics can be observed in Table 5.1, Figures 5.4, 5.5, 5.6, and 5.7. Therefore, caution must be exercised while collecting the training samples and performing shadow and illumination invariant classification of MSIs.

In this chapter, the current study has also performed ablation experiments simultaneously by comparing the results of SCM, SID, OCSVM, SVDD against the combination schemes of SI^{2}CS. The presented experimental results reveal that shadows in imagery are the primary contributor to omission errors. The difference in using the two-stage models versus individual models is visible from Figures 5.4 and 5.5, especially for the Specim HSI data. The role of OCC and SMM methods in the SI^{2}CS is to correctly predict the IL-KC samples and concurrently increase the mapping subspace for classifying the IS-KC samples.

5.6 Chapter Conclusions

We present a new category of shadow and illumination invariant classification algorithms using spectral curve features for high-resolution imagery. We evaluated the performance of our methods using several image classification scenarios designed from various indoor-outdoor scenes having complex shadows. Our results demonstrate that among the three proposed approaches, the SI^2CS achieved the highest performance by classifying classes beneath shadows into informational LULC types with minimum errors. The proposed algorithms show promise for interesting and practical applications because of their simplicity and ease of implementation than competing techniques. Our algorithms have the potential to offer an affordable alternate to perform shadow and illumination invariant image classification by exploiting the latent power of spectroradiometric information present in multispectral and hyperspectral imagery.

CHAPTER 6

OPEN-SET IMAGE CLASSIFICATION ALGORITHMS FOR REAL-TIME ENVIRONMENTS

Prelude: This chapter presents a set of new supervised image classification algorithms that are capable of operating with stable performance in both closed-set and open-set environments. In addition, this chapter also presents hardware architecture designs of FPGA for demonstrating and comparing the speed of computation of the proposed efficient algorithms. Several classification experiments are carried out on six different multi-sensor and multi-platform MSIs and HSIs with varying spatial and spectral resolutions containing different LULC settings. The accuracy performance of the proposed methods is compared against SVM, SC^{2}S, DCNN, and OSNN^{NNDR}. The experimental evaluation of the FPGA architecture designs of the proposed methods is performed using metrics such as timing, speed, and logic capacity, and power usage.

6.1 Introduction

Image classification is the process of uniquely assigning a single class label among a finite set of labels for each pixel in an image. It is one of the most active and vibrant areas of research in the field of remote sensing because of its potential for large-scale use in a wide variety of practical imaging applications (Scheirer, Jain and Boult, 2014; Ghamisi *et al.*, 2017). Taking advantage of spectral information in image data like MSIs/HSIs, a plethora of innovative image-driven applications are practically achievable. Such applications

---

This chapter will be published as an article and currently is in review in *Engineering Applications of Artificial Intelligence*, with the title: "A Real-time SC^{2}S-based Open-set Recognition in Remote Sensing Imagery". Authors: Dubacharla Gyaneshwar and Rama Rao Nidamanuri.

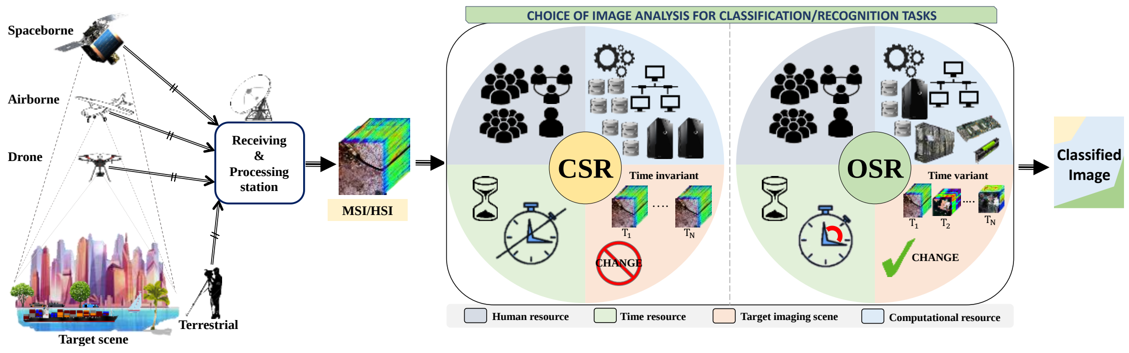


Figure 6.1: An illustration of several fundamental steps involved in digital image classification. The main emphasis is given to assessing the choice and performance of image analysis scheme to achieve classification or recognition tasks that is independent of application environments.

include, for example, monitoring and mapping Earth system dynamics, Planetary sciences, health care, homeland defense, precision agriculture, and industrial automation, etc (Eismann, 2012; Khan *et al.*, 2018). Many of these applications operate in open-set, and dynamic environments where addressing uncertainties and computational bottlenecks become crucial and immediate assessment is required (Wang *et al.*, 2016; Dubacharla and Nidamanuri, 2020; Dubacharla and Nidamanuri, 2021). Several classification algorithms have been proposed in the literature for transforming the massive amounts of collected remote sensing imagery data into the user-desired level of scientific understanding. It is well-known that existing methods work accurate and efficient for closed-set recognition (CSR) settings but are typically ineffective for open-set recognition (OSR) scenarios (Scheirer, Jain and Boult, 2014; Ghamisi *et al.*, 2017). In the most basic sense, a static or closed-set environment means that all the information about the testing classes is available at training time. Conversely, an open-set recognition (OSR) operates under the assumption that not all testing classes are known at training time, and UCs can be submitted to the trained model during testing (Júnior *et al.*, 2017). Figure 6.1 illustrates the comparison of image analysis using OSR and CSR in terms of four factors such as processing time, training information, computational resource, and supervision. As aforementioned, practical applications of image analysis and computer vision act in dynamic or open-set (usually uncertain or unknown) environments where addressing uncertainty and

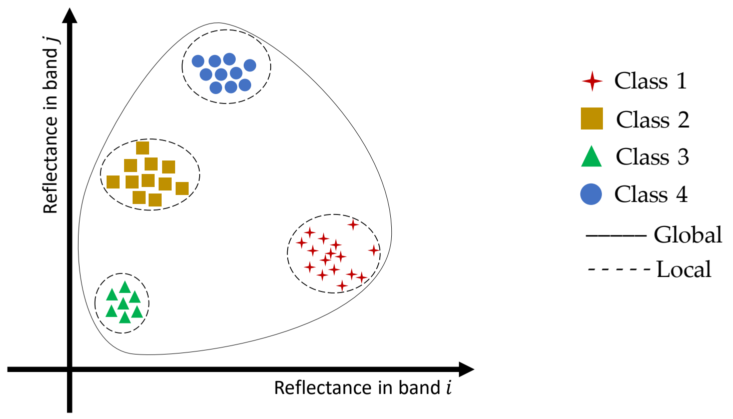


Figure 6.2: An example to show the advantage of global and local decision boundary usage by classifiers for a four-class classification problem. The scatter plot represents the two-band data distribution of four classes.

computational requirements is a critical component for success. As the scope of uncertainty in remote sensing is vast enough, it is essential to note that the focus of the current study is directed towards estimating the uncertainty (bound) due to the presence of UCs in image classification tasks.

The inherent uncertainty in classification tasks, for example, includes incomplete knowledge of labeled examples and the unknown information about the actual number of classes in the image, which are both challenging and yet unsolved problems. However, the former one has been extensively researched than the latter one. This fact has motivated us to define the objective of the present paper to address the ubiquitous challenge of minimizing (reducing) UCs uncertainty of a classifier. Typically, the examples of UCs are the test samples that are sufficiently different from ID, i.e., KCs training distribution (Scheirer, Jain and Boult, 2014; Júnior *et al.*, 2017; Dubacharla and Nidamanuri, 2021). Even though machine learning-based classification techniques have achieved high accuracy, they produce omission errors in their predictions for unseen future instances. These erroneous predictions by a classifier are referred to as substitution errors or undetected errors, which are highly related to the forced assignment problem. The forced assignment problem arises due to the incomplete knowledge of the actual number of possible information classes during training (Muzzolini, Yang and Pierson, 1998). As a result, the UC or OoD samples are forcefully labeled as one of the KCs. Thus, developing novel algorithms capable of providing stable classification performance for both OSR and CSR tasks is necessary for real-world applications (Khan *et al.*, 2018; Dubacharla and Nidamanuri, 2021).

In this work, our goal is to understand the forced assignment problem of supervised classifiers and propose rational algorithmic solutions that can alleviate this problem. In a recent paper in this journal (Dubacharla and Nidamanuri, 2020), we raised the research interest towards the open-set classification problem in the field of remote sensing, where UCs are encountered during testing. This is a critical and challenging problem to tackle. As an initial solution, we proposed an algorithm called supervised cascaded classifier system (SC^{2}S), which showed a promising future for classification tasks in an open-set scenario. This chapter refers to the SC^{2}S method as class-aware global SC^{2}S (CAG-SC^{2}S), a suitable domain name for distinction with the proposed local version of SC^{2}S. In essence, the CAG-SC^{2}S minimizes the false positives by partitioning the mapping feature space region separately for KCs and UCs. However, CAG-SC^{2}S still maps several UC samples as one of the KCs. This is because CAG-SC^{2}S considers a global level boundary for discriminating KCs and UCs (Homenda and Jastrzebska, 2015). In this regard, we introduce a new classification algorithm called the class-aware local SC^{2}S (CAL-SC^{2}S), where the class-specific decision boundary is used for mapping (see Figure 6.2). Moreover, an efficient improvement to the current model architecture of CAG-SC^{2}S is also proposed, and FPGA architectures are also designed to demonstrate the computational advantage. The proposed CAL-SC^{2}S algorithm is based on SC^{2}S and extends its general idea and scope to solve the forced assignment problem for UCs in real-world environments.

6.2 Problem Statement

An MSI/HSI $\mathbf{X} \in \mathbb{R}^{a \times b \times d}$ having d-spectral bands for every $(a \times b)$ spatial pixel can be represented as an array of N pixel vectors ($\mathbf{x}_i \in \mathbb{R}^d, i = 1, \dots, N$) known as a data matrix, say $X = [\mathbf{x}_1\ \mathbf{x}_2 \dots \mathbf{x}_N] \in \mathbb{R}^{d \times N}$. For each $\mathbf{x}$ in $\mathbf{X}$, there exists a corresponding discrete class label y that belong to the set of all possible classes or Universum classes $y^{\Psi} \in \{1, \dots, c\}$

(where c is the actual number of information classes) present in the given scene. This mapping objective can be achieved by using a classification technique with supervised approaches. To perform supervised classification of **X**, we require sample data or training data of the classes. However, collecting data of the full extent of information classes in any given scene of study can be difficult, expensive, and is not always possible (Scheirer, Jain and Boult, 2014; Júnior *et al.*, 2017; Dubacharla and Nidamanuri, 2021). In such cases, information about only a subset of classes, i.e., classes of interest, are only available.

Let $y^{\Omega} \in \{1, \dots, m\}$ be a set of m KCs $(m < c)$ and $y^{\mho} = \{y \in y^{\Psi} | y \notin y^{\Omega}\}$ be a set of $(c - m)$ UCs which are the absolute complement of y^{Ω} (i.e., $y^{\Omega} \cup y^{\mho} = \emptyset$), then we can define $y^{\Psi} = y^{\Omega} \cup y^{\mho}$. We assume that the classifier is trained with a suitable set of representative training data that belong to the KCs. Our proposed SC^{2}S method aims to reduce the false-known predictions by classifying each pixel vector to one of the KCs (y^{Ω}), otherwise as a UC ($y^{\mho}$).

6.3 Datasets Used

We used six different sources of multi-platform and multi-sensor spectral images (two MSIs each from WorldView-3, Tertracam Micro-MCA6, and four HSIs each from Hyperspec VNIR-C, AVIRIS-NG, ROSIS-3, Hyperspec VNIR-E). The six images with different spatial and spectral resolutions cover several land cover categories and sites with varying complexities. The six datasets were selected to form a complete and non-trivial experimental setup to assess the generalization of any given methods. Three-band composites and ground truth maps of the six images are shown in Figure 6.3 and Figure 6.4.

WV3 image: The spaceborne WV3 MSI is from the DigitalGlobe and distributed by the SpaceNet. The WV3 was acquired, on 15th October 2015, over the city of Mumbai, India, under clear sky conditions (SpaceNet on Amazon Web Services (AWS)., 2021). This data is pan-sharpened satellite imagery comprising of 1300×1300 spatial pixels, each having a very high-spatial resolution of 30 cm, and eight spectral bands were collected in

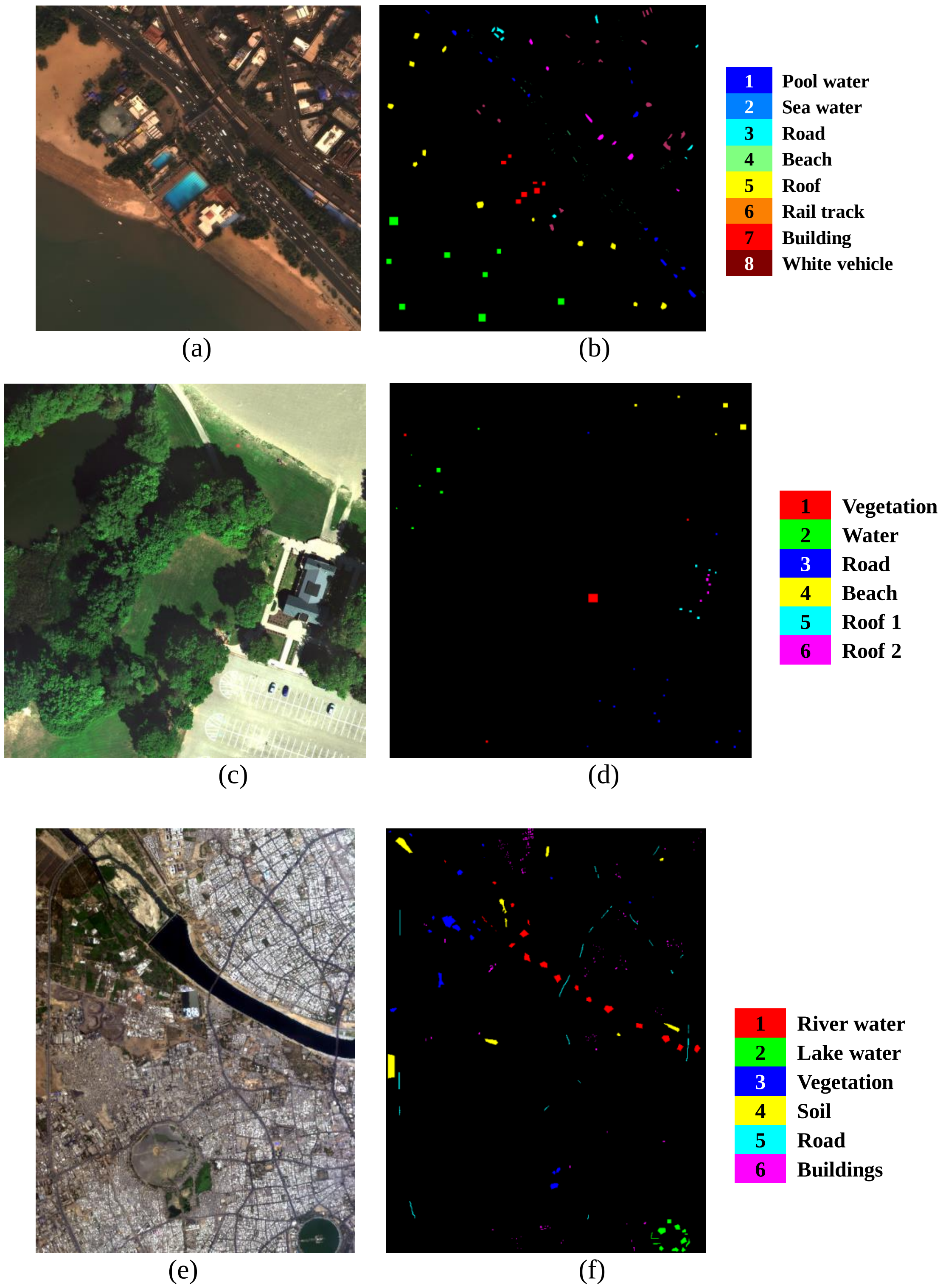


Figure 6.3: (a) RGB image and (b) reference map of WV3 MSI data. (c) RGB image and (d) reference map of R18 MSI data. (e) RGB image and (f) reference map of ANG HSI data.

the spectral range 0.4 to 1.04 μm with a varying spectral resolution of 0.04 to 0.68 μm. There are 21197 ground-truth reference samples available for eight different information classes. As shown in Figures 6.3(a) and 6.3(b), the eight dominant LULC types in this study site include pool water, sea water, road, beach sand, roof, railway tracks, buildings, and white vehicles.

