TOWARDS REDUCED-COST HYPERSPECTRAL AND MULTISPECTRAL IMAGE CLASSIFICATION

by

Giorgio L. Morales Luna

A thesis submitted in partial fulfillment
of the requirements for the degree

of

Master of Science

in

Computer Science

MONTANA STATE UNIVERSITY
Bozeman, Montana

October 2021

©COPYRIGHT

by

Giorgio L. Morales Luna

2021

All Rights Reserved

ACKNOWLEDGEMENTS

The research reported here has been supported in part by the Intel Corporation (SRA 10-18-17), the National Science Foundation (OIA-1757351), and the US Department of Agriculture (NR213A750013G021).

I would like to thank Dr. John Sheppard, my advisor and committee chair, for his support and guidance throughout this new exciting chapter of my life, and my fellow members of the Numerical Intelligent Systems Laboratory at Montana State University for their valuable comments and suggestions.

I would also like to thank the members of my thesis committee, Dr. Sean Yaw and Dr. Joseph Shaw, and the members of the Optical Technology Center at Montana State University, especially Bryan Scherrer and Riley Logan, for all the information and help during the development of this work.

Finally, I cannot begin to express my thanks to my beloved fiancee, Flor, for her constant support and inspiration during this thesis writing process, and my parents, Jorge and Sandra, for their unconditional love and for always being there for me.

TABLE OF CONTENTS

1. INTRODUCTION 1

2. BACKGROUND 8

2.1 Hyperspectral and Multispectral Imaging Systems 8
2.1.1 Hyperspectral Imaging Systems 8
2.1.2 Multispectral Imaging Systems 10
2.2 Convolutional Neural Networks 11
2.2.1 Common Layers in a CNN 12
2.2.2 Densely Connected Blocks 17
2.2.3 Separable Convolutions 17
2.2.4 Three-dimensional Convolutional Neural Networks 19
2.3 Dimensionality Reduction 20
2.3.1 Feature Extraction 20
2.3.1.1 Principal Component Analysis 21
2.3.1.2 Partial Least Squares Discriminant Analysis 22
2.3.2 Feature Selection 23

3. EXPERIMENTAL IMAGE DATASETS 25

3.1 Datasets Description 25
3.1.1 Remote Sensing HSI Datasets 25
3.1.2 EuroSAT Dataset 26
3.1.3 Kochia dataset 27
3.2 Data Pre-processing 28
3.2.1 Hyperspectral Image Classification Approach 29
3.2.1.1 Remote sensing datasets 29
3.2.1.2 Kochia Dataset 30
3.2.2 Hyperspectral Band Selection Approach 32
3.2.2.1 Remote Sensing Datasets 32
3.2.2.2 Kochia Dataset 33

4. REDUCED-COST HYPERSPECTRAL IMAGE CLASSIFICATION 34

4.1 Related Work 34
4.2 The Hyper3DNet Architecture 36
4.3 Results 37
4.3.1 Compared Methods 40
4.3.2 Results on the IP, PU, and SA datasets 41
4.3.3 Results on the EuroSAT dataset 44

TABLE OF CONTENTS – CONTINUED

4.3.4 Results on the Kochia dataset 45
4.3.5 Discussion 49
4.4 Conclusions 51

5. HYPERSPECTRAL BAND SELECTION 52

5.1 Related Work 52
5.2 Proposed Methods 56
5.2.1 Convolutional Neural Network Architecture 56
5.2.2 Inter-Band Redundancy Analysis 57
5.2.3 Band Selection Using Pre-Selected Bands 62
5.2.4 Multispectral Imager Design 66
5.2.5 Feature Extraction Framework 68
5.3 Results 69
5.3.1 Training with Pre-Selected Bands Alone 70
5.3.2 Greedy Spectral Selection 70
5.3.3 Comparative Results 77
5.3.4 Discussion 84
5.4 Conclusion 90

6. CONCLUSION 92

REFERENCES CITED 97

APPENDICES 108

APPENDIX A: GSS on Reduced Datasets 109
A.1 Kochia Dataset 110
A.2 Indian Pines Dataset 115
A.3 Salinas Dataset 116
APPENDIX B: Band Selection Methods on Reduced Datasets 117
B.1 Kochia Dataset 118
B.2 Indian Pines Dataset 123
B.3 Salinas Dataset 125

LIST OF TABLES

Table Page

4.1 Comparison of number of trainable parameters and MFLOPS of the different networks trained on the IP, PU, SA, and EuroSAT datasets. 42

4.2 Comparison of number of trainable parameters and MFLOPS of the different networks trained on the Kochia dataset. 42

4.3 Metrics comparison on the IP dataset. 43

4.4 Metrics comparison on the PU dataset. 43

4.5 Metrics comparison on the SA dataset. 43

4.6 Metrics comparison on the EuroSAT dataset. 44

4.7 Metrics comparison on the Kochia dataset. 46

5.1 Hyper3DNet-Lite architecture for the Kochia dataset. 57

5.2 Performance with and without IBRA preselection ($\theta = 10$). 71

5.3 Greedy Spectral Selection on the Kochia dataset ($k = 6$). 72

5.4 Greedy Spectral Selection on the Kochia dataset ($k = 8$). 73

5.5 Greedy Spectral Selection on the Kochia dataset ($k = 10$). 74

5.6 Greedy Spectral Selection on the Indian Pines dataset. 75

5.7 Greedy Spectral Selection on the Salinas dataset. 75

5.8 Feature selection method comparison — Kochia ($k = 6$). 78

5.9 Feature selection method comparison — Kochia ($k = 8$). 79

5.10 Feature selection method comparison — Kochia ($k = 10$). 79

5.11 Feature selection method comparison — IP ($k = 5$). 80

5.12 Feature selection method comparison — SA ($k = 5$). 81

5.13 Hypothesis tests — Kochia dataset. p-values obtained comparing the classification performance of IBRA-GSS to that of the other methods. 86

LIST OF TABLES – CONTINUED

Table Page

5.14 Hypothesis tests — IP and SA datasets. p-values obtained comparing the classification performance of IBRA-GSS to that of the other methods. 86

5.15 Dispersion analysis of the F1 scores obtained using all the compared dimensionality reduction methods on the full Kochia dataset. 87

A.1 GSS on 75% of the Kochia dataset ($k = 6$). 110

A.2 GSS on 75% of the Kochia dataset ($k = 8$). 110

A.3 GSS on 75% of the Kochia dataset ($k = 10$). 111

A.4 GSS on 50% of the Kochia dataset ($k = 6$). 111

A.5 GSS on 50% of the Kochia dataset ($k = 8$). 112

A.6 GSS on 50% of the Kochia dataset ($k = 10$). 112

A.7 GSS on 25% of the Kochia dataset ($k = 6$). 113

A.8 GSS on 25% of the Kochia dataset ($k = 8$). 113

A.9 GSS on 25% of the Kochia dataset ($k = 10$). 114

A.10 GSS on 75% of the Indian Pines dataset. 115

A.11 GSS on 50% of the Indian Pines dataset. 115

A.12 GSS on 25% of the Indian Pines dataset. 115

A.13 GSS on 75% of the Salinas dataset. 116

A.14 GSS on 50% of the Salinas dataset. 116

A.15 GSS on 25% of the Salinas dataset. 116

B.1 Feature selection method comparison on 75% of the Kochia dataset ($k = 6$). 118

B.2 Feature selection method comparison on 75% of the Kochia dataset ($k = 8$). 118

B.3 Feature selection method comparison on 75% of the Kochia dataset ($k = 10$). 119

LIST OF TABLES – CONTINUED

Table Page

B.4 Feature selection method comparison on 50% of the Kochia dataset ($k = 6$).....119

B.5 Feature selection method comparison on 50% of the Kochia dataset ($k = 8$).....120

B.6 Feature selection method comparison on 50% of the Kochia dataset ($k = 10$).....120

B.7 Feature selection method comparison on 25% of the Kochia dataset ($k = 6$).....121