R18 image: The R18 is a very-high resolution MSI dataset provided by the Rochester Institute of Technology (Kemker, Salvaggio and Kanan, 2018). The airborne image was captured on 29th August 2016 with the Tetracam Micro-MCA6 multispectral imaging sensor mounted on-board on a DJI-S1000 octocopter and flown over the Hamlin Beach State Park, New York, USA. The image has a spatial dimension of 3361×3241 pixels with a very high spatial resolution of 4.7 cm. It has six spectral bands in the wavelength range of 0.4 to 0.9 μm with a varying spectral resolution of 0.04 to 0.16 μm. This six-band dataset has 24788 number of ground truth reference pixels for six information classes. The LULC classes in this study site are vegetation, water, road, beach, roof1, and roof2. Figures 6.3(c) and 6.3(d) show the true color composition image and ground truth map, respectively.

ANG image: The ANG HSI dataset was acquired over the urban area of the Ahmedabad region, India, in January 2016. The ANG image has a spatial dimension of 750×565 pixels with a fine spatial resolution of 4 m and has 351 spectral bands in the wavelength range of 0.38 to 2.51 μm with a spectral resolution of 0.05 μm. There are 5263 ground-truth reference pixel vectors available for six information classes. The different land use categories in the study site are asphalt, building, soil, river water, lake water, and vegetation. Figures 6.3(e) and 6.3(f) show the true color composition image and ground truth reference map, respectively, for the ANG dataset.

CHK image: The airborne CHK HSI dataset is provided by Space Application Laboratory, Department of Advanced Interdisciplinary Studies, the University of Tokyo (Yokoya and Iwasaki, 2016). The dataset was acquired using a Headwall Hyperspec-VNIR-C imaging sensor mounted on aircraft and flown over agricultural and urban areas

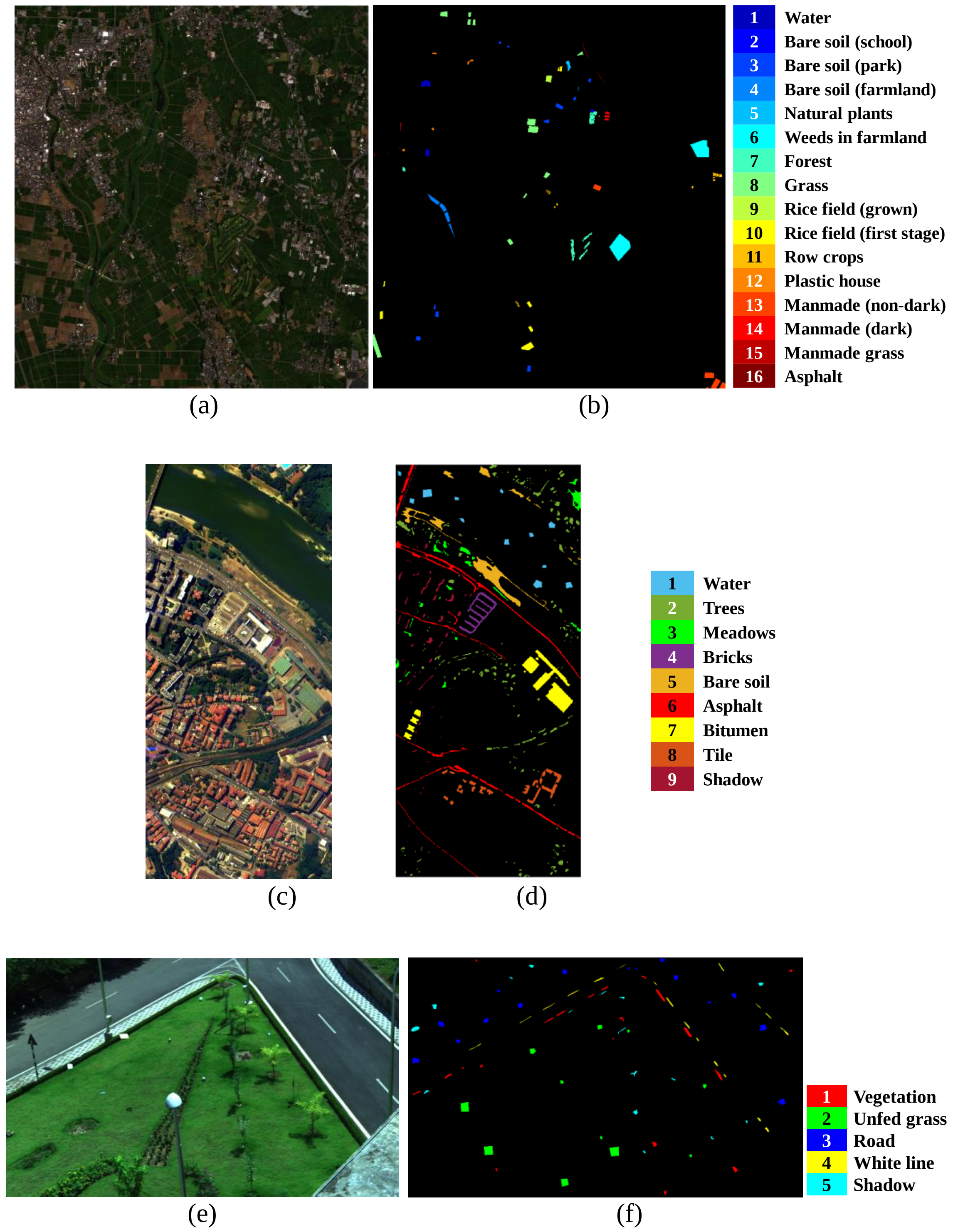


Figure 6.4: (a) RGB image and (b) reference map of CHK HSI data. (c) RGB image and (d) reference map of PC HSI data. (e) RGB image and (f) reference map of TI HSI data.

in Chikusei, Ibaraki, Japan, on July 29, 2014. The scene consists of 2318×2136 spatial pixels with a high spatial resolution of 2.5 m and has 128 spectral bands in the spectral range from 0.363 to 1.018 μm with a spectral resolution of 0.005 μm. There are 73941 number of ground truth reference pixels available for 16 information classes. The LULC categories in the study site are impervious surfaces, soils, and vegetation. Figures 6.4(a) and 6.4(b) shows the true color composition image and ground truth map, respectively.

Pavia center (PC) image: The airborne PC HSI is provided by Prof. Paolo Gamba from the Telecommunications and Remote sensing laboratory (TRSL), University of Pavia, Italy. The ROSIS dataset was acquired during a flight campaign by ROSIS-03 sensor, in July 2001, over the University of Pavia, Italy, under cloud-free sky conditions. This data comprising of 1096×492 spatial pixels, each having a high-spatial resolution of 1.3 m and 102 spectral bands (after removing the water absorption and bad bands), was collected in the spectral range of 0.43 to 0.86 μm with a nominal spectral resolution of 0.04 μm. There are 40429 ground truth samples available for nine information classes. Figures 6.4(c) and 6.4(d) show the true color image and ground truth map, respectively, for the PC image.

Terrestrial-IIST (TI) image: The outdoor TI HSI captured the natural terrestrial scene of one of the pathways in the IIST campus, Kerala, India, on 27th February 2020. The TI dataset was acquired using a Headwall VNIR-E sensor mounted on a tripod stand. It has 941×1551 spatial pixels with a very-high nominal spatial resolution of 1 cm and has 285 spectral bands in the wavelength range 0.4 to 0.72 μm with a spectral resolution of 0.01 μm. The number of available ground truth reference pixels is 17113 for five classes. This study site consists of five predominant LULC categories: road, white lane, vegetation, unfed grass, and shadow. Figures 6.4(e) and 6.4(f) show the true color composition and ground truth map, respectively, for the TI image.

6.4 Methodology

This section provides a brief description of the existing CAG-SC^{2}S and the proposed CAL-SC^{2}S methods used for generating different variants of CA-SC^{2}S algorithms. Further,

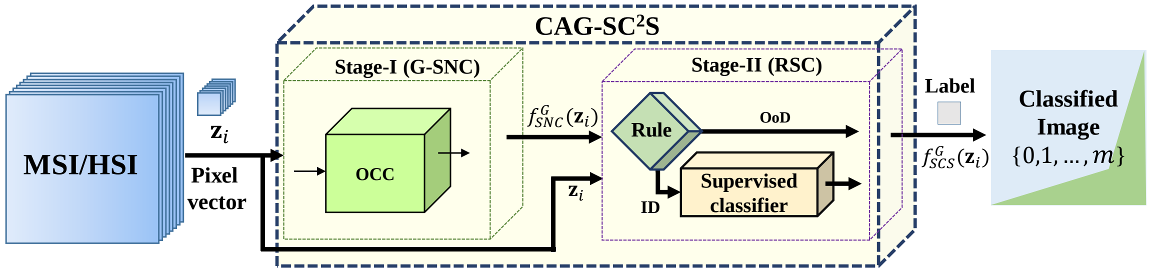


Figure 6.5: An illustration of the detailed architecture diagram of the pixel-wise CAG-S^{2}CS method for the i^{th} test pixel vector $\mathbf{z}_i$.

different classification model architectures of CA-SC^{2}S and their FPGA-based hardware design modules are also described in this section.

6.4.1 CAG-SC^{2}S

The CAG-SC^{2}S was introduced by Gyaneshwar *et al.* as a supervised classification technique for OSR tasks that can make class predictions even in the presence of UCs. The idea behind the CAG-SC^{2}S is to describe a discriminant function that can merge the objective of novelty detection and classification for efficient OSR. To ensure the efficient performance of the classification model, we combine the classification techniques from different modalities. The schematic diagram in Figure 6.5 shows that the CAG-SC^{2}S model architecture is a two-stage interconnected cascade system. The first stage of CAG-SC^{2}S is the global supervised novelty classifier (G-SNC) designed to achieve novelty detection, whereas the second stage is the rule-based supervised classifier (RSC) in which the supervised mapping is performed based on a rule-set. A brief explanation of the two stages in the CAG-SC^{2}S algorithm is described as follows.

6.4.1.1 Stage-I: G-SNC

The G-SNC is the first stage of the two-stage system (see Figure 6.5), which uses a supervised novelty classifier as the base classification technique to formulate a binary classification problem. The G-SNC is flexible to use any desired choice of novelty detection technique (e.g., OCC) for grouping ID (i.e., KCs) and OoD (i.e., UCs) samples separately in the two disjoint sets. In this study, we use OCSVM as G-SNC, which is one of the popular and widely used novelty detection techniques (Schölkopf *et al.*, 2000; Jun and Ghosh, 2013). This popularity is because of its ability to provide sparse solutions, robustness against noise, and flexibility to learn a mapping function using samples of a single class training data S to characterize non-target class (i.e., UC) and target class (i.e., KC). In the training stage of OCSVM, the origin is considered as the only available UC and then formulates an optimum hyperplane with maximum margin from the origin. The OCSVM as G-SNC, instead of using a single target class to find a single hyperplane, tries to find a single global hyperplane by considering all the available information classes as a single target class, i.e., as KC.

Mathematically, we can represent the training data for G-SNC as $S = \{(\mathbf{x}_i, {y_i}^{\Omega})\}_{i=1}^{n}$ where ${y_i}^{\Omega} = \{1, \dots, m\} \in \{+1\}$ to be a positive class and origin as a negative class. During OCSVM training, the free parameters $\boldsymbol{\theta}_{\mathrm{f}} = \{\nu, \gamma\}$ are given as input to the learning model to formulate an optimal hyperplane f_{os} which is defined by the model parameters $\boldsymbol{\theta}_{\mathrm{m}} = \{\mathbf{x}^{\dagger}, \boldsymbol{\alpha}, \rho\}$. The two free parameters are $\nu \in (0,1]$ representing the upper bound on the fraction of outliers in S and γ defining the width of the Gaussian curve of radial basis function (RBF) kernel. The three model parameters are $\mathbf{x}^{\dagger}$ representing the K number of support vectors, $\boldsymbol{\alpha}$ denotes the K size vector of Lagrange multipliers, and ρ is the bias. In the classification phase of G-SNC based on OCSVM, the trained f_{SNC}^{G} with $\boldsymbol{\theta}_{\mathrm{m}}$ is used to produce the labels $\{-1, +1\}$ where $+1$ denotes the KC group and -1 means the UC group. The expression of RBF kernel based f_{SNC}^{G} for any given test pixel vector $\mathbf{z}$ is given as follows

$$f_{SNC}^{G}(\mathbf{z}; \mathbf{x}^{\dagger}, \boldsymbol{\alpha}, \rho) = sign\left(\left(\sum_{i=1}^{\mathrm{K}} \alpha_i \exp\left(-\gamma \left\|\mathbf{x}_i^{\dagger} - \mathbf{z}\right\|^2\right)\right) - \rho\right). \tag{6.1}$$

6.4.1.2 Stage-II RSC

The RSC is the second stage in the two-stage CAG-SC^{2}S method, which employs a decision rule to perform class predictions. The RSC has two main components, namely

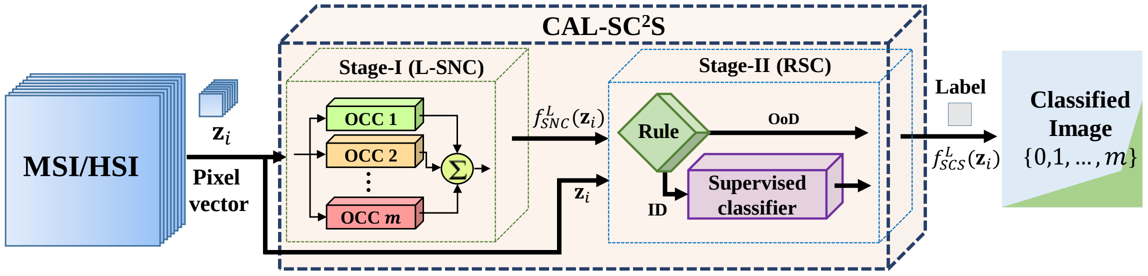


Figure 6.6: Detailed architecture diagram of the pixel-wise CAL-S^{2}CS method for the i^{th} test pixel vector $\mathbf{z}_i$.

decision rule and supervised classifier. As shown in Figure 6.5, RSC has two inputs (i.e., $\mathbf{z}$ and the label f_{OS} from G-SNC) and one output f^G_{SNC}. The output $f^G_{SNC} \in \{0,1,\dots,m\}$ from RSC is the final class prediction result of the CAG-SC^{2}S algorithm. For any $\mathbf{z}$, an RSC uses an optimum decision rule-set of the following form.

$$\begin{aligned}
&\underline{Rule}: \\
&Assign \quad f^G_{SCS}(\mathbf{z}) \rightarrow f_S(\mathbf{z};\boldsymbol{\theta}) \quad if\ f^G_{SNC}(\mathbf{z};\mathbf{x}^\dagger,\boldsymbol{\alpha},\rho) = +1 \qquad (6.2) \\
&Otherwise \quad f^G_{SCS}(\mathbf{z}) \rightarrow 0.
\end{aligned}$$

where the label “0” represent outliers or anomaly classes present in test data and $f_S(\mathbf{z};\boldsymbol{\theta}) \in y^\Omega$ is the discriminant function of a supervised classifier with a set of n-input parameters $\boldsymbol{\theta} = \{\theta_1,\dots,\theta_n\}$. Contrary to G-SNC, RSC follows a traditional training procedure to construct the decision function of a multi-class supervised classifier f_S using the set of m KCs in the training set $S = \{(\mathbf{x}_i, {y_i}^\Omega)\}_{i=1}^n$ where ${y_i}^\Omega = \{C_1,\dots,C_m\}$. In this chapter, we use a multiclass support vector machine (MSVM) (Cortes and Vapnik, 1995; Ghamisi *et al.*, 2017) and a deep convolutional neural network (DCNN) (Hu *et al.*, 2015) as the supervised classifiers in Stage-II.

6.4.2 Proposed CAL-SC^{2}S

The proposed CAL-SC^{2}S algorithm follows the technical line of CAG-SC^{2}S to explore and enhance the classification performance in any given application environment. As shown in Figure 6.6, CAL-SC^{2}S is also a two-stage interconnected system similar to the CAG-SC^{2}S.

The two cascaded decision stages in CAL-SC^{2}S are local SNC (L-SNC) as the first stage and RSC as the second stage. The main difference between the CAG-SC^{2}S and CAL-SC^{2}S algorithms is the usage of novelty detection schemes in Stage-I. The G-SNC in CAG-SC^{2}S performs refinement from the global view, whereas the L-SNC in CAL-SC^{2}S performs from the local view (Homenda and Jastrzebska, 2015; Dubacharla and Nidamanuri, 2021) (see Figure 6.2). A detailed description of the L-SNC and RSC to implement the cascade classification follows.

6.4.2.1 Stage-I: L-SNC

Unlike CAG-SC^{2}S, the first stage of CAL-SC^{2}S uses L-SNC that performs class-specific model learning to build multiple local boundaries instead of building a single global boundary for mapping ID and OoD samples (see Figures 6.2 and 6.6). For a m-KC classification problem, the L-SNC based on OCC is trained on a dataset from m KCs by independently computing m discriminant functions where each discriminant function is built using one KC subset of the m-KC training data. During the training phase of L-SNC, we initially partition the training set S having m-KCs into m training data subsets S_m and then construct j^{th} decision function using a subset S_j where $j = 1, \dots, m$. In the classification phase, m OCC classifiers' outputs m individual decision values for a given test sample. Then it uses a combination strategy to merge these m intermediate decision labels to produce a final label. Let $f_{OS}^{j} \in \{-1, +1\}$ be the j^{th} decision function of OCSVM with RBF kernel and $f_{SNC}^{L} \in \{-1, +1\}$ be the decision function of L-SNC. For a given $\mathbf{z}$, the mathematical expressions of f_{OS}^{j} and f_{SNC}^{L} are given as follows

$$f_{OS}^{j}\left(\mathbf{z}; \boldsymbol{\theta}_{\mathrm{m}}^{j}\right) = sign\left(\left\{\sum_{i=1}^{\mathrm{K}} \alpha_i^j \exp\left(-\gamma^j \left\|\mathbf{x}_i^{j\dagger} - \mathbf{z}\right\|^2\right)\right\} - \rho^j\right), \forall\, j = 1, \dots, m \qquad (6.3)$$

$$f_{SNC}^{L}(\mathbf{z}; \boldsymbol{\theta}_{\mathrm{m}}) = I\left(\left\{\sum_{j=1}^{m} f_{OS}^{j}\left(\mathbf{z}; \boldsymbol{\theta}_{\mathrm{m}}^{j}\right)\right\} + m > 0\right), \qquad (6.4)$$

$$I(A) \text{ is assigned to} \begin{cases} 1, & if\ \ A\ is\ true \\ -1, & otherwise, \end{cases} \tag{6.5}$$

where $\boldsymbol{\theta}_{\mathrm{m}}^{j}$ represent the j^{th} OCSVM model parameter set, $\boldsymbol{\theta}_{\mathrm{m}}$ represent the L-SNC model parameters, and $I(.)$ represents the function that combines the intermediate decision values.

6.4.2.2 Stage-II: RSC

The RSC is the second stage of CAL-SC^{2}S which operates the same as the RSC of CAG-SC^{2}S. The RSC of CAL-SC^{2}S employs a decision rule whose output depends on two inputs and the decision function f_S of a supervised classifier. The two inputs of RSC are $\mathbf{z}$ and $f_{SNC}^{L} \in \{-1, +1\}$. The output of RSC $f_{SCS}^{L} \in \{0,1, \ldots, m\}$ produces the final class label of CAL-SC^{2}S f_{SNC}^{L}. The decision rule-set used by RSC of CAL-SC^{2}S is of the following form.

$$\begin{array}{lll} \underline{Rule}: & & \\ Assign & f_{SCS}^{L}(\mathbf{z}) \rightarrow f_S(\mathbf{z}; \boldsymbol{\theta}) & if\ f_{SNC}^{L}(\mathbf{z}; \boldsymbol{\theta}_{\mathrm{m}}) = +1 \\ Otherwise & f_{SCS}^{L}(\mathbf{z}) \rightarrow 0. & \end{array} \tag{6.6}$$

6.4.3 Model architectures of CA-SC^{2}S

For real-world applications, accuracy and computation time are two crucial parameters that determine the efficacy of the classification algorithm. Over the years, realizing image classification algorithms capable of producing stable accuracy and performing real-time computation has been one of the vibrant areas of research across various disciplines (Dubacharla and Nidamanuri, 2020). The proposed CAG-SC^{2}S and CAL-SC^{2}S algorithms provide stable classification accuracy performance in both CSR and OSR application environments. An efficient classification model architecture of the proposed CA-SC^{2}S algorithms would address the real-time application requirements. Since it is impossible to develop an instantaneous processing system, near real-time or real-time processing systems which satisfy the timing constraints are typically a proposed option in many practical applications. In this context, we propose an efficient modification to the existing

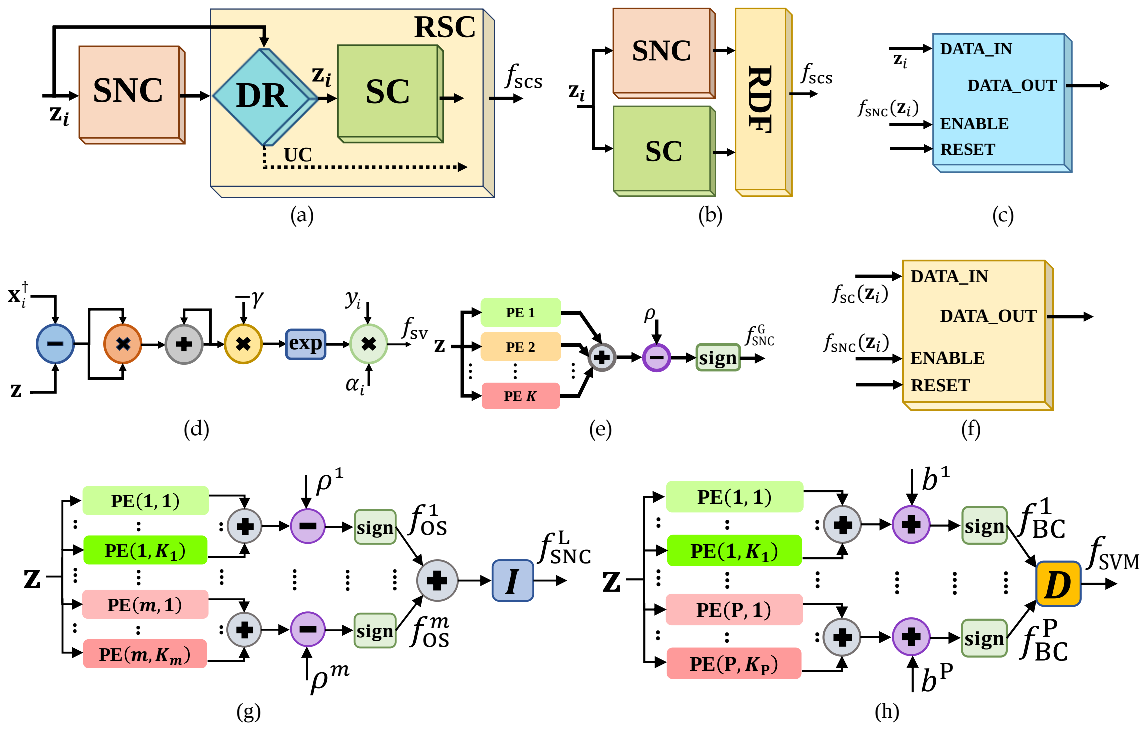


Figure 6.7: Hardware architectures of the pixel-wise classification models of the CA-SC^{2}S methods for the i^{th} test pixel vector $\mathbf{z_i}$. (a) Type-I CA-SC^{2}S model architecture, (b) Type-II CA-SC^{2}S model architecture. (c) DR module architecture. (d) The architecture of the PE module. (e) Stage-I G-SNC module architecture. (f) The architecture of the RDF module. (g) Stage-I L-SNC module architecture, and (h) SVM-based Stage-II SC module architecture.

two-stage cascade model architecture of the CA-SC^{2}S algorithm. For naming convention, we refer to the current model architecture as Type-I (see Figure 6.7(a)) and the new proposed model architecture as Type-II (see Figure 6.7(b)). As shown in Figure 6.7(a), Type-I architecture has two modules, namely SNC (Stage-I) and RSC (Stage-II). The RSC has two submodules, namely decision region (DR) and supervised classifier (SC). There are three modules in Type-II architecture, namely, SNC, SC, and rule-based decision fusion (RDF). Each of the three modules focuses on performing different objectives. The RDF uses a rule-set that is similar to the rule-set used in RSC and is shown below.

Rule:

$$\begin{aligned} &Assign \quad f^{\dagger}_{\text{SCS}}(\mathbf{z}) \rightarrow f_S(\mathbf{z};\boldsymbol{\theta}) \quad if\ f^{\dagger}_{SNC}(\mathbf{z};\boldsymbol{\theta}_{\text{m}}) = +1 \\ &Otherwise \quad f^{\dagger}_{\text{SCS}}(\mathbf{z}) \rightarrow 0. \end{aligned} \tag{6.7}$$

Unlike Type-I, which uses sequential stages, Type-II architecture uses parallel stages to perform intermediate class predictions using SNC and SC simultaneously. As a result, Type-II architecture can perform computation much more quickly than Type-I. Then, we have four different variants of the CA-SC^{2}S algorithm, namely, CAG-SC^{2}S Type-I (CAG-SC^{2}S-I), CAL-SC^{2}S Type-I (CAL-SC^{2}S-I), CAG-SC^{2}S Type-II (CAG-SC^{2}S-II), and CAL-SC^{2}S Type-II (CAG-SC^{2}S-II). In this chapter, we proposed three efficient classification models in terms of accuracy and computation time. Nevertheless, Type-I and Type-II architectures result in the same classification accuracy performance. Moreover, we also assess and demonstrate the real-time classification performance of the above-mentioned four model architectures of CA-SC^{2}S by designing the FPGA-based hardware architecture of each model. The accuracy and real-time computation performance of the four FPGA architectures of CA-SC^{2}S using SVM as SC is compared against the FPGA design of multi-class pairwise SVM. We use XSG, which is an architectural-level design tool, in a MATLAB-Simulink environment to design and evaluate the real-time performance of the five FPGA architectures (Xilinx User Guide, 2016). The training phase of the five models is performed offline since the training includes searching an optimum parameter that induces uncertain computational overheads depending on the training set used.

The model parameters obtained from offline computer training are used to design the FPGA architecture of the corresponding decision function and perform FPGA-based online classification. All the five FPGA models use SVs and processing elements (PEs) to perform class prediction for any **z**. Figure 6.7(d) shows the FPGA architecture of a PE using i^{th} SV and **z** whose mathematical expression is given in Equation (6.8). Figure 6.7(h) shows the FPGA architecture of the MSVM method with RBF kernel having P binary classifiers (BCs) and P biases. The i^{th} BC, f^{i}_{BC}, has K_i SVs and PEs. The CAG-SC^{2}S-I FPGA architecture has three main modules such as G-SNC, DR, and SC. The hardware architectures of G-SNC, DR, and SC are shown in Figures 6.7(e), 6.7(c), and 6.7(h), respectively. The CAG-SC^{2}S-II architecture also has three main modules: G-SNC, rule-based decision fusion (RDF), and SC. The hardware architectures of G-SNC, RDF, and SC are shown in Figures 6.7(e), 6.7(f), 6.7(h), respectively. The CAL-SC^{2}S-I has three main modules such as L-SNC, DR, and SC. The hardware architectures of L-SNC, DR, and SC are shown in Figures 6.7(g), 6.7(c), and 6.7(h), respectively. The CAL-SC^{2}S-II model architecture also has three main modules such as L-SNC, RDF, and SC. The hardware architectures of L-SNC, RDF, and SC are shown in Figures 6.7(g), 6.7(f), and 6.7(h), respectively. The architectures of DR and RDF are designed using registers with enable and synchronous reset inputs. To shorten the critical path in the FPGA design, we add overhead registers based on timing closure.

$$f_{\mathrm{SV}} = \alpha_i \exp\left(-\gamma\left\|\mathbf{x}_i^{\dagger} - \mathbf{z}\right\|^2\right). \tag{6.8}$$

6.5 Design of Experimental Setup of Various CSR and OSR Scenarios

In this chapter, we propose to use a comprehensive evaluation scheme to certify the efficacy of any given algorithm in terms of classification accuracy and real-time computation. The scheme uses an experimental setup that is focused on simulating two realistic classification scenarios of application environments, i.e., CSR and OSR. The proposed setup consists of five different case studies of various classification scenarios designed from multi-sensor and multi-platform MSIs and HSIs. The first four case studies are used to examine the classification accuracy performance, and the fifth case study is used to examine the real-time computation performance. Each type of case study helps us to validate the efficiency of a classification algorithm. Further, each of the first four case studies contains six test cases where each test case is prepared from each imagery. Each test case has two mutually exclusive sets, namely, KC set of size m and UC set of size $(c - m)$. The model training is performed using the samples belonging to the KC set, and then we perform class prediction on the dataset that includes the samples from the KC set and

UC set. A detailed description of the use of five case studies to validate the performance of a classifier is given as follows.