B.8 Feature selection method comparison on 25% of the Kochia dataset ($k = 8$).....121

B.9 Feature selection method comparison on 25% of the Kochia dataset ($k = 10$).....122

B.10 Feature selection method comparison on 75% of the IP dataset. 123

B.11 Feature selection method comparison on 50% of the IP dataset. 123

B.12 Feature selection method comparison on 25% of the IP dataset. 124

B.13 Feature selection method comparison on 75% of the SA dataset.....125

B.14 Feature selection method comparison on 50% of the SA dataset.....125

B.15 Feature selection method comparison on 25% of the SA dataset.....126

LIST OF FIGURES

Figure Page

1.1 A Resonon Pika L hyperspectral imager mounted on a DJI Matrice 600 Pro while the drone is in flight taking images of a sugar beet field [91]. 2

2.1 Example of a hyperspectral data cube. The full spectrum captured by the imager is available at each pixel [58]. 9

2.2 A 2-D convolution example. **(a)** Original image. **(b)** Vertical edges extracted. (c) Horizontal edges extracted. 12

2.3 Application process of a convolutional layer with two convolutional filters over an input with three channels. 13

2.4 Visualization of features in a fully trained CNN [123]. Feature maps obtained by the **(a)** second layer, **(b)** third layer, **(c)** fourth layer, and **(d)** fifth layer. 14

2.5 Graphical examples of **(a)** 4-layer densely connected block and **(b)** residual block. 18

2.6 3-D convolution operation. 19

3.1 Color-corrected samples from the EuroSat dataset. 27

3.2 A Resonon Pika L hyperspectral imager taking data in a controlled greenhouse setting. 28

3.3 Kochia leaves at **(a)** 8 weeks and **(b)** 2 weeks. **(c)** An example of the extraction of 25×25 pixel patches. 31

4.1 Hyper3DNet architecture with a $25 \times 25 \times 30 \times 1$ HSI input. The network includes a 3-D feature extractor (left) and a 2-D spatial encoder (right). 37

4.2 Mean spectral reflectance curve of all the pixels of the Kochia dataset. 39

4.3 Box plot of F1 scores on the **(a)** Indian Pines, **(b)** Pavia University, and **(c)** Salinas datasets. Numbers in the abscissa corresponding to 1) Hyper3DNet; 2) HybridSN; 3) ResNet50; 4) SpectrumNet. 44

LIST OF FIGURES – CONTINUED

Figure Page

4.4 Classification maps of the Indian Pines dataset. **(a)** RGB image. **(b)** Ground-truth labels. **(c)** SpectrumNet. **(d)** ResNet50. **(e)** HybridSN. **(f)** Hyper3DNet. Zoomed images show some misclassified pixels in yellow. 45

4.5 Classification maps of the Pavia University dataset. **(a)** RGB image. **(b)** Ground-truth labels. **(c)** SpectrumNet. **(d)** ResNet50. **(e)** HybridSN. **(f)** Hyper3DNet. Zoomed images show some misclassified pixels in yellow. 46

4.6 Classification maps of the Salinas dataset. **(a)** RGB image. **(b)** Ground-truth labels. **(c)** SpectrumNet. **(d)** ResNet50. **(e)** HybridSN. **(f)** Hyper3DNet. Zoomed images show some misclassified pixels in yellow. 47

4.7 Box plot of F1 scores of different methods on the EuroSAT dataset. 48

4.8 Box plot of F1 scores of different methods on the Kochia dataset. 48

4.9 Confusion matrices of different networks on the Kochia dataset. **(a)** Hyper3DNet. **(b)** HybridSN. **(c)** SpectrumNet. **(d)** KochiaFC. 49

5.1 Example of the calculation of distances d_{left} and d_{right} for bands $\mathbf{x}_4$ and $\mathbf{x}_5$ used in the IBRA process. Note that the value of $\text{VIF}(\mathbf{x}_4, \mathbf{x}_5)$ is calculated when analyzing the variable $\mathbf{x}_4$ so that this value is simply reused in the next iteration. 60

5.2 Spectral response and Spectral index vs. distance d plots for the Kochia dataset using different VIF thresholds. **(a)** $\theta = 8$ **(b)** $\theta = 10$. **(c)** $\theta = 12$. Local minima in the three graphs are indicated with an '$\times$'. 61

5.3 Demonstration of the Greedy Spectral Selection (GSS) process for the Kochia dataset using a VIF threshold of 10 and $k = 6$. 65

5.4 Example of a multispectral filter simulation. 68

LIST OF FIGURES – CONTINUED

Figure Page

5.5 Box plot of F1 scores of different band selection methods on the Kochia dataset with $k = 6$. **(a)** Original band selection. **(b)** Multispectral filter simulation. 82

5.6 Box plot of F1 scores of different band selection methods on the Kochia dataset with $k = 8$. **(a)** Original band selection. **(b)** Multispectral filter simulation. 82

5.7 Box plot of F1 scores of different band selection methods on the Kochia dataset with $k = 10$. **(a)** Original band selection. **(b)** Multispectral filter simulation. 83

5.8 Box plot of F1 scores of different band selection methods on the Indian Pines dataset with $k = 5$. **(a)** Original band selection. **(b)** Multispectral filter simulation. 84

5.9 Box plot of F1 scores of different band selection methods on the Salinas dataset with $k = 5$. **(a)** Original band selection. **(b)** Multispectral filter simulation. 84

5.10 Average F1 scores obtained by each dimensionality reduction method for the Kochia dataset across different dataset sizes. **(a)** Original band selection using $k = 6$. **(b)** Multispectral filter simulation using $k = 6$. **(c)** Original band selection using $k = 8$. **(d)** Multispectral filter simulation using $k = 8$. **(e)** Original band selection using $k = 10$. **(f)** Multispectral filter simulation using $k = 10$. 85

LIST OF ALGORITHMS

Algorithm Page

2.1 Partial Least Squares Discriminant Analysis 23

5.2 Calculating the interband redundancy 59

5.3 Greedy spectral selection 64

ABSTRACT

In recent years, Hyperspectral Imaging systems (HSI) have become a powerful source for reliable data in applications such as remote sensing, agriculture, and biomedicine. However, the abundant spectral and spatial information of hyperspectral images makes them highly complex, which leads to the need for specialized Machine Learning algorithms to process and classify them. In that sense, the contribution of this thesis is multi-folded. We present a low-cost convolutional neural network designed for hyperspectral image classification called Hyper3DNet. Its architecture consists of two parts: a series of densely connected 3-D convolutions used as a feature extractor, and a series of 2-D separable convolutions used as a spatial encoder. We show that this design involves fewer trainable parameters compared to other approaches, yet without detriment to its performance. Furthermore, having observed that hyperspectral images benefit from methods to reduce the number of spectral bands while retaining the most useful information for a specific application, we present two novel hyperspectral dimensionality reduction techniques. First, we propose a filter-based method called Inter-Band Redundancy Analysis (IBRA) based on a collinearity analysis between a band and its neighbors. This analysis helps to remove redundant bands and dramatically reduces the search space. Second, we apply a wrapper-based approach called Greedy Spectral Selection (GSS) to the results of IBRA to select bands based on their information entropy values and train a compact Convolutional Neural Network to evaluate the performance of the current selection. We also propose a feature extraction framework that consists of two main steps: first, it reduces the total number of bands using IBRA; then, it can use any feature extraction method to obtain the desired number of feature channels. Finally, we use the original hyperspectral data cube to simulate the process of using actual filters in a multispectral imager. Experimental results show that our proposed Hyper3DNet architecture in conjunction with our dimensionality reduction techniques yields better classification results than the compared methods, producing more suitable results for a multispectral sensor design.

CHAPTER ONE

INTRODUCTION

Hyperspectral imaging systems are becoming widely used due to their increasing accessibility and their ability to provide detailed spectral responses based on hundreds of spectral bands [60]. Because of this fine level of spectral detail, hyperspectral imaging systems enable us to better characterize or differentiate objects based on subtle spectral discrepancies that could not have been noticed using conventional digital cameras [61]. However, the resulting hyperspectral images (HSIs) come at the cost of increased storage requirements, increased computational time to process, and highly redundant data [101].