1. The first case study is to evaluate the performance of a classifier for a CSR setting where the information classes are the same during training and testing. All the available ground truth reference classes are considered KCs (i.e., $m = c$), and the UC set has no classes.
2. The second case study is used to evaluate the classifier's performance for the OSR setting. In this case study, for each one of the six test cases, we consider at least one information class to be in the KC set (i.e., $m > 1$ and $m < c$) and the remaining $(c - m)$ classes as UC set.
3. The third case study also mimics the OSR setting to validate the performance of a classification algorithm. In each of the six test cases of this case study, we preset the size or number of classes in the KC set and make a random selection of the type of information classes (i.e., m out of c) in the KC set. The remaining $(c - m)$ classes are considered in the UC set.
4. The fourth case study also contains six different test cases of the OSR setting. In this case study, the size and type of information classes in the KC set and UC set are chosen randomly.
5. In the fifth case study, we select one OSR scenario and evaluate the real-time classification performance of the proposed four FPGA-based hardware designs of the considered algorithms against the classifier without a reject option.

In each of the above-mentioned first four case studies, we repeat each of the six test-case experiments for ten random realizations or trials. As a final result of each test case, we report the mean and SD of the performance metrics across ten trails. For the fifth case study, we report the results that include classification accuracies and hardware design metrics. Hence, there are a total of 241 different experiments of classification scenarios from five various case studies.

6.6 Results and Analysis

This section presents the obtained experimental results and performs the comparative performance analysis of our two-fold objectives. The first objective aims at conducting the first four experimental case studies to obtain the classification results for assessing the accuracy performance of the classifiers. At the same time, the second objective is to conduct the fifth experimental case study and obtain the corresponding results to assess the FPGA design performance of the algorithms for a given real-time constraint. In this chapter, we assess the performance of seven different classification algorithms. The seven algorithms can be divided into two disjoint categories: algorithms without-reject option and algorithms with-reject option. In the category of without-reject option, we choose SVM and DCNN methods. In the category of with-reject option, we choose OSNN$^{\text{NNDR}}$, CAG-SC^{2}S (OCSVM+SVM), CAL-SC^{2}S (OCSVM+SVM), CAG-SC^{2}S (OCSVM+DCNN), and CAL-SC^{2}S (OCSVM+DCNN). The class prediction performance is assessed using various objective metrics, such as mean (SD) of classification accuracies [i.e., OA and AA], mean FNR, and FPR, derived from the error matrix.

The six considered datasets that are used in the experiments are preprocessed using min-max normalization over the interval [0,1]. We have performed training and classification of all the considered classifiers in the python programming language. The FPGA architecture design and implementation are done using XSG in the MATLAB-Simulink environment. The power usage results are obtained using the Xilinx Vivado tool (Xilinx User Guide, 2016). During the training phase, ten percent of the whole available reference data has been randomly selected for learning the seven classification models. In terms of the OCSVM and SVM, the RBF kernel is considered. The hyperplane parameters $\boldsymbol{\theta} = \{C, \gamma, \nu\}$ have been searched in the ranges of $C = \{0.001, \dots, 960.001\}$ with a step size of 40, $\gamma = \{0.001, 1.001, \dots, 24.001\}$ and $\nu = \{0.001, 0.012 \dots, 0.265\}$. The open-set classifier OSNN$^{\text{NNDR}}$ is modeled, and a detailed explanation of the same is provided in Chapter 2. In our DCNN architecture, we have the input layer (L1), convolution layer (L2), max-pooling layer (L3), fully connected layer (L4), and output layer (L5). The layers L2 and L3 jointly act as the feature extraction layer. Whereas layer L4 functions as the classification layer. The L1 parameter is set to be $n_1 = d$ and n_1 is the size of the input layer. In the first hidden layer, L2, there are five kernels of size 3×1 and contain 20 nodes. In the second hidden layer, L3, the stride width is set to 2, and there are no trainable parameters in this layer. The third hidden layer, L4, has 20 nodes, and the output layer L5 has m nodes. In addition, the DCCN architecture is trained over 1500 epochs with a learning rate of 0.01, validation split of 0.1, and batch size of 600.

To implement SVM and OCSVM algorithms, we used LIBSVM from the scikit-learn library (Chang and Lin, 2011), whereas DCNN is implemented using the torch library. The selection of model parameters of OCSVM, SVM, and OSNN$^{\mathrm{NNDR}}$ threshold $\tau = \{0, 0.02, \ldots, 0.98\}$ are tuned during the training stage using grid search over ten-fold cross-validation. Tables 6.1, 6.2, 6.3, and 6.4 summarizes the class prediction results of four classification case studies by seven different classifiers in terms of mean (SD) of OA and AA. In addition, Figures 6.8 and 6.9 show the mean error rates (FNR and FPR) of the four case studies and detailed classification maps of particular test case scenarios, respectively. Table 6.5 displays the FPGA-based hardware design results to demonstrate the real-time performance of the proposed algorithms and architectures. A detailed analysis of each of the seven classifiers across different case studies of CSR and OSR scenarios is provided as follows.

6.6.1 Performance evaluation of the list of classifiers using case study I

Table 6.1 shows the list of seven classifiers and their classification accuracy performance results averaged over ten trails corresponding to each test case scenario of CSR setting. In this CSR case study, we analyze whether the established classifiers and proposed classifiers are sensitive to the changes in the training set used. From the comparison of the OAs and AAs in Table 6.1, it can be inferred that, despite slight differences, all the seven classification techniques yielded better classification performance across all cases of MSIs and HSIs. These results indicate that the CA-SC^{2}S classification method pairs and OSNN$^{\mathrm{NNDR}}$ have produced nearly as good accuracies as SVM and DCNN. However, SVM and DCNN have resulted in better mean OAs and AAs with relatively low SD than the variants of CA-SC^{2}S methods. As shown in Table 6.1, the highest mean OA of 100% and AA of 100% with SD of 0% was produced by SVM for test case 2 of R18 MSI dataset. And the CAG-SC^{2}S with DCNN has the lowest mean OA of 88.41% and AA of 88.33%, with SD of at least 17% for test case 1 using the WV3 MSI dataset. Although we have performed rigorous DCNN training, the relatively low performance of CAG-SC^{2}S using DCNN as a supervised classifier is may be due to the chosen architecture or parameters. Among the four variants of CA-SC^{2}S, CAG-SC^{2}S and CAL-SC^{2}S using SVM have performed better than DCNN. Further, there is a less significant difference in the classification accuracies between CAG-SC^{2}S and CAL-SC^{2}S using SVM.

Moreover, to assess the errors produced by the seven multiclass classifiers, we have also estimated the mean percentages of FNR and FPR over the ten realizations of each test case. The estimated mean percentages of FNR and FPR of all the classifiers for each test case are shown visually in Figures 6.8(a) and 6.8(b), respectively. The lowest mean FNR and FPR of 0% is obtained by SVM for test case 2. And the highest mean FNR of at least 30.77% and FPR of at least 0.67% are obtained for CA-SC^{2}S variants using DCNN for test cases 1 and 5. The obtained results from Table 6.1, Figures 6.8(a), and 6.8(b) indicates that the performance of classifiers without reject-option (SVM and DCNN) is marginally high than the classifiers with reject-option. This is particularly due to the less sensitivity of classifiers without reject-option to the training set used for model learning than the classifiers with reject-option. These observations indicate that the suitability of the CA-SC^{2}S variants with the representative training set, especially the CAL-SC^{2}S variants, for LULC classification for the CSR environment setting.

6.6.2 Performance evaluation of the list of classifiers using case study II

Table 6.2 shows the classification accuracies obtained by the seven classification techniques for the case study II of OSR setting with more than one UC. Unlike case study I, SVM and DCNN have produced very low classification accuracy results when compared to OSNN^{NNDR} and four variants of CA-SC^{2}S. From Table 6.2, a statistically significant difference between classifiers without reject-option and with reject-option in terms of

Table 6.1: OA and AA results obtained by the list of classifiers for the six test cases with no UCs in case study I.

Datasets	Test cases y^{Ω} $(m = c)$	Metrics	Classifiers without reject option		Classifiers with reject option					
			SVM	DCNN	NNDR	OCSVM + SVM		OCSVM + DCNN		
						CAG-SC²S	CAL-SC²S	CAG-SC²S	CAL-SC²S	
			Mean (SD)	**Mean (SD)**	**Mean (SD)**	**Mean (SD)**	**Mean (SD)**	**Mean (SD)**	**Mean (SD)**	
WV3	$\{1, 2, \dots, 8\}$	**OA**	**99.81 (0.07)**	97.97 (0.28)	99.32 (0.20)	90.07 (19.77)	99.45 (0.12)	88.41 (19.53)	97.62 (0.30)	
		AA	**99.58 (0.14)**	97.00 (0.49)	98.68 (0.35)	90.68 (17.35)	98.79 (0.28)	88.33 (17.21)	96.23 (0.52)	
R18	$\{1, 2, \dots, 6\}$	**OA**	**100.00 (0.00)**	99.86 (0.03)	99.73 (0.32)	99.77 (0.11)	99.52 (0.19)	99.64 (0.09)	99.40 (0.17)	
		AA	**100.00 (0.00)**	99.78 (0.04)	99.73 (0.31)	99.81 (0.09)	99.44 (0.23)	99.59 (0.07)	99.23 (0.21)	
ANG	$\{1, 2, \dots, 6\}$	**OA**	**99.90 (0.07)**	99.87 (0.06)	99.46 (0.53)	98.78 (0.71)	97.39 (0.15)	98.75 (0.72)	97.40 (0.16)	
		AA	**99.88 (0.08)**	99.84 (0.07)	99.24 (0.84)	98.80 (0.83)	96.99 (0.49)	98.76 (0.81)	97.01 (0.50)	
CHK	$\{1, 2, \dots, 16\}$	**OA**	**99.58 (0.03)**	98.33 (0.07)	98.95 (0.03)	99.50 (0.08)	99.53 (0.09)	98.25 (0.11)	98.27 (0.13)	
		AA	**98.30 (0.02)**	91.01 (0.52)	96.16 (0.62)	98.03 (0.09)	98.16 (0.17)	90.75 (0.58)	90.89 (0.64)	
PC	$\{1, 2, \dots, 9\}$	**OA**	**97.53 (0.11)**	95.79 (0.16)	94.84 (0.14)	97.37 (0.12)	96.30 (1.62)	95.62 (0.20)	94.55 (1.57)	
		AA	**97.59 (0.10)**	95.30 (0.23)	94.66 (0.25)	97.36 (0.15)	95.42 (3.32)	95.07 (0.34)	93.13 (3.41)	
TI	$\{1, 2, \dots, 5\}$	**OA**	**99.89 (0.04)**	99.83 (0.02)	98.96 (1.34)	99.62 (0.29)	98.88 (0.32)	99.56 (0.31)	98.82 (0.31)	
		AA	**99.87 (0.05)**	99.81 (0.03)	99.17 (0.95)	99.45 (0.43)	98.48 (0.43)	99.39 (0.43)	98.43 (0.42)	

Table 6.2: OA and AA results obtained by the list of classifiers for the specific six test cases in case study II.

Datasets	Test cases y^{Ω} $(m > 1$ and $m < c)$	Metrics	Classifiers without reject option		Classifiers with reject option				
			SVM	DCNN	NNDR	OCSVM + SVM		OCSVM + DCNN	
						CAG-SC²S	CAL-SC²S	CAG-SC²S	CAL-SC²S
			Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)	Mean (SD)
WV3	$\{1, 4, 6, 8\}$	**OA**	60.81 (0.07)	60.49 (0.08)	70.55 (5.26)	69.56 (1.26)	**89.87 (2.50)**	69.25 (1.30)	89.56 (2.47)
		AA	79.59 (0.24)	78.84 (0.25)	82.81 (2.05)	83.00 (0.89)	**93.66 (1.86)**	82.28 (0.94)	92.95 (1.73)
R18	$\{1, 2, 5\}$	**OA**	22.42 (0.00)	22.42 (0.00)	83.79 (12.23)	81.33 (7.07)	**87.73 (8.46)**	81.33 (7.07)	87.73 (8.46)
		AA	66.67 (0.00)	66.67 (0.00)	93.03 (5.25)	91.75 (2.93)	**93.95 (3.57)**	91.75 (2.93)	93.95 (3.57)
ANG	$\{1, 4, 6\}$	**OA**	49.31 (5.55)	49.29 (5.57)	80.58 (16.37)	81.53 (14.82)	**93.70 (3.32)**	81.51 (14.79)	93.70 (3.32)
		AA	74.95 (0.10)	74.89 (0.19)	89.74 (8.36)	89.52 (8.13)	**94.17 (2.75)**	89.47 (8.06)	94.18 (2.75)
CHK	$\{1, 2, 5, 7, 10, 12, 15\}$	**OA**	47.06 (0.01)	46.99 (0.02)	48.12 (0.52)	48.11 (0.13)	**51.76 (2.38)**	48.04 (0.14)	51.69 (2.39)
		AA	87.40 (0.03)	87.22 (0.09)	87.53 (0.15)	87.50 (0.10)	**87.57 (1.49)**	87.31 (0.11)	87.38 (1.51)
PC	$\{1, 4, 7\}$	**OA**	28.25 (0.06)	28.21 (0.07)	39.79 (7.56)	65.60 (11.83)	**70.57 (7.73)**	65.56 (11.84)	70.54 (7.74)
		AA	73.78 (0.17)	73.71 (0.21)	76.01 (2.48)	86.45 (3.86)	**88.01 (2.52)**	86.37 (3.90)	87.94 (2.55)
TI	$\{1, 4\}$	**OA**	31.82 (0.00)	31.82 (0.00)	70.16 (15.81)	49.24 (17.31)	**98.27 (2.13)**	49.24 (17.31)	98.27 (2.13)
		AA	66.67 (0.00)	66.67 (0.00)	85.40 (7.72)	74.32 (8.29)	**98.07 (1.12)**	74.32 (8.29)	98.07 (1.12)

accuracies can be observed. This accuracy difference is of a large margin, of about 9% to 67% for OA and 1% to 32% for AA, for all the test cases except for test case 4. For test case 4, there is a small margin of difference of about 2% to 5% for OA and 0.09% to 0.17% for AA. As shown in Table 6.2, the highest mean OA of 98.27% and AA of 98.07% with an SD of 2.13% was produced by CAL-SC^{2}S using SVM and CAL-SC^{2}S using DCNN for test case 6 of TI HSI dataset. And DCNN has produced the lowest mean OA of 28.21% for test case 5 and AA of 66.67% for test cases 2 and 6. There is no significant difference between SVM and DCNN in terms of OAs and AAs. The poor classification performance by SVM and DCNN can be inferred as the existence of forced assignment problems for UCs. It can be observed from Table 6.2 that the OSNN^{NNDR} has yielded better results than SVM and DCNN but less than the four variants of CA-SC^{2}S. However, in some cases, OSNN^{NNDR} offered better classification accuracy than CAG-SC^{2}S but less than CAL-SC^{2}S.