Hyperspectral imaging is a remote, non-destructive spectroscopic technique, which makes it appealing for many diverse applications, ranging from lab-based analysis of food quality and safety [117] to space-based contributions to archaeology [20]. For instance, an important application area is produce quality monitoring. Conventional methods, such as mass spectrometry, can be used to monitor the quality of produce but their destructive nature prevents them from being used for continuous classification or sorting tasks [24]. However, it has been shown that observation of spectral changes throughout the ripening process of fruits is possible [81, 87]; therefore, new efforts have been put on using hyperspectral imaging systems for automatic quality assurance [59, 94].

Until a few years ago, most of the use of hyperspectral imaging systems was confined to laboratories due to their large size, difficulty to operate, and affordability issues [92]. Nevertheless, recently, a wide range of small and low-cost hyperspectral sensors have been developed and are available for commercial use [2, 60]. These hyperspectral imagers can mounted on drones and monitor large portions of land in short periods of time, which makes

Figure 1.1: A Resonon Pika L hyperspectral imager mounted on a DJI Matrice 600 Pro while the drone is in flight taking images of a sugar beet field [91].

them suitable for precision agriculture or forestry applications [2]. For example, the Resonon Pika L hyperspectral imager is a small instrument that can be mounted on a small drone, as shown in Fig 1.1. Scherrer *et al.* [91] used this imager to create the Kochia dataset analyzed in this thesis with the aim of learning to differentiate between three different classes of herbicide-resistance of the weed Kochia (*Bassia scoparia*).

Although HSIs have been used for decades [22, 106], their use has presented many challenges [54]. In general, the large number of spectral bands and the fine spectral resolution cause redundancy of the data [98], which makes it difficult to train HSI classifiers efficiently and increases the chances of overfitting [50, 89]. This is a by-product of a phenomenon that is known as the "curse of dimensionality" [5] or "Hughes phenomenon" [39]. However, it is worth noting that redundancy of information may be helpful in some applications such as information recovery [56]. Furthermore, dealing with HSIs becomes a challenge due to the increased volume of data; therefore, we should prioritize computational efficiency when storing and processing hyperspectral images [78].

With these issues in mind, in this thesis, we focus on developing a method to

classify hyperspectral images both accurately and efficiently. In particular, we present a convolutional neural network (CNN) model called Hyper3DNet with two sections: a feature extractor and a spatial encoder. The feature extractor consists of a series of densely connected three-dimensional (3-D) convolutions and is used to extract deep spectral features while the spatial resolution remains the same. Afterward, the spatial encoder is composed of a series of two-dimensional (2-D) separable convolutions that reduce the spatial resolution to a one-dimensional (1-D) vector, followed by a final fully connected layer. We hypothesize that this architecture will lead us to train more efficient HSI classification models than the compared methods, given a set of different HSI datasets. Here, we talk about efficiency in the sense of the performance metrics a model achieves and the number of parameters and computational operations it requires.

As an alternative to hyperspectral imaging systems, multispectral imaging systems capture several distinct spectral bands. These systems are useful for capturing information in known areas of the spectrum. For example, if an application requires the detection of vegetation, a multispectral imager may only be required to capture the reflectance at 680 nm (red) and 800 nm (near-infrared), two commonly used spectral channels to capture the chlorophyll content [120]. In contrast, a hyperspectral imager would capture data from hundreds of bands surrounding the wavelengths of interest. In such situations, the complexity introduced by the HSI system may be unnecessary if similar detection results can be achieved with fewer spectral channels.

Though powerful, the spectrally dense images captured by hyperspectral imagers involve computational limitations when storing and processing these types of images. For example, the Hyperspectral Infrared Imager (HyspIRI) launched by NASA's Jet Propulsion Laboratory (JPL) has a continuous data rate of 65 MB/s onn average, which produces a data volume of 5.2 TB/day [35]; therefore, the inherent computational cost makes fast on-board processing infeasible. Thus, the ability to determine the most important

wavelengths in a hyperspectral image would greatly simplify the data capture and processing requirements. Namely, it would enable using multispectral imagers in place of hyperspectral imagers, greatly reducing complexity and cost. Unfortunately, in many applications, relevant wavelengths are not known *a priori* and selecting salient wavelengths from an HSI system is not a trivial task.

To allow for this simplification, in this thesis, we present a dimensionality reduction technique called Inter-Band Redundancy Analysis (IBRA) that removes redundant spectral bands, thus simplifying spectrally-dense image data captured by a hyperspectral imager. IBRA is based on a recursive collinearity analysis between each spectral band and its neighbors, which allows for an approximation of the minimum number of bands we need to move away from a band to find spectral bands with sufficiently distinct information. Having calculated this distance metric for each spectral band, its distribution across the spectrum is used to cluster the spectrum into sets of similar and contiguous spectral bands. This allows us to identify a reduced set of independent bands that act as the centroids of their corresponding clusters in the spectrum.

Along with simplifying the hyperspectral data cubes, our IBRA method can be used as part of a novel two-step hybrid feature selection process. The first step applies IBRA to find a reduced set of independent representative bands, then it applies our proposed wrapper-based method called Greedy Spectral Selection (GSS) to select a user-defined number of spectral channels. GSS ranks the candidate bands according to their information entropy to obtain an initial selection of bands, which is used to train a classifier that evaluates the effectiveness of the given selection of bands based on the achieved classification performance. Then, we remove from the selection the band that shows the most severe indication of multicollinearity and repeat the process by considering the next band with the highest information entropy, verifying if the classification performance improves. This band selection process is implemented as a part of achieving our goal of designing low-cost, high accuracy

MSI systems for various applications. That is, given a classification problem, our method allows us to select a user-defined number of salient wavelengths from an HSI dataset which can then be used for MSI while maintaining or improving the accuracy of the original HSI-based system.

Furthermore, we explore the applicability of IBRA in combination with other dimensionality reduction methods besides the various feature selection methods. Specifically, we propose a feature extraction framework whose first step aims to remove the spectral redundancy by using IBRA. The second step consists of applying any desired dimensionality reduction method on the IBRA-selected set of spectral bands to obtain a new representation with the desired number of dimensions or feature channels. In this work, we show experimental results for this framework considering both a supervised and an unsupervised feature extraction method for the second step. Our results show that our feature extraction framework yields better or competitive results in comparison to several alternative feature selection methods in the context of hyperspectral image classification.

This work makes a number of contributions regarding accurate and efficient computational models for hyperspectral image analysis. Our specific contributions are summarized as follows:

- We present a low-cost convolutional neural network architecture designed for hyperspectral image classification. We show that this design involves fewer trainable parameters compared to other approaches, yet without detriment to its performance.

- We present a filter-based selection method called Inter-Band Redundancy Analysis (IBRA) that works as a pre-selection method to remove redundant bands and reduce the search space dramatically.

- We present a two-step band selection method that first applies IBRA to obtain a reduced set of candidate bands and then selects the desired number of bands using a

wrapper-based method called Greedy Spectral Selection (GSS).

- We show that IBRA can be used as part of a more general two-step feature extraction framework where any dimensionality reduction method can be applied following IBRA to obtain the desired number of feature channels.

- Since one of the objectives of this work is to aid in the design of multispectral imaging systems based on the wavelengths recommended by a band selection method, we also present an extensive set of experiments that use the original hyperspectral data cube to enable simulating the process of using actual filters in a multispectral imager.