Among the four variants of CA-SC^{2}S, CAL-SC^{2}S using SVM resulted in significantly better classification accuracy for all the test cases. In fact, a good improvement has been observed with CAL-SC^{2}S compared to CAG-SC^{2}S. For example, for the TI HSI dataset 49.24%, and 98.27% of OA are observed respectively for the CAG-SC^{2}S using SVM and CAL-SC^{2}S using SVM; and 69.56% and 89.97% AA observed for WV3 image. A similar trend in OA and AA can be observed for the remaining images or test cases. However, the validity of this observation depends upon the test cases and the training set used for learning. There is a less significant difference in the classification accuracies between CAL-SC^{2}S using DCNN and CAL-SC^{2}S using SVM in few cases.

Further, examining the better-performing classifier relative to each test case in the OSR environment can be done using FNR and FPR. In this regard, we illustrate the obtained mean percentages of FNR and FPR using a 3D bar chart plot for the classifiers relative to each test case in Figures 6.8(c) and 6.8(d). A low FNR and low FPR indicate a good class prediction with minimum errors. When compared with the FNR and FPR results, it can be observed that SVM and DCNN produced more errors (i.e., high FNR and FPR) with low SD for all the cases. These results confirm the confident errors often made by the classifiers without reject-option. This is an important observation because a classifier with an inclusion of the reject-option is particularly advantageous when dealing with unknown samples or UCs to reduce false predictions.

6.6.3 Performance evaluation of the list of classifiers using case study III

The OA and AA estimation results obtained from different sets of classifiers for all the six test case scenarios in case study III are shown in Table 6.3. This case study was explicitly designed to evaluate the dependency and sensitivity of the classification performance on the number of KCs or UCs. As can be seen from the table, we can derive some interesting inferences. It can be seen, in all experimental test cases, the difference in mean percentages of OA and AA results between the classifiers without and with reject-option is significantly large. For instance, a minimum of 2% and a maximum of 58% OA difference are observed respectively for the classifiers without and with reject-option; and 4% and 39% AA difference. As shown in Table 6.3, the highest mean OA of 98.92% with SD of 0.34% and AA of 98.73% with SD of 0.43% was produced by CAL-SC^{2}S using SVM for test case 6 of the TI HSI dataset. And DCNN has made the worst mean OA of 33.06% and AA of 33.04% for the test cases 2. Despite the observation that both SVM and DCNN present nearly similar results, it is noteworthy that the SVM is able to reach a slightly better performance. The OSNN^{NNDR} method presents better results than SVM and DCNN but less than the four variants of CA-SC^{2}S. Nevertheless, in test cases 1 and 4, OSNN^{NNDR} obtained better classification accuracy than CAG-SC^{2}S but less than CAL-SC^{2}S.

When compared with AA results obtained from SVM and DCNN for all the test cases, it can be observed that UCs are completely misclassified into one of the KCs. For example, for the first test case, there are three KCs and five UCs. Then we consider the five UCs as one category for performing the classification and accuracy evaluation. Now, there are a total of four classes (i.e., three KCs and one UC), and their AA is about 79.59% for SVM, which confirms that the remaining 20% error is due to the UC category. A similar trend can be observed for the remaining test cases and also in the four case studies. This critical observation cannot be inferred by OA results alone because of the presence of unbalanced datasets. In the case of classifiers with rejection, the OSNN^{NNDR} and the four derivative

Table 6.3: OA and AA results obtained by the classifiers for the test cases with fixed KC set size in case study III.

Datasets	**Test cases**		**Metrics**	**Classifiers without reject option**		**Classifiers with reject option**				
				SVM	**DCNN**	**OSNN^{NNDR}**	**OCSVM + SVM**		**OCSVM + DCNN**	
	$\lvert KC\rvert$	**$\lvert UC\rvert$**					**CAG-SC^{2}S**	**CAL-SC^{2}S**	**CAG-SC^{2}S**	**CAL-SC^{2}S**
	m	**$(c-m)$**		**Mean (SD)**	**Mean (SD)**	**Mean (SD)**	**Mean (SD)**	**Mean (SD)**	**Mean (SD)**	**Mean (SD)**
WV3	**3**	**5**	**OA**	30.70 (11.91)	30.64 (11.90)	71.27 (22.23)	70.65 (17.57)	**83.90 (11.81)**	70.60 (17.54)	83.86 (11.79)
			AA	72.25 (4.12)	72.07 (4.02)	86.63 (8.50)	86.37 (6.70)	**91.67 (4.88)**	86.19 (6.59)	91.52 (4.82)
R18	**2**	**4**	**OA**	33.06 (10.19)	33.04 (10.20)	78.46 (13.14)	84.18 (9.39)	**90.44 (11.06)**	84.16 (9.39)	90.43 (11.07)
			AA	66.66 (0.01)	66.60 (0.15)	89.00 (7.05)	92.32 (3.57)	**94.83 (5.88)**	92.25 (3.55)	94.79 (5.89)
ANG	**3**	**3**	**OA**	55.62 (8.38)	48.22 (17.37)	74.99 (14.86)	84.37 (11.22)	**95.20 (3.09)**	77.24 (12.85)	88.29 (12.91)
			AA	74.93 (0.09)	60.68 (24.38)	84.20 (8.99)	89.22 (6.87)	**95.71 (1.67)**	75.47 (20.92)	82.45 (23.73)
CHK	**6**	**10**	**OA**	38.33 (17.00)	38.14 (16.78)	40.27 (16.62)	47.92 (22.06)	**57.48 (19.23)**	47.73 (21.84)	57.29 (19.00)
			AA	84.77 (1.26)	80.70 (7.89)	84.15 (2.47)	86.54 (1.94)	**89.10 (1.27)**	82.48 (5.71)	85.05 (5.37)
PC	**4**	**5**	**OA**	53.23 (7.68)	52.99 (7.53)	58.06 (8.54)	65.86 (14.87)	**77.24 (9.09)**	65.62 (14.65)	76.99 (8.97)
			AA	78.82 (1.05)	78.30 (1.70)	79.54 (4.93)	84.35 (4.39)	**88.69 (4.05)**	83.83 (3.81)	88.16 (3.86)
TI	2	**3**	**OA**	40.32 (7.23)	37.72 (9.52)	79.42 (22.19)	77.82 (16.41)	**98.92 (0.34)**	75.24 (15.55)	96.35 (5.38)
			AA	66.61 (0.08)	59.97 (14.04)	88.48 (12.06)	87.39 (8.46)	**98.73 (0.43)**	80.82 (13.81)	92.17 (13.73)

Table 6.4: OA and AA results obtained by the classifiers for the random selection of KCs in case study IV.

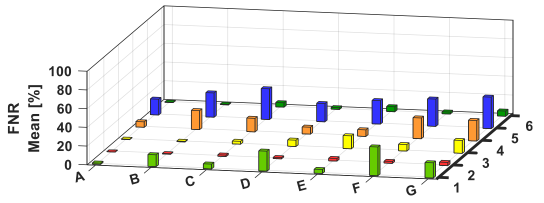
Datasets	**Test cases y^{Ω} $(m > 1$ and $m < c)$**	**Metrics**	**Classifiers without reject option**		**Classifiers with reject option**				
			SVM	**DCNN**	**OSNN^{NNDR}**	**OCSVM + SVM**		**OCSVM + DCNN**	
						CAG-SC^{2}S	**CAL-SC^{2}S**	**CAG-SC^{2}S**	**CAL-SC^{2}S**
			Mean (SD)	**Mean (SD)**	**Mean (SD)**	**Mean (SD)**	**Mean (SD)**	**Mean (SD)**	**Mean (SD)**
WV3	**Random**	**OA**	47.04 (26.66)	46.70 (26.15)	63.97 (21.08)	72.88 (17.23)	**82.07 (15.22)**	72.54 (16.84)	81.73 (14.90)
		AA	76.01 (7.71)	75.48 (7.07)	83.55 (5.80)	87.27 (5.95)	**90.75 (6.45)**	86.73 (5.59)	90.21 (6.13)
R18	**Random**	**OA**	71.04 (17.48)	70.99 (17.47)	85.19 (10.00)	88.82 (8.02)	**94.70 (6.28)**	88.77 (8.00)	94.67 (6.27)
		AA	78.66 (5.20)	78.55 (5.17)	88.65 (4.76)	90.02 (7.03)	**95.66 (3.98)**	89.91 (6.98)	95.59 (3.97)
ANG	**Random**	**OA**	43.42 (19.69)	41.31 (21.87)	92.33 (11.90)	**97.18 (3.26)**	94.04 (5.35)	95.15 (4.40)	92.02 (4.90)
		AA	70.29 (6.00)	62.91 (17.67)	93.06 (6.34)	**95.48 (4.72)**	94.03 (3.77)	88.39 (13.48)	86.98 (12.69)
CHK	**Random**	**OA**	25.86 (29.88)	23.64 (31.27)	34.30 (25.05)	56.08 (33.57)	**78.60 (8.97)**	53.86 (36.89)	76.38 (7.17)
		AA	77.98 (10.54)	46.80 (35.51)	79.94 (9.78)	87.18 (7.50)	**93.02 (3.08)**	56.00 (35.49)	61.91 (26.66)
PC	**Random**	**OA**	51.44 (24.96)	51.04 (24.72)	64.03 (14.13)	67.85 (20.86)	**75.18 (9.21)**	67.45 (20.79)	74.78 (9.36)
		AA	78.79 (7.94)	78.14 (7.67)	82.77 (5.67)	85.10 (6.88)	**87.61 (3.74)**	84.45 (6.89)	86.96 (4.25)
TI	**Random**	**OA**	63.72 (21.99)	56.28 (26.66)	97.35 (6.23)	85.44 (16.02)	**98.50 (1.51)**	78.03 (22.35)	91.15 (15.20)
		AA	76.40 (9.94)	63.76 (25.75)	98.20 (3.14)	90.28 (10.10)	**98.51 (1.12)**	77.72 (24.71)	86.05 (21.35)

techniques of CA-SC^{2}S offered better class predictions by rejecting to classify the UCs as one of the KCs. Figures 6.8(e) and 6.8(f) show the mean percentages of FNR and FPR by the classifier for the test cases. The best FNR and FPR scores are obtained for test case 6 by CAL-SC^{2}S using the SVM method. CAL-SC^{2}S using SVM presents relatively better OAs, AAs, FNR, and FPR results for all the images among the seven classifiers.

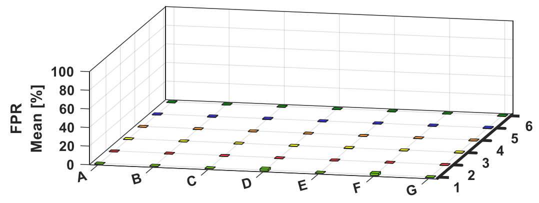
6.6.4 Performance evaluation of the list of classifiers using case study IV

From the results detailed in Table 6.4, it can be observed that the proposed CA-SC^{2}S methods exhibit superior performance in terms of OA and AA. Similar to the previous two case studies, there is a significant difference of at least a 10% change in classification accuracies between the classifiers without rejection and with rejection. The SVM and DCNN offered the lowest mean percentages of OAs and AAs among the seven classifiers across all the test cases. For example, SVM presented a minimum and maximum mean OA of 25.86% for CHK image and 63.72% for TI image, respectively. In the case of DCNN, no significant improvement is observed across the six images. The classification results obtained by the classifiers with reject-option point out substantial differences in the magnitude of OA and AA increase. The observed significant increase in the classification accuracies of the classifiers with reject-option supports the theoretical advantage of using reject-option and an extra-label. The OSNN^{NNDR} performed fairly well across the six cases and presented higher accuracies than CAG-SC^{2}S methods for test case 6. Unlike the CAG-SC^{2}S, the performance of OSNN^{NNDR} is susceptible to the training data and may result in errors in the presence of noise.

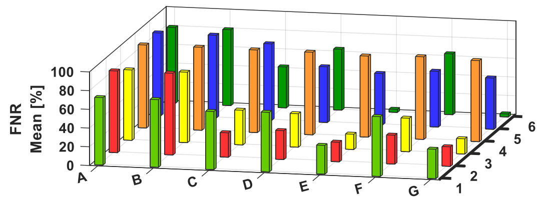
When the accuracy results of classifiers with reject-option are compared, it can be seen that CAL-SC^{2}S using the SVM method offered stability in classification performance. As shown in Table 6.4, the highest mean percentages of OA 98.50% and AA 98.51% are obtained by CAL-SC^{2}S using SVM for the TI HSI image. The lowest mean percentages among the classifiers with reject-option are obtained for the CHK image with the OSNN^{NNDR} method 34.30% of OA and with CAG-SC^{2}S using the DCNN method 56% of OA AA. Figures 6.8(g) and 6.8(h) show that the mean percentages of FNR and FPR results



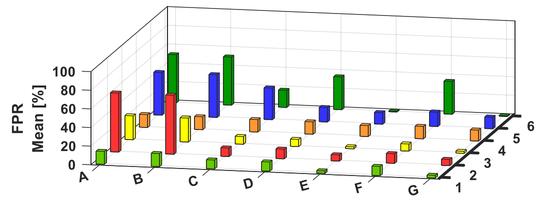
(a)



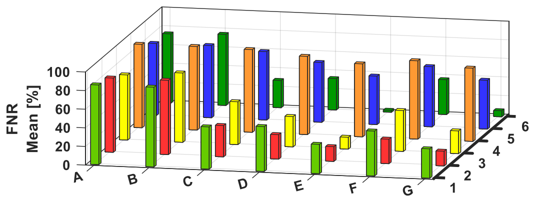
(b)



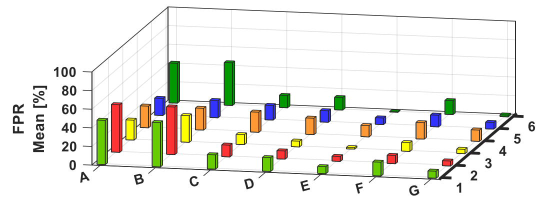
(c)



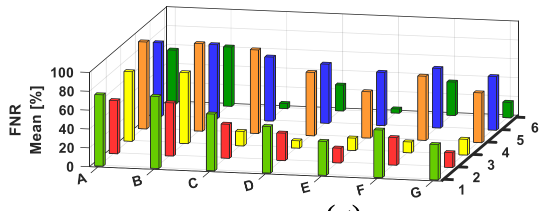
(d)



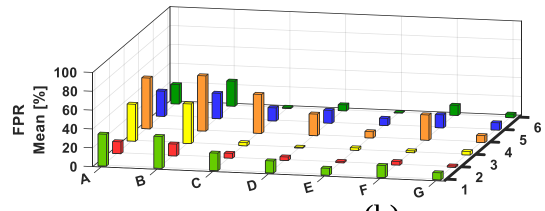
(e)



(f)



(g)



(h)

A	B	C	D	E	F	G
SVM	**DCNN**	**OSNN^{NNDR}**	**CAG-SC^{2}S (OCSVM+SVM)**	**CAL-SC^{2}S (OCSVM+SVM)**	**CAG-SC^{2}S (OCSVM+DCNN)**	**CAL-SC^{2}S (OCSVM+DCNN)**

Test cases	**1**	**2**	**3**	**4**	**5**	**6**
	🟩	🟥	🟨	🟧	🟦	🟩

Figure 6.8: 3D bar plot visualization of the obtained mean percentages of FNR [(a), (c), (e), and (g)] and FPR [(b), (d), (f), and (h)] results by seven classifiers for six different test cases in four case studies. The increasing order of case studies from top to bottom. The legend for the list of classifiers and test cases is also shown above.

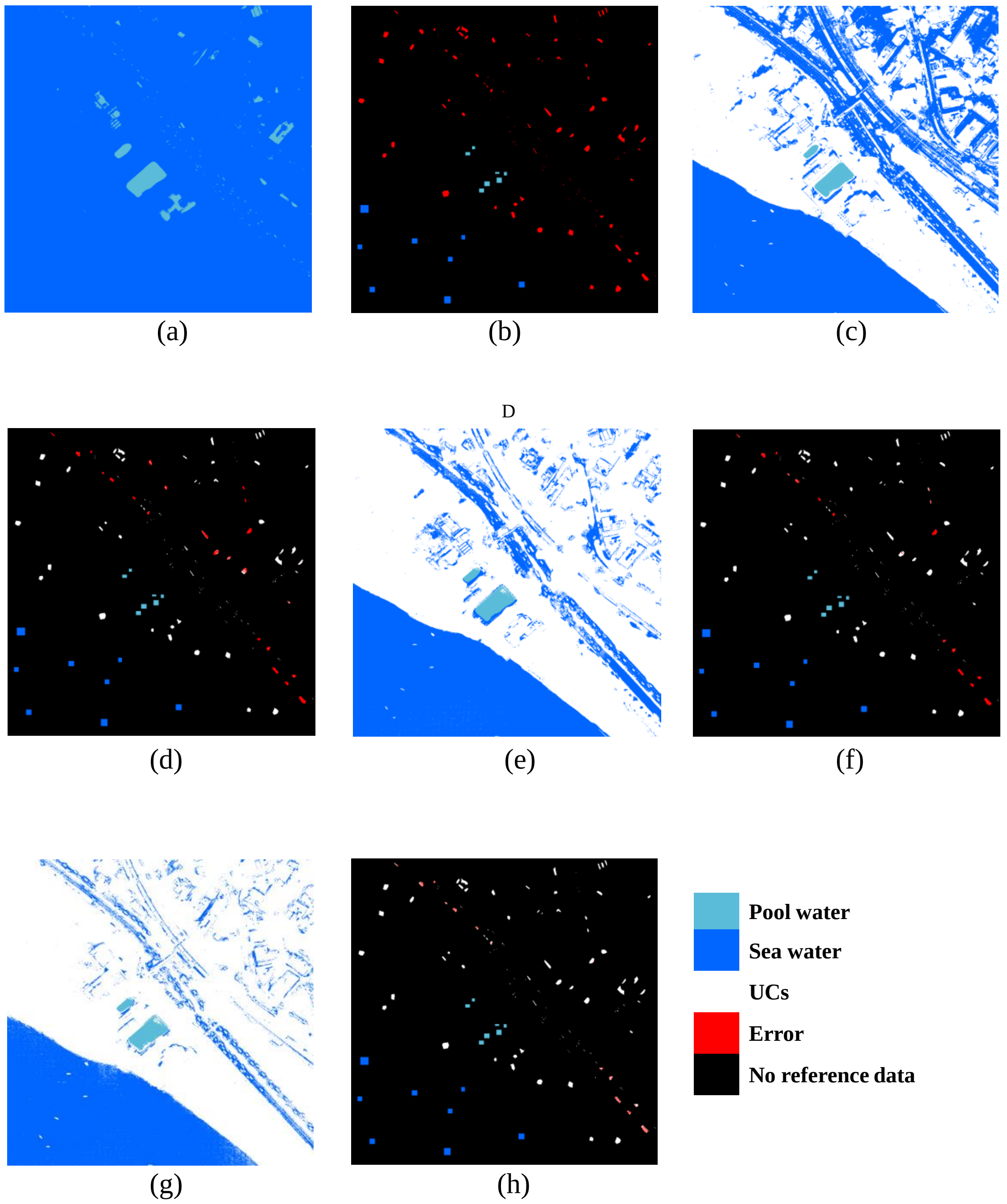


Figure 6.9: Classified images produced by the list of classifiers for the specific realization of WV3 dataset. The maps (a) and (b) are produced by SVM. (c) and (d) are produced by OSNN^{NNDR}. (e) and (f) are produced by CAG-SC^{2}S using SVM as SC. (g) and (h) are produced by CAL-SC^{2}S using SVM as SC. The class predictions by algorithms are performed only on the ground truth reference data to show the errors produced by the classifiers for the UCs.

are illustrated using 3D bar plots. The SVM and DCCN methods presented the worst FNR of 57.23% to 92.97% and FPR 20.60% to 58.71% values for all the images. In comparison, the CAL-SC^{2}S using SVM offered relatively better FNR of 4.54% to 56.55% and FPR of 0.52% to 7.73% values across all the datasets. From Table 6.4, Figures 6.8 (g), and 6.8(h), we can confirm that the CAL-SC^{2}S using the SVM method presented superior performance among the considered algorithms.

Figure 6.9 visually highlights the better insights to the errors produced by the classifiers. In this illustration, the training set included 687 pixel vectors of 2 KCs (i.e., pool water and river water), whereas the testing set included 21197 (6870 of KCs and 14327 of UCs) samples of all possible class examples (i.e., 8). The (OA, AA, FPR, FNR) % results of SVM, OSNN^{NNDR}, CAG-SC^{2}S, and CAL-SC^{2}S are (32.41, 66.67, 80.67, 51.05), (80.89, 90.58, 32.07, 10.56), (87.51, 93.72, 22.28, 6.69), and (94.65, 97.12, 10.16, 2.75), respectively. Overall, the classification maps in Figure 6.9 shows that CAL-SC^{2}S with SVM as SC achieved superior classification performance. In the above discussed four case studies, the instability or fluctuations in SD percentage values for both OA and AA across all the test cases is due to the following reasons. First, due to the random selection of the training samples, the model training is subjective to the quality of the training set used. Second, the accuracy performance also depends on the KCs, UCs, and their intraclass and interclass relations with each other.

6.6.5 FPGA-based hardware design performance evaluation of the list of classifiers using case study V

The computation time is one of the most important indicators along with classification accuracy which determines the efficacy of the algorithm. For this reason, the computational advantage of the proposed approach is obtained by restructuring the architecture of the classification model of CA-SC^{2}S-I to CA-SC^{2}S-II. The computational speed of Type-II models of CA-SC^{2}S can be increased by making use of a parallel computing strategy. In this case study, we assess both the classification accuracy and computation time of the classification algorithms for the specific realization of the test case of ANG image. The test case scenario consists of two KCs namely, river water and lake water. And the remaining four classes are considered as UCs. The five classifiers considered in this case study are SVM, CAG-SC^{2}S-I, CAL-SC^{2}S-I, CAG-SC^{2}S-II, and CAL-SC^{2}S-II. As shown in Table 6.5, evaluation is performed using metrics like the number of SVs, classification accuracies, resource usage, latency, throughput, computation time, and power usage. As can be seen in Table 6.5 (Top), the number of support vectors is more for CAL-SC^{2}S methods, and the least is for the SVM method. The CAL-SC^{2}S methods are two-stage systems consisting of L-OCSVM (with two OCSVM models) and SVM as subsystems. By analyzing the accuracy and error values, there are statistically significant differences between the SVM and four variants of CA-SC^{2}S using SVM. It should be noted that the two models CAL-SC^{2}S-I and CAL-SC^{2}S-II, provide the same classification results as both consist of identical subsystems with different connections. Similarly, CAG-SC^{2}S-I and CAG-SC^{2}S-II also present the same results. However, CAL-SC^{2}S models achieved the highest classification accuracies with the least error rates.

As can be seen in Table 6.5 (Bottom), FPGA-based results obtained by the list of classifiers are presented. For the ANG HSI dataset, the real-time constraint of 15.6 μs is defined by the sensor scanning time for the per-pixel vector. To achieve real-time or near real-time processing of one pixel vector, the processing system must classify the pixel vector within 15.6 μs (Basterretxea *et al.*, 2016; Dubacharla and Nidamanuri, 2020). In FPGA-based classification systems, the processing time mainly depends on the F_{max} (in MHz) and the clock cycles or latency(Xilinx User Guide, 2016; Zhang, 2017). As expected, the CAG-SC^{2}S-I and CAL-SC^{2}S-I methods reported large clock cycle values, i.e., double the dimensionality of the data. Whereas the SVM, CAG-SC^{2}S-II, and CAL-SC^{2}S-II used $d + 20$ clock cycles to classify one pixel vector. It should be noted that the delay or latency is added only at the beginning of the computation, and in the later processing stages, there is no latency in processing the inputs. The main reason for this is due to pipelining feature in the proposed designs. As can be observed from Table 6.5, all the five FPGA designs of the classifiers have obtained a F_{max} of at least 65 MHz with a minimum hardware resource usage. The FPGA processing times of the five classifiers indicate that the FPGA designs have achieved the real-time processing of per-pixel vector classification by satisfying the

Table 6.5: Assessment of the classification accuracies (Top), errors (Top), and real-time processing (Bottom) results using FPGA obtained by the list of five classification techniques for the specific realization of ANG HSI dataset. The results are estimated for the considered realization of ANG image with river water and lake water as two KCs in case study V.

Techniques	**No. of SVs**	**Accuracy [%]**		**Error [%]**	
		OA	**AA**	**FNR**	**FPR**
SVM	2	20.71	66.66	66.12	100
CAG-SC^{2}S-I	4	98.30	98.55	1.79	1.60
CAL-SC^{2}S-I	6	99.69	99.03	1.70	1.47
CAG-SC^{2}S-II	4	98.30	98.55	1.79	1.60
CAL-SC^{2}S-II	6	99.69	99.03	1.70	1.47

Techniques	**No. of SVs**	**Resource usage [%]**	**F_{max} (MHz)**	**Per pixel vector**			**Throughput (Pixel vectors/s)**	**FPGA computation time (in s)**		**On-chip power (W)**
				Latency						
				Real-time constraint	**Clock cycles**	**FPGA processing time**		**Only KC samples**	**Full image**	
SVM	2	18.73	72.40		371	5.12 μs	1,953,125	0.0269	2.16	0.224
CAG-SC^{2}S-I	4	35.37	69.90		740	10.59 μs	944,287	0.055	4.48	0.308
CAL-SC^{2}S-I	6	51.69	65.60	15.6 μs	741	11.29 μs	885,739	0.059	4.78	0.351
CAG-SC^{2}S-II	4	33.31	70.40		371	5.27 μs	1,897,533	0.0277	2.23	0.312
CAL-SC^{2}S-II	6	43.32	65.20		372	5.70 μs	1,754,385	0.0299	2.41	0.358

timing constraint. As observed from the table, the Type-II models have exhibited an improvement in processing time compared to the Type-I models. Likewise, the low-latency values of the FPGA designs have achieved high throughput rates, leading to fast computation. Further, the power usage estimates of the five FPGA design reported in the table indicate the low power usage of the classifiers with less significant differences. Overall, the Type-II models of CAG-SC^{2}S and CAL-SC^{2}S have achieved the highest accuracy results along with the fast computation time by taking advantage of pipeline parallelism.

6.7 Chapter Conclusions

This chapter assessed the accuracy and real-time computation performance of the proposed CAL-SC^{2}S algorithms using MSIs and HSIs. The comprehensive experimental evaluation using five different case studies containing 241 test case classification scenarios is presented. The experimental analyses state that the choice of including reject-option in the classification tasks can significantly increase the reliability of the class predictions in real-world dynamic environments. The classification accuracy results of the proposed CAL-SC^{2}S algorithm with real-time processing capabilities using FPGA demonstrate the efficiency of the proposed approach.

CHAPTER 7

DISCUSSIONS AND MAJOR CONTRIBUTIONS

Prelude: In this chapter, a comprehensive discussion on the overall observations of the studies presented in the previous chapters of this thesis is briefly described. Further, a detailed summary of the important contributions of this thesis to the state-of-the-art remote sensing image analysis is also presented in this chapter.

7.1 Discussions

This thesis has developed and implemented novel algorithmic schemes to make robust and efficient class predictions in real-world, real-time environments using MSIs/HSIs. The improvements to the existing methods have been proposed in two different areas of image classification, namely, accuracy and computation time. Many studies have been conducted on the importance of classification accuracies for MSIs/HSIs, but only a few studies were conducted about other critical issues, such as the simplicity and speed of the approaches. However, several novel classification techniques proposed in the literature are typically aimed at improving the accuracy and processing time in the closed-set environment. They do not account for the dynamic, open-set environments. In fact, multiple studies have demonstrated the potential use of MSI/HSI classification applications across a diverse range of scientific and engineering domains. To be deployed in real-world application environments, the classifiers must produce reliable predictions regardless of the environments or inputs; otherwise, they may end up with suboptimal performance.

Regarding the computation time requirements, the FPGA-based MSVM designs proposed in Chapter 3 offers image classification to be performed in real-time constraints with minimum latency. The Arty-35T based designs showed a speedup of 38.7 times the software version for the Pavia University dataset. And for the new generation, AVIRIS-NG HSI sensor, a speedup of 14.2 times the software-based execution is achieved. The hardware designs were developed using a low-complexity, architectural-level tool. One of the advantages of the proposed hardware-based MSVM designs is that the computation time is independent of the number of support vectors. However, the MSVM classification algorithm has presented good accuracies for the KCs, but simultaneously, it produced unnoticed omission errors for the unseen classes. These erroneous class predictions for UCs can be seen in the classified images shown in Figure 3.7. It is understood that the forced assignment problem produces errors for UC samples. In the past several researchers have proposed techniques to address the forced assignment problem of classifiers for UCs. Still, these methods are complex to operate, and also few of them are sensitive to initial settings. The proposed SC^{2}S method provides a simple scheme yet practical approach to handle the UCs. The two-stage SC^{2}S algorithm presents an improvement over SVM concerning OA in the range of 1% to 74% and errors rate in the range of 3% to 74% (see Table 4.2). In the case of open-set scenarios, SC^{2}S offers superior classification performance over SVM while maintaining fairly equal performance for closed-set scenarios.