The results concerning our proposed reduced-cost CNN architecture for hyperspectral image classification have been published in the following article:

- G. Morales, J. Sheppard, B. Scherrer, and J. Shaw. Reduced-cost hyperspectral convolutional neural networks. *Journal of Applied Remote Sensing*, 14(3):036519, 2020

Parts of the results concerning our proposed hyperspectral dimensionality reduction techniques have been published in the following conference article and its extended journal version:

- G. Morales, J. Sheppard, R. Logan, and J. Shaw. Hyperspectral band selection for multispectral image classification with convolutional networks. In *IEEE International Joint Conference on Neural Networks (IJCNN)*, June 2021

- G. Morales, J. W. Sheppard, R. D. Logan, and J. A. Shaw. Hyperspectral dimensionality reduction based on inter-band redundancy analysis and greedy spectral selection. *Remote Sensing*, 13(18), 2021

The remainder of this thesis is organized as follows. In Chapter 2, we cover necessary information to make the reader familiar with the various topics discussed in this thesis,

including hyperspectral and multispectral imaging systems, convolutional neural networks, and dimensionality reduction.

Chapter 3 describes the datasets that were used to design and validate the methods proposed in this thesis. We also provide details on the pre-processing techniques applied to these datasets depending on if we perform hyperspectral image classification using the full spectrum or just a reduced set of features obtained by a dimensionality reduction process.

In Chapter 4, we introduce our 3-D/2-D convolutional neural network architecture for hyperspectral image classification called Hyper3DNet. We present experimental results testing our architecture on four public remote sensing datasets and one in-greenhouse hyperspectral dataset, achieving comparable state-of-the-art results.

We focus on our novel dimensionality reduction methods for hyperspectral images in the context of image classification in Chapter 5. In particular, we provide details on our IBRA algorithm and explain its use as a filter-based selection method that helps to remove spectral redundancy. Then, we define our wrapper-based selection method, GSS, and show how it is applied to the results of IBRA to form our proposed hyperspectral band selection method. Furthermore, we present a feature extraction framework, which extends the use of our IBRA method to include feature extraction. We present classification results obtained from our methods and compare them to other dimensionality reduction methods. Finally, we briefly conclude and discuss the future work in Chapter 6.

CHAPTER TWO

BACKGROUND

In this chapter, we introduce the main concepts and methods that form the foundation for the work presented in later chapters.

2.1 Hyperspectral and Multispectral Imaging Systems

A conventional digital camera can be defined as a visible imaging sensor; that is, an imager that captures the reflected light at wavelengths of light ranging from 400 to 700 nm, which is approximately the same spectrum that the human eye perceives [99]. Generally, this type of camera utilizes three bands of wavelengths that capture the light in the blue range (440 to 485 nm), the green range (500 to 565 nm), and the red range (625 to 740 nm). In contrast, hyperspectral and multispectral imaging systems differ from visible imaging systems in that they collect information from a wider range of the electromagnetic spectrum and not only from the range of visible wavelengths.

2.1.1 Hyperspectral Imaging Systems

Hyperspectral images (HSIs) originated from the combination of spectroscopy and digital imaging; consequently, they consist of hundreds of narrow contiguous spectral channels that measure reflectance over a portion of the visible and near-infrared (NIR) spectrum, typically ranging from 400 to 2500 nm [63]. The high spectral resolution allows us to identify some materials (or their properties) by their absorption-band characteristics [93], while the high spatial resolution enables us to capture details such as shapes or textures that complement the spectral information and achieve finer results [53, 76].

HSIs are three-dimensional images with two spatial dimensions and one spectral dimension. As such, a hyperspectral image can be thought of as a data cube that consists of

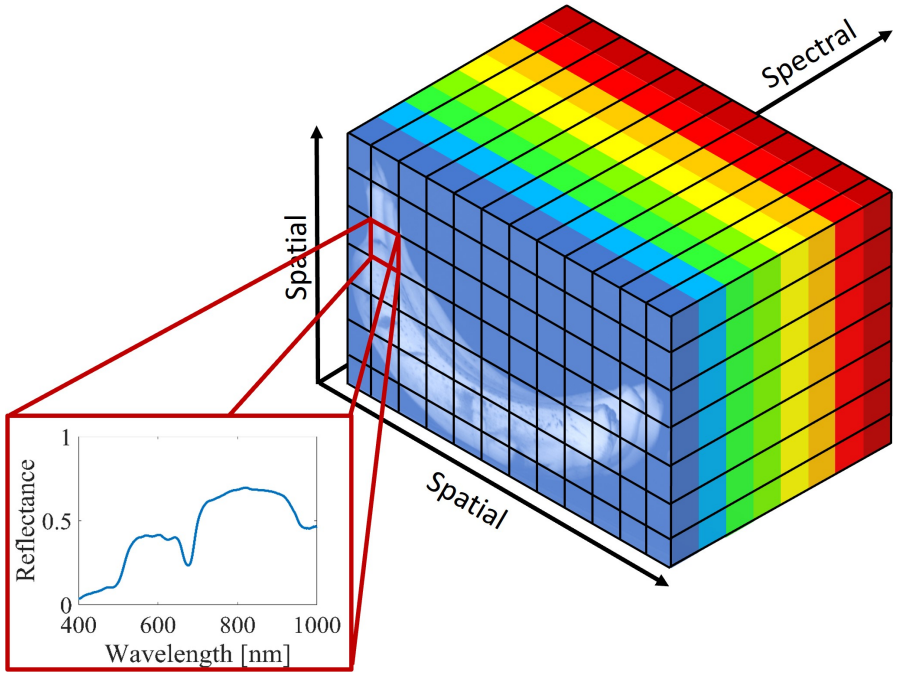

Figure 2.1: Example of a hyperspectral data cube. The full spectrum captured by the imager is available at each pixel [58].

hundreds of two-dimensional images that represent the same spatial objects using contiguous spectral bands. For example, if we select one pixel of the image and extract its values across the spectral dimension, we would obtain a spectral reflectance curve that tells the percentage of light that the captured object reflects across various wavelengths. An example of a spectral reflectance curve is shown in Fig. 2.1.

The main reason for using HSIs is that they encompass highly complex data that can enable us to extract information that could not have been achieved using only conventional digital cameras [2, 61]. Because of this, HSIs are used in a variety of applications, such as remote sensing [7, 78, 124, 125], agriculture [1, 74], food quality [86], and biomedicine [28, 96].

2.1.2 Multispectral Imaging Systems

Multispectral imaging systems are able to collect information from multiple spectral bands (typically, a dozen or fewer) using bandpass filters that record the reflectance centered around a certain wavelength of light. Thus, these systems encompass the visible imaging systems described at the beginning of this section; however, in general, multispectral imagers are not restricted to use only three spectral channels nor to operate within the range of visible wavelengths.

The main difference between multispectral and hyperspectral imaging systems is that the former utilize spaced spectral bands while the latter record a contiguous sequence of narrow spectral bands [26]. Moreover, multispectral imagers usually acquire wider spectral bands (e.g., 20 and 30 nm) than hyperspectral imagers (e.g., 1 and 2 nm). Therefore, the design of multispectral imaging systems depends on three customizable aspects: the number of filters, the wavelength at which each filter is centered, and the bandwidth of each filter.

One of the most important advantages of multispectral imagers over hyperspectral imagers is the ability to use a reduced number of filters so that only the most relevant information is captured; thus, avoiding capturing redundant or irrelevant data. These optical filters may be known *a priori* based on expert knowledge [62, 67, 120] or determined through a feature selection process [110, 111].

There exist two general approaches for multispectral imaging: scanning and non-scanning methods [34, 105]. Scanning systems offer good spectral and spatial resolution and are mainly used to study stationary objects due to their relative slowness of capture. For example, spatial scanning systems read one pixel at a time to obtain reflectance measurements from all the available wavelengths [31]; alternatively, spectral scanning systems use a revolving wheel to switch spectral filters so that one monochromatic image is obtained at a time [30]. On the other hand, non-scanning systems capture the full spatial and spectral information simultaneously using multiple-bandpass filters; that is, filters that

consist of a set configuration of bandpass filters [105].

2.2 Convolutional Neural Networks

A Convolutional Neural Network (CNN) [52] is a specialized type of artificial neural network designed for processing data with a grid-like topology, such as 2-D images or 1-D time series. As a consequence, CNNs have been applied to a wide range of computer vision and pattern recognition applications, becoming state-of-the-art in many of them [51].