Some of the advantages of using the SC^{2}S algorithm are as follows. First, the SC^{2}S method can perform reliable predictions for both MSIs and HSIs. Second, the SC^{2}S algorithm is simple to use and modify accordingly to application or user requirements. Third, classification errors can be traced easily due to the two-stage architecture. We explored the potential of SC^{2}S by testing the prediction performance using several test case scenarios. Although the SC^{2}S method obtained better classification results than RF, SVM, OSNN^{CV}, OSNN^{NNDR}, and P-SVM, there are a few shortcomings or limitations (see Figure 4.5). One of the limitations of SC^{2}S is that it fails to find the similarity between the IL-KC sample and the IS-KC sample. Whereas the proposed SI^{2}CS and SMMs in Chapter 5 overcome the limitation of not grouping the IS and high IL sample as one categorical class. As shown in Table 5.1, there is a good improvement of accuracies by SI^{2}CS in the range of 4% to 40% over the traditional OCC methods. Further, the shadow detection rate of SI^{2}CS and SMMs is increased by 20% to 90%. However, there are still a few influential factors limiting the accuracy of the proposed algorithms, as discussed in Section 5.5.4.

Another shortcoming of the SC^{2}S or CAG-SC^{2}S method is that there are still a few forced assignment errors for UCs. The reason for these errors is due to the usage of a global-level novelty detection scheme in SNC. In other words, a single global decision boundary is used to find the novelty samples. This novelty detection scheme is insensitive to UCs present in between the KCs. This shortcoming is overcome by the proposed CAL-SC^{2}S method that considers m decision boundaries for m KCs instead of using only one boundary. There is a significant improvement in the classification accuracies by CAL-SC^{2}S over the CAG-SC^{2}S and other existing methods. But, to achieve reliable and high classification accuracy, CAL-SC^{2}S sacrificed the processing time. Hence, it will be beneficial to have an efficient framework that reduces the computation burden without reducing accuracy. In this regard, Type-II architecture for CA-SC^{2}S algorithms is proposed to enhance the processing time capabilities of existing Type-I architecture. Compared to the CAG-SC^{2}S-I and CAL-SC^{2}S-I methods, CAG-SC^{2}S-II and CAL-SC^{2}S-II have presented significantly better improvement of at least 5 μs in FPGA processing time. The number of support vectors can further influence this improvement.

7.2 Major Contributions of Thesis

In this section, we briefly summarize the main contributions of this thesis. We partition them into three different areas and classify them accordingly to the area they naturally fit.

1. *In the area of design and implementation of FPGA-based real-time HSI classification, the contribution is following:*

A novel FPGA-based hardware architecture design approach is developed using high-level design tools for pixel-level hyperspectral data classification. A rapid FPGA prototyping approach using low-complexity design tools for real-time HSI classification is presented. This approach is based on performing multiclass prediction by streaming pixels into the predesigned FPGA fabric. The proposed hardware FPGA design performs real-time processing under strict time, hardware fabric, and power constraints.

2. *In the area of closed-set and open-set MSI/HSI classification in real-world, real-time environments, the main contribution is following:*

The developed novel image classification techniques are highly adaptive to both static and dynamic environments. In general, image classification techniques are designed to work under the assumption of static environments where the classifier faces the same set of KCs in training and testing. However, practical applications are dynamic and open-set, i.e., the presence of unseen or UCs is unavoidable. To address this issue, novel two-stage approaches with different model architectures are developed to perform classification efficiently in real-world, real-time environments.

3. *In the area of shadow and illumination invariant image classification, the contribution is following:*

The proposed set of new categories of shadow invariant image classification algorithms using high-resolution MSIs and HSIs are easily generalizable. It is known that shadows are one of the ever-existing ubiquitous challenges in image analysis and still an active area of research. As an affordable solution, the proposed novel approaches can successfully perform invariant classification of shadows and illumination effect regions by taking advantage of the inherent spectral similarity between IL and IS material spectra. Unlike existing methods, the proposed techniques are relatively simple and utilize the same amount of training data that is required for any supervised MSI/HSI classification.

CHAPTER 8

SUMMARY, CONCLUSIONS, AND FUTURE DIRECTIONS

Prelude: This chapter summarizes the overall observations and conclusions of the studies presented in the previous chapters of this thesis. Further, recommendations and future research directions in this high-impact and diverse application area of remote sensing image analysis research are also presented in this chapter.

8.1 Summary and Conclusions

This thesis has investigated and addressed the vital research requirement of developing robust and efficient image classification algorithms. This thesis proposed four different novel image classification frameworks to improve the accuracy and processing time. For ease of readability, we summarize the major conclusions of this thesis as chapter-wise below.

- In Chapter 3, a low-complexity hardware design procedure is introduced to perform FPGA-based MSI/HSI classification in real-time environments. The MSVM algorithm is designed using a low-power, low-cost, and small-size Arty-35T FPGA board. The XSG tool used to verify the hardware logic presents a quick debug process in the MATLAB-Simulink environment. The results obtained by the thesis indicate the potential future of FPGAs in the development of reconfigurable-based online MSI/HSI classification in dynamic environments.
- Chapter 3 also demonstrated the forced assignment problem of classifiers when the information classes are under-sampled.
- Chapter 4 presented an efficient and robust classification algorithm named SC^{2}S, which performs class prediction even in the presence of UCs.

- Chapter 4 also demonstrated a benchmark study that can be used to evaluate the performance of classification algorithms in open-set test case scenarios of real-world application environments. The SC^{2}S algorithm developed in this thesis shows a positive scope to improves the reliability of classification performance. Hence the research queries and hypotheses considered in this thesis may be further extended as future studies.
- Chapter 5 introduced a set of completer automatic frameworks for shadow and illumination invariant classification using spectral signatures. This chapter also presented a benchmark study to design experimental cases and measure the shadow detection rate of an algorithm.
- Chapter 6 extended the research problem and hypothesis of Chapter 4 to overcome the shortcomings of SC^{2}S or CAG-SC^{2}S algorithms. A new algorithm called CAL-SC^{2}S is proposed to minimize the false predictions for UCs further. Moreover, a new modification to the existing Type-I architecture is proposed called Type-II to provide computational benefits.

8.2 Recommendations and Future Research Directions

Image classification is one of the most vibrant areas, especially in the remote sensing field. The following are recommendations for future work related to MSI/HSI classification

- Other than FPGAs, IC devices and systems such as ASICs, PSoCs, GPUs, and cluster computing can be used to explore the time-critical potential hardware candidate solutions. Moreover, using different programming approaches and architectures for acceleration could also be of interest for future work. Indeed, minimal studies are reported in this area.
- Typically for initiation of the classification work in a real-time environment, pre-acquired images are used. However, we recommend that if a system connected in real-time mode, and deployment of algorithms proposed in this thesis on an onboard computation system will be much more critical to understand the intricacies of real-time image classification perspectives.

- The extension and investigation of the proposed techniques on different HSIs/MSIs containing diverse LULC settings and varying imaging environment parameters could also be of interest. Further studies in this direction can increase the reliability of class predictions in real-world environments. We believe that the increase in accuracy could be improved by including features other than spectral information.
- Some critical problems challenging remote sensing image processing to extract user-desired relevant information are analyzing the UCs, shadows, and changing illumination effects. Devising new frameworks and addressing the challenges posed by the uncertainties mentioned above can be undertaken for future study.

8.3 Acknowledgements for Data

I would like to acknowledge Prof. Paolo Gamba from TRSL for providing the Pavia center and Pavia University dataset freely online. I would also like to acknowledge SpaceNet of DigitalGlobe for providing WV3 satellite MSI and Space Application Laboratory, Department of Advanced Interdisciplinary studies, the University of Tokyo for providing the CHK hyperspectral data.

REFERENCES

1. Adep, R. N., shetty, A. and Ramesh, H. (2017) ‘EXhype: A tool for mineral classification using hyperspectral data’, *ISPRS Journal of Photogrammetry and Remote Sensing*, 124, pp. 106–118. doi: 10.1016/j.isprsjprs.2016.12.012.

2. Adler-Golden, S. M. *et al.* (2001) ‘Shadow-insensitive material detection/classification with atmospherically corrected hyperspectral imagery’, in *Algorithms for Multispectral, Hyperspectral, and Ultraspectral Imagery VII*. International Society for Optics and Photonics, pp. 460–469.

3. Akhtar, N. and Mian, A. (2018) ‘Hyperspectral recovery from rgb images using gaussian processes’, *IEEE transactions on pattern analysis and machine intelligence*, 42(1), pp. 100–113.

4. Basterretxea, K. *et al.* (2016) ‘Elm-based hyperspectral imagery processor for onboard real-time classification’, in *Design and Architectures for Signal and Image Processing (DASIP), 2016 Conference on*. IEEE, pp. 43–50.

5. Behmann, J. *et al.* (2018) ‘Specim IQ: Evaluation of a New, Miniaturized Handheld Hyperspectral Camera and Its Application for Plant Phenotyping and Disease Detection’, *Sensors*, 18(2), p. 441. doi: 10.3390/s18020441.

6. Bendale, A. and Boult, T. (2015) ‘Towards Open World Recognition’, in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA: IEEE, pp. 1893–1902. doi: 10.1109/CVPR.2015.7298799.

7. Bernabe, S. *et al.* (2011) ‘FPGA Design of an Automatic Target Generation Process for Hyperspectral Image Analysis’, in. IEEE, pp. 1010–1015. doi: 10.1109/ICPADS.2011.64.

8. Bioucas-Dias, J. M. *et al.* (2013) ‘Hyperspectral Remote Sensing Data Analysis and Future Challenges’, *IEEE Geoscience and Remote Sensing Magazine*, 1(2), pp. 6–36. doi: 10.1109/MGRS.2013.2244672.

9. Breiman, L. (2001) ‘Random forests’, *Machine learning*, 45(1), pp. 5–32.

10. Camps-Valls, G. *et al.* (2004) ‘Robust support vector method for hyperspectral data classification and knowledge discovery’, *IEEE Transactions on Geoscience and Remote Sensing*, 42(7), pp. 1530–1542. doi: 10.1109/TGRS.2004.827262.

11. Chakrabarti, A. and Zickler, T. (2011) ‘Statistics of real-world hyperspectral images’, in *CVPR 2011*. IEEE, pp. 193–200.

12. Chang, C.-C. and Lin, C.-J. (2011) ‘LIBSVM: a library for support vector machines’, *ACM transactions on intelligent systems and technology (TIST)*, 2(3), p. 27.

13. Chang, C.-I. (2003) *Chein-I Chang - Hyperspectral Imaging_ Techniques for Spectral Detection and Classification (2003, Springer).pdf*. Springer.

14. Chang, C.-I. (2016) *Real-time progressive hyperspectral image processing*. Springer.

15. Chang, C.-I., Ren, H. and Chiang, S.-S. (2001) ‘Real-time processing algorithms for target detection and classification in hyperspectral imagery’, *IEEE Transactions on Geoscience and Remote Sensing*, 39(4), pp. 760–768.

16. Chein-I Chang (1999) ‘Spectral information divergence for hyperspectral image analysis’, in *IEEE 1999 International Geoscience and Remote Sensing Symposium. IGARSS’99 (Cat. No.99CH36293)*. *IEEE 1999 International Geoscience and Remote Sensing Symposium. IGARSS’99 (Cat. No.99CH36293)*, pp. 509–511 vol.1. doi: 10.1109/IGARSS.1999.773549.

17. Chow, C. (1970) ‘On optimum recognition error and reject tradeoff’, *IEEE Transactions on information theory*, 16(1), pp. 41–46.

18. Clark, R. N. *et al.* (2003) ‘Imaging spectroscopy: Earth and planetary remote sensing with the USGS Tetracorder and expert systems: IMAGING SPECTROSCOPY REMOTE SENSING’, *Journal of Geophysical Research: Planets*, 108(E12). doi: 10.1029/2002JE001847.

19. Condessa, F., Bioucas-Dias, J. and Kovačević, J. (2016) ‘Supervised hyperspectral image classification with rejection’, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(6), pp. 2321–2332.

20. Cortes, C. and Vapnik, V. (1995) ‘Support-vector networks’, *Machine learning*, 20(3), pp. 273–297.

21. Dalponte, M. *et al.* (2009) ‘The role of spectral resolution and classifier complexity in the analysis of hyperspectral images of forest areas’, *Remote Sensing of Environment*, 113(11), pp. 2345–2355. doi: 10.1016/j.rse.2009.06.013.

22. Damodaran, B. B. and Nidamanuri, R. R. (2014) ‘Assessment of the impact of dimensionality reduction methods on information classes and classifiers for hyperspectral image classification by multiple classifier system’, *Advances in Space Research*, 53(12), pp. 1720–1734. doi: 10.1016/j.asr.2013.11.027.

23. Dare, P. M. (2005) ‘Shadow analysis in high-resolution satellite imagery of urban areas’, *Photogrammetric Engineering & Remote Sensing*, 71(2), pp. 169–177.

24. De Rosa, R., Mensink, T. and Caputo, B. (2016) ‘Online Open World Recognition’, *arXiv:1604.02275 [cs, stat]*. Available at: http://arxiv.org/abs/1604.02275 (Accessed: 14 July 2021).

25. Du, Q. and Nekovei, R. (2009) ‘Fast real-time onboard processing of hyperspectral imagery for detection and classification’, *Journal of Real-Time Image Processing*, 4(3), pp. 273–286. doi: 10.1007/s11554-008-0106-9.

26. Duda, R. O. and Hart, P. E. (1973) *Pattern classification and scene analysis*. Wiley New York.

27. Eismann, M. T. (2012) *Hyperspectral Remote Sensing*. 1000 20th Street, Bellingham, WA 98227-0010 USA: SPIE. doi: 10.1117/3.899758.

28. Estlick, M. *et al.* (2001) ‘Algorithmic transformations in the implementation of K-means clustering on reconfigurable hardware’, in *Proceedings of the 2001 ACM/SIGDA ninth international symposium on Field programmable gate arrays*. ACM, pp. 103–110.

29. Farfan-Escobedo, J. D., Enciso-Rodas, L. and Vargas-Munoz, J. E. (2017) ‘Towards accurate building recognition using convolutional neural networks’, in *2017 IEEE XXIV International Conference on Electronics, Electrical Engineering and Computing (INTERCON)*. IEEE, pp. 1–4.

30. Feist, T. (2012) ‘Vivado design suite’, *White Paper*, 5.

31. Foody, G. M. and Atkinson, P. M. (eds) (2002) *Uncertainty in remote sensing and GIS*. Chichester: Wiley.

32. Fowers, J. *et al.* (2012) ‘A performance and energy comparison of FPGAs, GPUs, and multicores for sliding-window applications’, in *Proceedings of the ACM/SIGDA international symposium on Field Programmable Gate Arrays*. ACM, pp. 47–56.

33. Ghamisi, P. *et al.* (2017) ‘Advanced Spectral Classifiers for Hyperspectral Images: A review’, *IEEE Geoscience and Remote Sensing Magazine*, 5(1), pp. 8–32. doi: 10.1109/MGRS.2016.2616418.

34. Goetz, A. F. H. (2009) ‘Three decades of hyperspectral remote sensing of the Earth: A personal view’, *Remote Sensing of Environment*, 113, pp. S5–S16. doi: 10.1016/j.rse.2007.12.014.