According to Goodfellow et al. [23], a CNN is a neural network that uses a mathematical operation called *convolution* instead of general matrix multiplication in at least one of their layers. The convolution operation between an input x and a kernel or filter w along one dimension t is defined as:

$$(x * w)(t) = \int_{-\infty}^{\infty} x(a)w(t-a)da.$$

If x and w are discrete along the t axis, the convolution operation is as follows:

$$(x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a).$$

A convolution operation can also be defined for inputs and kernels that have more than one dimension. For example, a convolution operation between a 2-D image I and a 2-D kernel K is expressed as:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n). \tag{2.1}$$

Here, the output S is considered a feature map resulting from the convolution operation. Depending on the values or weights of the kernel K, the matrix S will represent different features extracted from I. For example, Fig. 2.2 shows the result of two convolutional

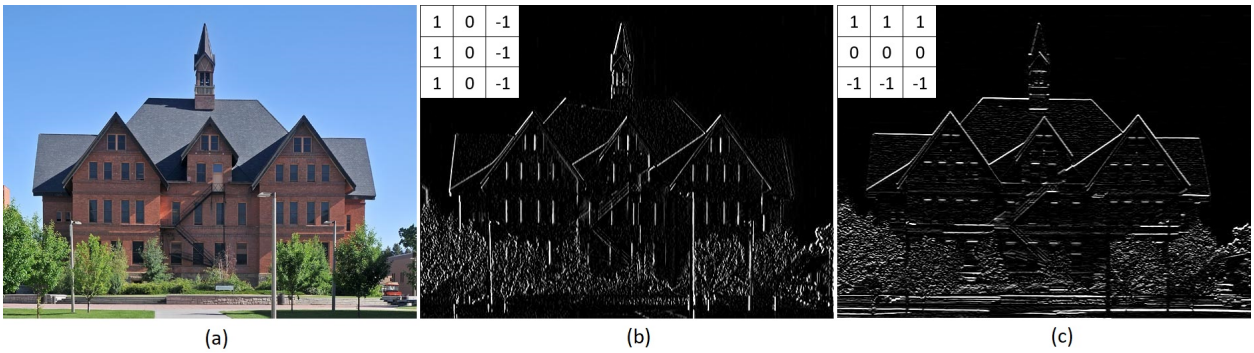

Figure 2.2: A 2-D convolution example. **(a)** Original image. **(b)** Vertical edges extracted. (c) Horizontal edges extracted.

operations using 3×3 Prewitt filters with different orientations. Both results extracted vertical and horizontal lines, respectively, from the original image by analyzing the spatial structure of 3×3 local neighborhoods. In the context of deep learning, the appropriate values of the kernel w will be found by a learning algorithm.

2.2.1 Common Layers in a CNN

Fig. 2.2 shows how very simple features can be extracted with only one convolutional filter. However, in a typical CNN architecture, a convolutional layer includes several filters in order to extract different features from the input. As an example, Fig. 2.3 shows the process of applying a convolutional layer with two convolutional filters over an input with multiple channels. Here, each convolutional filter is applied over the input separately. In each case, three copies of the convolutional filter are stacked along the third dimension so that the number of input channels matches the number of channels of the filter; that is, the new dimensions of the filter are $3 \times 3 \times 3$ pixels. Then, the convolutional filter is considered a sliding volume that moves horizontally and vertically performing element-wise multiplications and summations at each visited position as follows:

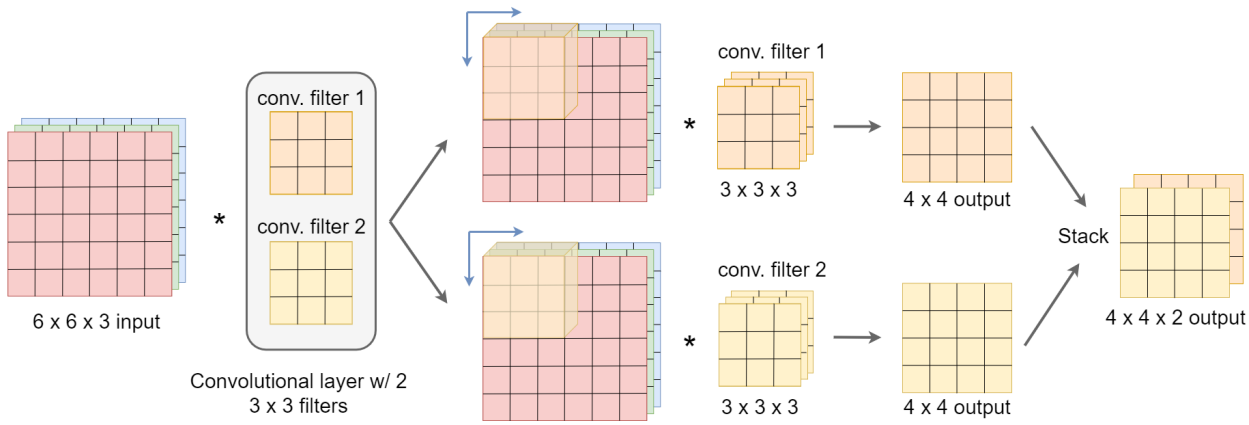

Figure 2.3: Application process of a convolutional layer with two convolutional filters over an input with three channels.

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n \sum_p I(m,n,p)K(i-m, j-n, p),$$

where p denotes the p-th channel. As a result, the output of this convolutional operation is a 2-D feature map. This process is also known as *convolution over volume*. Finally, the feature maps obtained by each convolutional filter are stacked together to represent the output of the convolutional layer.

A CNN architecture consists of multiple convolutional layers. The intuition behind this is that the deeper the network is, the more complex the features extracted and, therefore, the better the performance achieved [102]. For example, Fig. 2.4 illustrates the kinds of patterns recognized by the different layers of a CNN called AlexNet [48]. Here, each part contains four groups of nine images that strongly activated a particular filter or feature detector; on the left side, we show the feature maps that highlight the pixels that were most responsible for the strong activation and, on the right side, we show the original input images. It can be seen that the features obtained by the second layer (Fig. 2.4.a) correspond to simple shapes and textures, while those obtained by the fifth layer (Fig. 2.4.d) correspond to more complex

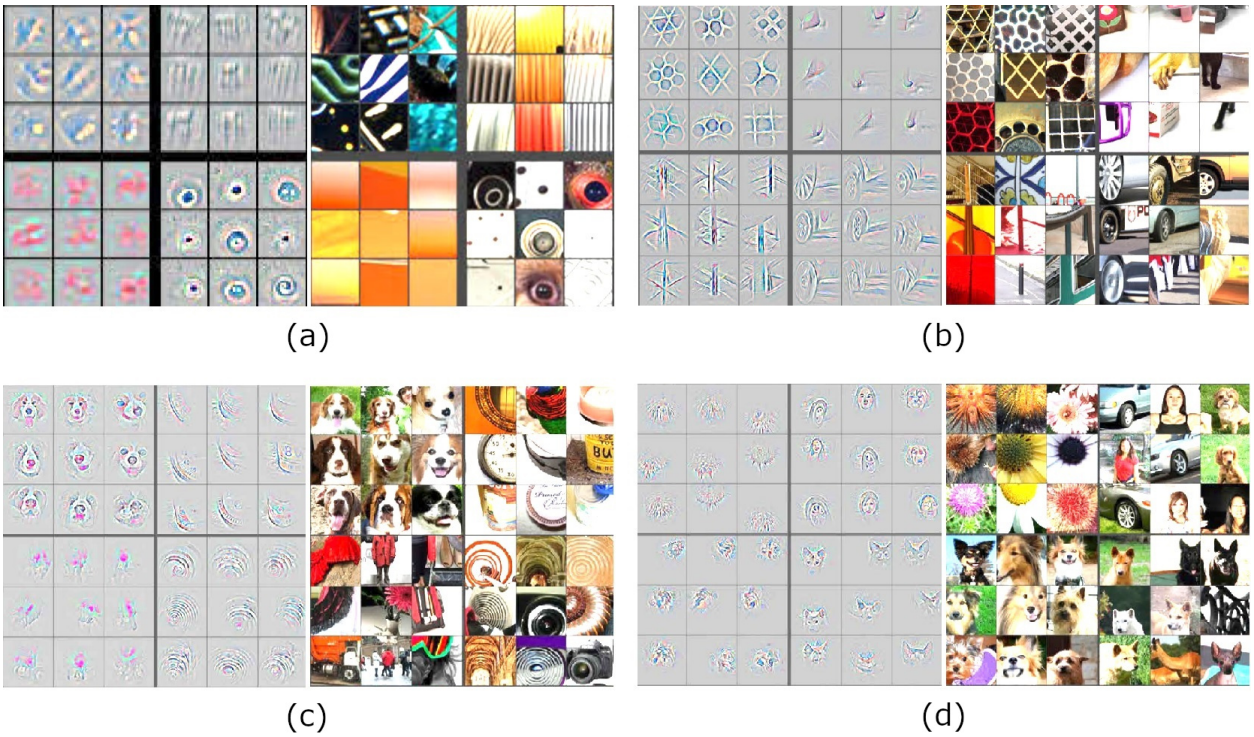

Figure 2.4: Visualization of features in a fully trained CNN [123]. Feature maps obtained by the **(a)** second layer, **(b)** third layer, **(c)** fourth layer, and **(d)** fifth layer.

objects such as flowers, faces, and dogs.

One of the main motivations for using convolutional layers is that they have sparse interactions [23]. In a layer of a traditional feedforward network, an input with m units is connected to an output with n units with a parameter matrix that describes the interaction between each input unit and each output unit; that is, this layer requires $m \times n$ parameters and computing the output has a runtime of $O(m \times n)$. On the other hand, consider a convolutional filter that requires k connections (e.g., a 3×3 pixel filter requires $k = 9$ connections) and is considerably smaller than the input ($k \ll m$). Then, the number of connections each output has is k so that the runtime required by this approach is $O(k \times n)$.

Another important motivation for using convolutional layers is parameter sharing, which refers to the fact that the same set of values of a convolutional filter is applied at every

position of the input instead of learning a separate set of parameters for every location. In other words, a convolutional layer is dramatically more efficient in terms of memory requirements as it requires only k parameters instead of $m \times n$.

Neural networks are commonly trained using the backpropagation algorithm. To do this, we need to define a loss function that we want to minimize and that is used to determine the error between the outputs of the network and the given target values. Then, the backpropagation algorithm computes the gradient of the loss function with respect to each weight of the network using the chain rule. The gradients are computed one layer at a time, iterating backward from the last layer. Here, each weight is updated proportionally to the gradient of the loss function with respect to the current weight. However, when the computed gradient approaches to zero, it prevents the weight from changing its value. This is known as the vanishing gradient problem. For that reason, convolutional layers are usually followed by non-linear activation functions such as the Rectified Linear Unit (ReLU) defined by $\texttt{ReLU}(\texttt{x}) = \max(\texttt{0}, \texttt{x})$. These functions help to prevent the vanishing gradient problem, which allows the network to learn faster and perform better [21].

Convolutional layers can also be followed by pooling layers that allow us to further modify the output of the convolutional layer. A pooling layer works as a sub-sampling technique that replaces the value of a feature map at a certain location with a summary statistic of its neighbors. For instance, a max pooling operation slides a moving window through the input in a similar way to a convolutional operation but taking the maximum value of the neighborhood while discarding the others. The average pooling operation works in a similar way but it calculates the average value of the neighborhood instead of the maximum. Decreasing the size of the feature maps using pooling layers not only reduces the total number of parameters required by the network but also makes the network more robust to small translations of the input [23]. Another sub-sampling technique consists of modifying the stride with which we slide the filter. By default, we consider a stride of 1,

meaning that we move the filters one pixel at a time. If the stride is s ($s > 1$), the filters jump s pixels at a time as we slide them around. This will produce smaller output volumes spatially.

CNNs also benefit from the inclusion of Batch Normalization (BN) layers [40]. BN is implemented during training and it standardizes the inputs to a layer using the mean and standard deviation of said inputs per mini-batch. Consider a mini-batch $B = \{z_1, \ldots, z_m\}$ of m samples. Each sample z_i is normalized to zero mean and unit variance:

$$\hat{z}_i = \frac{z_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}},$$

where μ_B and σ_B^2 are the mean and variance of B, respectively, and ϵ is a small number that avoid inconsistencies. Finally, we scale and shift $\hat{z}_i$ using the learnable parameters γ and β to obtain the output $\tilde{z}_i$:

$$\tilde{z}_i = \gamma \hat{z}_i + \beta.$$

During training, the mean and standard deviation are calculated using samples in the mini-batch. However, this is not necessary during testing. Hence, BN calculates a running mean μ and running variance σ^2 during training that are used to normalize the inputs of a layer during testing. Specifically, when using the batch B, μ and σ^2 are updated as follows:

$$\mu = \text{momentum} * \mu + (1 - \text{momentum}) * \mu_B,$$

$$\sigma^2 = \text{momentum} * \sigma^2 + (1 - \text{momentum}) * \sigma_B^2,$$

This technique accelerates the training process significantly and provides some regularization.

Finally, in the context of image classification, the network can include one or more fully-connected layers at the end. The last fully-connected layer consists of c units, where

c represents the number of classes of the given classification problem. This layer uses a softmax activation function defined for each element of a vector $\mathbf{x} = \{x_0, ..., x_c\}$ as:

$$\sigma(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

so that it produces a valid probability distribution over the outputs of the network.

2.2.2 Densely Connected Blocks

Densely connected blocks were introduced by Huang et al. [38] to ensure maximum information flow between layers in the network. This is done by connecting all layers directly with each other using skip connections or short paths while preserving the feature-map sizes. Residual Networks (ResNets) [32] also use the idea of connecting early layers to later layers to prevent network degradation problems such as gradient vanishing or exploding when the network becomes too deep. Specifically, a densely connected block reads and transforms the state from its preceding block and passes the transformed state to the next block, along with preceding information that needs to be preserved. Unlike ResNets, a densely connected block does not add its input state to the transformed state. Instead, the preservation of the preceding information of each block is ensured by the concatenation of the previous features and their transformations (Fig. 2.5).

2.2.3 Separable Convolutions

In 2014, Sifre [97] proposed the design of separable multidimensional convolutions, an operation that has become popular due to its ability to reduce model size and complexity [12, 36]. A separable convolution consists of a depthwise convolution followed by a pointwise convolution. A depthwise convolution is a spatial operation applied independently over each input channel, while the pointwise convolution is a 1×1 convolution used to project the output channels into a new channel space.

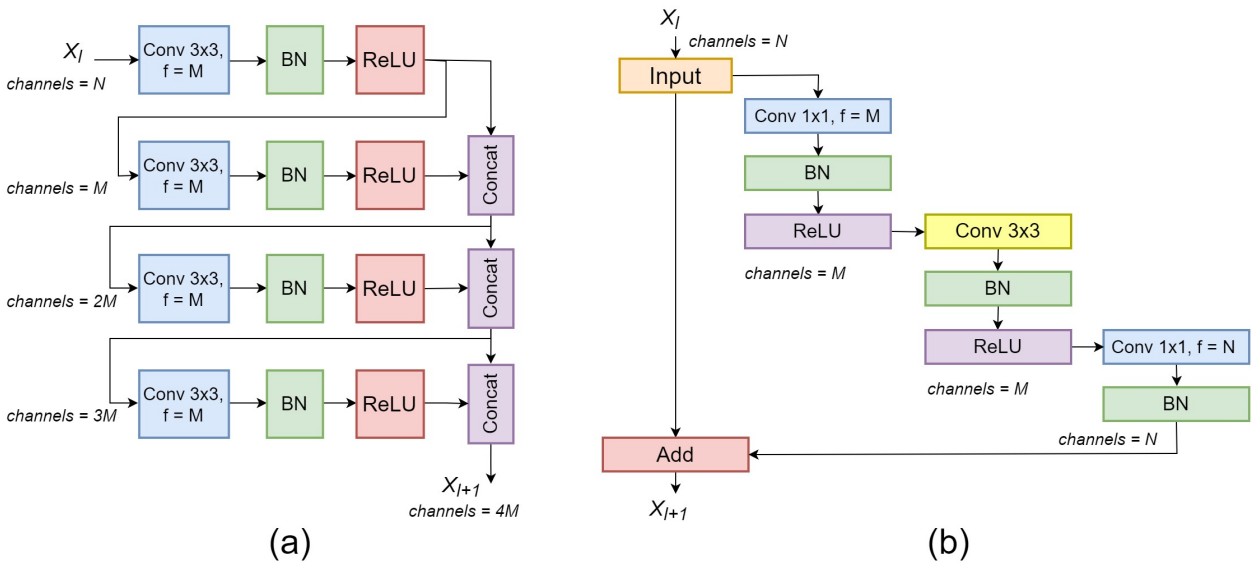

Figure 2.5: Graphical examples of **(a)** 4-layer densely connected block and **(b)** residual block.