35. González, C. *et al.* (2013) ‘Use of FPGA or GPU-based architectures for remotely sensed hyperspectral image processing’, *Integration, the VLSI Journal*, 46(2), pp. 89–103. doi: 10.1016/j.vlsi.2012.04.002.

36. Gorte, B. and Gorte-Kroupnova, N. (1995) ‘Non-parametric classification algorithm with an unknown class’, in *iscv*. IEEE, p. 443.

37. Gowen, A. *et al.* (2007) ‘Hyperspectral imaging – an emerging process analytical tool for food quality and safety control’, *Trends in Food Science & Technology*, 18(12), pp. 590–598. doi: 10.1016/j.tifs.2007.06.001.

38. Dubacharla, G and Nidamanuri, R. R. (2020) ‘A real-time FPGA accelerated stream processing for hyperspectral image classification’, *Geocarto International*, 0(0), pp. 1–18. doi: 10.1080/10106049.2020.1713231.

39. Dubacharla, G. and Nidamanuri, R. R. (2021) ‘A Novel Supervised Cascaded Classifier System (SC^{2}S) for Robust Remote Sensing Image Classification’, *IEEE Geoscience and Remote Sensing Letters*, 18(3), pp. 421–425. doi: 10.1109/LGRS.2020.2980186.

40. Hagen, N. A. and Kudenov, M. W. (2013) ‘Review of snapshot spectral imaging technologies’, *Optical Engineering*, 52(9), p. 090901.

41. Halicek, M. *et al.* (2017) ‘Deep convolutional neural networks for classifying head and neck cancer using hyperspectral imaging’, *Journal of Biomedical Optics*, 22(6), p. 060503. doi: 10.1117/1.JBO.22.6.060503.

42. Homenda, W. and Jastrzebska, A. (2015) ‘Global, local and embedded architectures for multiclass classification with foreign elements rejection: An

overview', in *2015 7th International Conference of Soft Computing and Pattern Recognition (SoCPaR). 2015 7th International Conference of Soft Computing and Pattern Recognition (SoCPaR)*, pp. 89–94. doi: 10.1109/SOCPAR.2015.7492789.

43. Hu, W. *et al.* (2015) 'Deep convolutional neural networks for hyperspectral image classification', *Journal of Sensors*, 2015.

44. Jiang, H. *et al.* (2018) 'To trust or not to trust a classifier', in *Advances in neural information processing systems*, pp. 5541–5552.

45. Jun, G. and Ghosh, J. (2013) 'Semisupervised Learning of Hyperspectral Data With Unknown Land-Cover Classes', *IEEE Transactions on Geoscience and Remote Sensing*, 51(1), pp. 273–282. doi: 10.1109/TGRS.2012.2198654.

46. Júnior, P. R. M. *et al.* (2017) 'Nearest neighbors distance ratio open-set classifier', *Machine Learning*, 106(3), pp. 359–386.

47. Kemker, R., Salvaggio, C. and Kanan, C. (2018) 'Algorithms for semantic segmentation of multispectral remote sensing imagery using deep learning', *ISPRS journal of photogrammetry and remote sensing*, 145, pp. 60–77.

48. Khan, M. J. *et al.* (2018) 'Modern Trends in Hyperspectral Image Analysis: A Review', *IEEE Access*, 6, pp. 14118–14129. doi: 10.1109/ACCESS.2018.2812999.

49. Khazai, S. *et al.* (2012) 'Improving the SVDD Approach to Hyperspectral Image Classification', *IEEE Geoscience and Remote Sensing Letters*, 9(4), pp. 594–598. doi: 10.1109/LGRS.2011.2176101.

50. Liu, D. and Han, L. (2017) 'Spectral curve shape matching using derivatives in hyperspectral images', *IEEE Geoscience and Remote Sensing Letters*, 14(4), pp. 504–508.

51. Liu, X. *et al.* (2019) 'Classification of Hyperspectral Image by CNN Based on Shadow Area Enhancement Through Dynamic Stochastic Resonance', *IEEE Access*, 7, pp. 134862–134870. doi: 10.1109/ACCESS.2019.2941872.

52. Lopez, S. *et al.* (2013) 'The Promise of Reconfigurable Computing for Hyperspectral Imaging Onboard Systems: A Review and Trends', *Proceedings of the IEEE*, 101(3), pp. 698–722. doi: 10.1109/JPROC.2012.2231391.

53. Madroñal, D. *et al.* (2017) ‘SVM-based real-time hyperspectral image classifier on a manycore architecture’, *Journal of Systems Architecture*, 80, pp. 30–40. doi: 10.1016/j.sysarc.2017.08.002.

54. Makki, I. *et al.* (2017) ‘A survey of landmine detection using hyperspectral imaging’, *ISPRS Journal of Photogrammetry and Remote Sensing*, 124, pp. 40–53. doi: 10.1016/j.isprsjprs.2016.12.009.

55. Manolakis, D., Marden, D. and Shaw, G. A. (2003) ‘Hyperspectral image processing for automatic target detection applications’, *Lincoln laboratory journal*, 14(1), pp. 79–116.

56. Mantero, P., Moser, G. and Serpico, S. B. (2005) ‘Partially supervised classification of remote sensing images through SVM-based probability density estimation’, *IEEE Transactions on Geoscience and Remote Sensing*, 43(3), pp. 559–570.

57. Mather, P. and Tso, B. (2016) *Classification methods for remotely sensed data*. CRC press.

58. van der Meero, F. and Bakker, W. (1997) ‘Cross correlogram spectral matching: Application to surface mineralogical mapping by using AVIRIS data from Cuprite, Nevada’, *Remote Sensing of Environment*, 61(3), pp. 371–382. doi: 10.1016/S0034-4257(97)00047-3.

59. Mũnoz-Marí, J. *et al.* (2010) ‘Semisupervised One-Class Support Vector Machines for Classification of Remote Sensing Data’, *IEEE Transactions on Geoscience and Remote Sensing*, 48(8), pp. 3188–3197. doi: 10.1109/TGRS.2010.2045764.

60. Muzzolini, R., Yang, Y.-H. and Pierson, R. (1998) ‘Classifier design with incomplete knowledge’, *Pattern Recognition*, 31(4), pp. 345–369.

61. Nascimento, S. M. C., Amano, K. and Foster, D. H. (2016) ‘Spatial distributions of local illumination color in natural scenes’, *Vision Research*, 120, pp. 39–44. doi: 10.1016/j.visres.2015.07.005.

62. Otsu, N. (1979) ‘A threshold selection method from gray-level histograms’, *IEEE transactions on systems, man, and cybernetics*, 9(1), pp. 62–66.

63. Papadonikolakis, M. and Bouganis, C. (2012) ‘Novel Cascade FPGA Accelerator for Support Vector Machines Classification’, *IEEE Transactions on Neural Networks and Learning Systems*, 23(7), pp. 1040–1052. doi: 10.1109/TNNLS.2012.2196446.

64. Piñeros, M. F., Ritchie, E. A. and Tyo, J. S. (2011) 'Estimating Tropical Cyclone Intensity from Infrared Image Data', *Weather and Forecasting*, 26(5), pp. 690–698. doi: 10.1175/WAF-D-10-05062.1.

65. Pingree, P. J., Scharenbroich, L. J. and Werne, T. A. (2008) 'Implementing legacy-C algorithms in FPGA co-processors for performance accelerated smart payloads'.

66. Platt, J. (1999) 'Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods', *Advances in large margin classifiers*, 10(3), pp. 61–74.

67. Plaza, A. J. (2008) 'Clusters versus FPGAs for spectral mixture analysis-based lossy hyperspectral data compression', in Huang, B., Heymann, R. W., and Serra-Sagrista, J. (eds), p. 708402. doi: 10.1117/12.798326.

68. Prasad, S., Bruce, L. M. and Chanussot, J. (2011) 'Introduction', in Prasad, S., Bruce, L. M., and Chanussot, J. (eds) *Optical Remote Sensing: Advances in Signal Processing and Exploitation Techniques*. Berlin, Heidelberg: Springer (Augmented Vision and Reality), pp. 1–8. doi: 10.1007/978-3-642-14212-3_1.

69. Qiao, X., Yuan, D. and Li, H. (2017) 'Urban Shadow Detection and Classification Using Hyperspectral Image', *Journal of the Indian Society of Remote Sensing*, 45(6), pp. 945–952. doi: 10.1007/s12524-016-0649-3.

70. Rüfenacht, D., Fredembach, C. and Süsstrunk, S. (2014) 'Automatic and Accurate Shadow Detection Using Near-Infrared Information', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8), pp. 1672–1678. doi: 10.1109/TPAMI.2013.229.

71. Saari, H. *et al.* (2017) 'Visible, very near ir and short wave ir hyperspectral drone imaging system for agriculture and natural water applications', *Int. Archives Photogrammetry, Remote Sens. Spatial Inf. Sci.*, 42, pp. 165–170.

72. Sabins, F. F. (1999) 'Remote sensing for mineral exploration', *Ore Geology Reviews*, 14(3–4), pp. 157–183.

73. Salem, F. and Kafatos, M. (2001) 'Hyperspectral image analysis for oil spill mitigation', in *Paper presented at the 22nd Asian Conference on Remote Sensing*, p. 9. Available at: http://www.crisp.nus.edu.sg/~acrs2001/pdf/108SALEM.PDF (Accessed: 23 August 2017).

74. Sanin, A., Sanderson, C. and Lovell, B. C. (2012) ‘Shadow detection: A survey and comparative evaluation of recent methods’, *Pattern Recognition*, 45(4), pp. 1684–1695. doi: 10.1016/j.patcog.2011.10.001.

75. Scheirer, W. J., Jain, L. P. and Boult, T. E. (2014) ‘Probability models for open set recognition’, *IEEE transactions on pattern analysis and machine intelligence*, 36(11), pp. 2317–2324.

76. Schölkopf, B. *et al.* (2000) ‘Support vector method for novelty detection’, in *Advances in neural information processing systems*, pp. 582–588.

77. SpaceNet on Amazon Web Services (AWS). (no date) ‘Datasets’, *The SpaceNet Catalog*. Available at: https://spacenet.ai/datasets/ (Accessed: 18 August 2020).

78. Sun, D.-W. (ed.) (2010) *Hyperspectral imaging for food quality analysis and control*. 1. ed. London: Academic.

79. Taneja, A., Ballan, L. and Pollefeys, M. (2015) ‘Geometric change detection in urban environments using images’, *IEEE transactions on pattern analysis and machine intelligence*, 37(11), pp. 2193–2206.

80. Tax, D. M. J. and Duin, R. P. W. (1999) ‘Support vector domain description’, *Pattern Recognition Letters*, 20(11), pp. 1191–1199. doi: 10.1016/S0167-8655(99)00087-2.

81. Tessier, R., Pocek, K. and DeHon, A. (2015) ‘Reconfigurable computing architectures’, *Proceedings of the IEEE*, 103(3), pp. 332–354.

82. Thenkabail, P. S. and Lyon, J. G. (2016) *Hyperspectral remote sensing of vegetation*. CRC press.

83. Thorpe, A. K. *et al.* (2016) ‘Mapping methane concentrations from a controlled release experiment using the next generation airborne visible/infrared imaging spectrometer (AVIRIS-NG)’, *Remote Sensing of Environment*, 179, pp. 104–115. doi: 10.1016/j.rse.2016.03.032.

84. Wang, S. *et al.* (2016) ‘A Scalable Dataflow Accelerator for Real Time Onboard Hyperspectral Image Classification’, in *International Symposium on Applied Reconfigurable Computing*. Springer, pp. 105–116. Available at: http://link.springer.com/chapter/10.1007/978-3-319-30481-6_9 (Accessed: 19 July 2017).

85. Windrim, L. *et al.* (2018) ‘A Physics-Based Deep Learning Approach to Shadow Invariant Representations of Hyperspectral Images’, *IEEE Transactions on Image Processing*, 27(2), pp. 665–677. doi: 10.1109/TIP.2017.2761542.

86. Xilinx User Guide (2016) ‘Vivado Design Suite User Guide: Model-Based DSP Design using System Generator, UG897, v2016.1 ed.’

87. Yokoya, N. and Iwasaki, A. (2016) ‘Airborne hyperspectral data over Chikusei’, *Space Appl. Lab., Univ. Tokyo, Tokyo, Japan, Tech. Rep. SAL-2016-05-27*.

88. Yue, J. *et al.* (2017) ‘Estimation of Winter Wheat Above-Ground Biomass Using Unmanned Aerial Vehicle-Based Snapshot Hyperspectral Sensor and Crop Height Improved Models’, *Remote Sensing*, 9(7), p. 708. doi: 10.3390/rs9070708.

89. Zebin Wu *et al.* (2015) ‘GPU Implementation of Composite Kernels for Hyperspectral Image Classification’, *IEEE Geoscience and Remote Sensing Letters*, 12(9), pp. 1973–1977. doi: 10.1109/LGRS.2015.2441631.

90. Zhai, W. *et al.* (2019) ‘Hyperspectral analysis of objects under shadow conditions based on field reflectance measurements’, *Applied Optics*, 58(17), pp. 4797–4805. doi: 10.1364/AO.58.004797.

91. Zhang, L. (2017) ‘System generator model-based FPGA design optimization and hardware co-simulation for Lorenz chaotic generator’, in *Intelligent Robot Systems (ACIRS), 2017 2nd Asia-Pacific Conference on*. IEEE, pp. 170–174.

LIST OF PUBLICATIONS, CONFERENCES, AND PATENTS

List of peer-reviewed journal publications

1. **D. Gyaneshwar** and R. R. Nidamanuri, "**A real-time FPGA accelerated stream processing for hyperspectral image classification**," *Geocarto International*, vol. 0, no. 0, pp. 1–18, Jan. 2020, doi: 10.1080/10106049.2020.1713231.
2. **D. Gyaneshwar** and R. R. Nidamanuri, "**A novel supervised cascaded classifier system (SC^{2}S) for robust remote sensing image classification**," *IEEE Geoscience and Remote Sensing Letters*, vol. 18, no. 3, pp. 421-425, Mar. 2021, doi: 10.1109/LGRS.2020.2980186.
3. **D. Gyaneshwar** and R. R. Nidamanuri, "**Shadow and illumination invariant classification algorithms for high-resolution remote sensing imagery**," *Pattern Recognition (Elsevier). (Under review).*
4. **D. Gyaneshwar** and R. R. Nidamanuri, "**A real-time SC^{2}S-based open-set recognition using remote sensing imagery data**," *Engineering Applications of Artificial Intelligence (Elsevier). (Under review).*

List of Conferences

1. **D. Gyaneshwar** and R. R. Nidamanuri, "**Reconfigurable computing based online image classification using hyperspectral imagery**," *IEEE GRSS Young Researchers Conclave 2020, Trivandrum, Dec 20. (Oral presentation).*

2. **D. Gyaneshwar** and R. R. Nidamanuri, "**Low-complexity reconfigurable computing based online one-class classification using high-resolution hyperspectral imagery**," *InGRASS 2021 conference. (Accepted and will be published in IEEE Xplore)*

List of Patents

1. **D. Gyaneshwar** and R. R. Nidamanuri, "**Shadow and illumination invariant image classification algorithms**," *(in progress).*