Since the depthwise convolution is a channel-wise operation, if its input has a shape of $D_F \times D_F \times M$ (where D_F is the width and height of the input, and M is the number of input channels), then its kernel has a shape of $D_K \times D_K \times M$ (where D_K is the width and height of the filter, and M is the number of convolutional filters) and the output shape is $D_G \times D_G \times M$ (where D_G is the width and height of the output). The number of multiplications in one kernel operation is $D_K \times D_K \times 1 = D_K^2$, so the total number of multiplications for the kernel over the whole input channel is $D_G^2 D_K^2$. Then, the computational cost of a depthwise convolution is $O(M D_G^2 D_K^2)$.

After the depthwise convolution is performed, a pointwise convolution with N filters is applied. Each kernel has the shape of $1 \times 1 \times M$, so the number of multiplications per kernel is M, and the number of multiplications per channel is $D_G^2 M$. In total, the computational cost of a pointwise convolution is $O(N D_G^2 M)$. Finally, the total computational cost of a separable convolution is $O(D_G^2 M (D_K^2 + N))$.

In comparison, a regular convolution with an input shape of $D_F \times D_F \times M$, an output

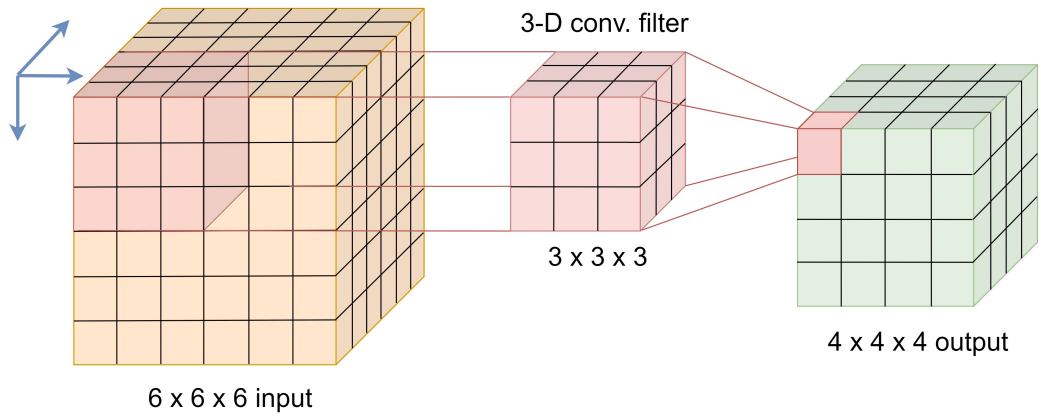

Figure 2.6: 3-D convolution operation.

shape of $D_G \times D_G \times M$, and a kernel size of $D_K \times D_K \times N$, has a computational cost of $O(NMD_G^2 D_K^2)$. Thus, compared to a regular convolution, a separable convolution reduces the computational cost by a factor of

$$\frac{D_G^2 M(D_K^2 + N)}{NMD_G^2 D_K^2} = \frac{1}{N} + \frac{1}{D_k^2}.$$

2.2.4 Three-dimensional Convolutional Neural Networks

Three-dimensional Convolutional Neural Networks (3-D CNNs) were proposed by Ji et al. [43] aiming to recognize human actions in surveillance videos. These models use 3-D convolutions to extract features from both the spatial and the temporal dimensions. In that sense, similar to Eq. 2.1, the 3-D convolution operation given a 3-D input I and a 3-D convolutional filter K is defined as:

$$S(i,j,l) = (I * K)(i,j,l) = \sum_m \sum_n \sum_p I(m,n,p)K(i-m, j-n, p-l).$$

2.3 Dimensionality Reduction

Working with hyperspectral images presents a challenge due to their high dimensionality. On the one hand, high-dimensional data requires specialized hardware to be processed efficiently. On the other, it introduces redundant or irrelevant information, which increases the size and complexity of the feature space. This leads to a phenomenon that is known as the "curse of dimensionality" [5] or the "Hughes phenomenon" [39]. This phenomenon refers to the inherent sparseness of high dimensional spaces, which implies that the amount of data needed to learn a model grows exponentially with respect to the number of dimensions (i.e., features).

In this context, hyperspectral images benefit from dimensionality reduction methods that reduce the number of spectral bands while retaining the most useful information for a specific application [5, 101]. Dimensionality reduction methods can be categorized as feature extraction methods and feature selection methods. Below, we provide a brief description of each type of method.

2.3.1 Feature Extraction

Feature extraction methods apply linear or non-linear transformations to extract composite features from the original data. The objective of such transformations is to obtain a very similar non-redundant feature space that facilitates the learning process and, in some cases, human interpretation.

Feature extraction methods can be subdivided into two classes: unsupervised and supervised. The former aims to find natural groupings from data according to some criterion, while the latter takes advantage of labeled datasets aiming to find new data representations that maximize differentiation between labels or classes. Here, we describe an unsupervised method called principal component analysis (PCA) and a supervised method called partial

least squares discriminant analysis (PLS-DA).

2.3.1.1 Principal Component Analysis Principal component analysis (PCA)is one of the most famous feature extraction methods. It can be defined as an orthogonal linear transformation that transforms the data into a new orthogonal basis, whose components are uncorrelated and ordered so that the first few retain most of the variation present in all of the original variables [46].

Consider a set of n samples $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n)$ in a d-dimensional space, such that $\mathbf{x}_i = (x_{i1}, x_{i2}, ..., x_{id})$. For the sake of conciseness, consider that $\mathbf{X}$ has zero-mean. Then, the sample variance of $\mathbf{X}$ is given by

$$v = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i)^2,$$

which measures the spread of the data along each axis.

On the other hand, we can verify how correlated two features $a, b \in [1, d]$ are based on their covariance v_{ab}:

$$v_{ab} = \frac{1}{n}\sum_{i=1}^{n} x_{ai}x_{bi}.$$

This is used to build the $d \times d$ sample covariance matrix, which is calculated as $\mathbf{X}^\top\mathbf{X}$. Then, we can use a transformation matrix $\mathbf{U}$ to obtain the diagonal matrix Λ:

$$\Lambda = \mathbf{U}^\top\mathbf{X}^\top\mathbf{X}\mathbf{U},$$

where the row vectors of $\mathbf{U}$ are called principal components and are the eigenvectors of $\mathbf{X}^\top\mathbf{X}$. Furthermore, the diagonal elements of Λ are the square root of the eigenvalues of $\mathbf{X}^\top\mathbf{X}$ and represent the variances of the transformed data along the new components.

Finally, we order the eigenvalues from largest to smallest to obtain the components in order of significance. To reduce the dimensionality of the dataset, we select the first k ($k < d$) eigenvectors with the greatest eigenvalues and ignore the rest. Then, we project the data

onto those eigenvectors using the $n \times k$ matrix $\mathbf{U}_k$ that consists of the first k eigenvectors and we obtain the transformed dataset $\mathbf{P}$ as follows:

$$\mathbf{P} = \mathbf{X}^\top \mathbf{U}_k,$$

where the $n \times k$ matrix $\mathbf{P}$ is a new data representation of $\mathbf{X}$ with reduced dimensionality whose components are decorrelated.

In summary, PCA is a dimensionality reduction technique that aims to retain as much of the variance in the dataset as possible. Machine learning models benefit from PCA in the sense that the training process is more efficient when using a reduced set of uncorrelated features.

2.3.1.2 Partial Least Squares Discriminant Analysis The partial least squares (PLS) algorithm [115] was first proposed for regression tasks and then adapted for classification using a variant known as partial least squares discriminant analysis (PLS-DA) [66]. PLS-DA consists of two main steps: PLS component construction (i.e., dimensionality reduction) and prediction model construction (i.e., discriminant analysis). In that sense, PLS-DA can be considered a supervised feature extraction technique because it achieves dimensionality reduction with full awareness of the class labels, unlike PCA.

Let $\mathbf{W} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n)$ be a zero-mean set of n training samples in a d-dimensional space such that $\mathbf{x}_i = (x_{i1}, x_{i2}, ..., x_{id})$, $\mathbf{y} = (y_1, y_2, ..., y_n)$ the set of class labels associated with $\mathbf{X}$, and l is the number of desired components. Algorithm 2.1 describes the PLS-DA process. Here, `PLS-DA(`X, y, l`)` calculates the regression coefficients $\mathbf{B}$ and the matrix transformation $\mathbf{W}$. To do this, we first estimate the weight vector $w^{(k)}$ of the k component based on the covariance between $\mathbf{X}$ and $\mathbf{y}$ (Line 3). Then, we determine the scores $\mathbf{t}^{(k)}$ (Line 5), the $\mathbf{X}$-loading factors $\mathbf{p}^{(k)}$ (Line 6) and $\mathbf{y}$-loading factors $\mathbf{q}^{(k)}$ (Line 7) sequentially. The scores $\mathbf{t}^{(k)}$ represent the projected coordinates of the data onto the new component. Line 9 is

Algorithm 2.1 Partial Least Squares Discriminant Analysis

```
function PLS-DA(X, y, l)
    X^(1) ← X
    w^(1) ← X^T y/||X^T y||
    for all k ∈ (1, l) do
        t^(k) ← X^(k) w^(k)
        p^(k) ← X^(k)^T t^(k)/||t^(k)||
        q^(k) ← y^(k)^T t^(k)/||t^(k)||
        if k < l then
            X^(k+1) ← X^(k) − t^(k) p^(k)^T
            w^(k+1) ← X^(k+1)^T y
    W ← [w^(0); w^(1); ...; w^(l)]^T
    P ← [p^(0); p^(1); ...; p^(l)]^T
    q ← [q^(0); q^(1); ...; q^(l)]^T
    B ← W(P^T W)^−1 q //Calculate regression coefficients
    return B, W
```

used to remove variability already explained from $\mathbf{X}^{(k)}$. Finally, the process is repeated for l components, after which we are able to form the projection matrix $\mathbf{W}$, whose columns consist of the vectors $\mathbf{x}^{(0)}, \mathbf{w}^{(1)}, ..., \mathbf{w}^{(l)}$.

Given that our goal is to reduce the dimensionality, we may ignore the regression coefficients B obtained by the Algorithm 2.1 and use the matrix $\mathbf{W}$ to project the data $\mathbf{X}$ onto the l orthogonal components as follows:

$$\mathbf{T} = \mathbf{X}^\top \mathbf{W},$$

where T is the projected $n \times l$ matrix with reduced dimensionality ($l < d$).

2.3.2 Feature Selection

Unlike feature extraction methods, feature selection methods aim to select the most useful subset of the features of the data without transforming them. Having determined the most relevant features, we may discard less relevant or redundant features.

There exist four types of feature selection methods: filter-based, wrapper-based, hybrid, and embedded. Filter-based methods select bands based on various statistical tests that assess the correlation between each band and the outcome variable or among other bands [112, 114]. Wrapper-based methods, on the other hand, rely on training a model to evaluate the suitability of a given set of bands. Hence, the problem becomes a search problem, which is computationally more expensive than filter-based methods [55, 111]. In order to avoid an exhaustive search, hybrid methods have been developed. They use a filter-based method as a first step to select some relevant bands. Then, a wrapper-based method is used to make the final selection [80, 101]. Finally, embedded methods train models that internally select the most relevant features during the training phase [104, 107].

CHAPTER THREE

EXPERIMENTAL IMAGE DATASETS

In this chapter, we describe the datasets we will use in Ch. 4 to design and validate efficient convolutional neural networks for hyperspectral image classification. Some of these datasets will be used in Ch. 5 to validate our method for hyperspectral band selection. We will also describe the pre-processing approaches we apply to the data before passing them to a classifier depending on if we use the full spectrum or just a reduced subset of spectral bands.

3.1 Datasets Description

In our experiments, we used three well-known remote sensing HSI datasets: Indian Pines (IP) [4], Pavia University (PU) [13], and Salinas (SA) [25]. We also experimented with the EuroSAT dataset (EU) [33] in spite of the fact that it is not a hyperspectral but a multispectral dataset, so that we validate the usefulness of the network architecture we propose in Ch. 4 for images with just a few spectral channels. Furthermore, we used an in-greenhouse controlled HSI dataset of Kochia leaves in order to classify three different herbicide-resistance levels.

3.1.1 Remote Sensing HSI Datasets

The Indian Pines dataset [4] is an aerial 145×145 pixel image of the Indian Pines site in Northwestern Indiana. It was acquired using the Airborne Visible / Infrared Imaging Spectrometer (AVIRIS) sensor [82] and it originally had 224 spectral bands in the wavelength range 400-2500 nm, resulting in a spectral resolution of approximately 9.5 nm. The number of bands was first reduced to 220 after removing four damaged bands without information (i.e., all of their pixels were zero value) [4], and then to 200 after removing 24 noisy bands

and bands covering the region of water absorption corresponding to the wavelengths (1333–1373 nm), (1789–1893 nm), and (2499 nm) that correspond to the band indices (104–108), (150–163) and 220, as recommended by Tadjudin and Landgrebe [103]. The data are divided into 16 classes containing agriculture, forest, and other natural perennial vegetation.

Similarly, the Salinas dataset [25] is a 512×217 pixel aerial image of Salinas Valley, California, that was gathered in 1998 using the same AVIRIS sensor as Indian Pines. Water absorption bands in the same wavelength ranges as Indian Pines were removed; however, in this case, they correspond to the following band indices: $(108 - 112)$, $(154 - 167)$ and 224. Thus, only 204 spectral bands are used. It is divided into 16 classes containing vegetables, vineyard fields, and bare soil.

On the other hand, the Pavia University dataset is a 610×610 pixel aerial image of Pavia, northern Italy, captured with the Reflective Optics System Imaging Spectrometer (ROSIS) sensor [49]. Unlike the other two datasets, Pavia University is an urban dataset that has 103 spectral bands and is divided into nine classes.

3.1.2 EuroSAT Dataset

The EuroSAT dataset [33] is based on Sentinel-2 satellite images covering 13 spectral bands. It consists of 27,000 64×64 images classified in 10 classes (see Fig. 3.1). Three of the spectral bands (i.e. aerosol, water vapor, and cirrus) do not present relevant information for our problem, so we discard them. This assumption is also supported by the fact that their spatial resolution, 60 m, is inconsistent with that of the rest of the bands: 10 and 20 m. In addition to this, Helber et al. [33] compared the contribution of each spectral band based on the classification accuracy achieved after training a ResNet50 network with each single band. Based on the results of that comparison, we decided to not use the B08A band (red edge 4), as it presents the lowest contribution. Thus, for this classification task, we use only nine out of the 13 spectral bands this satellite acquires.

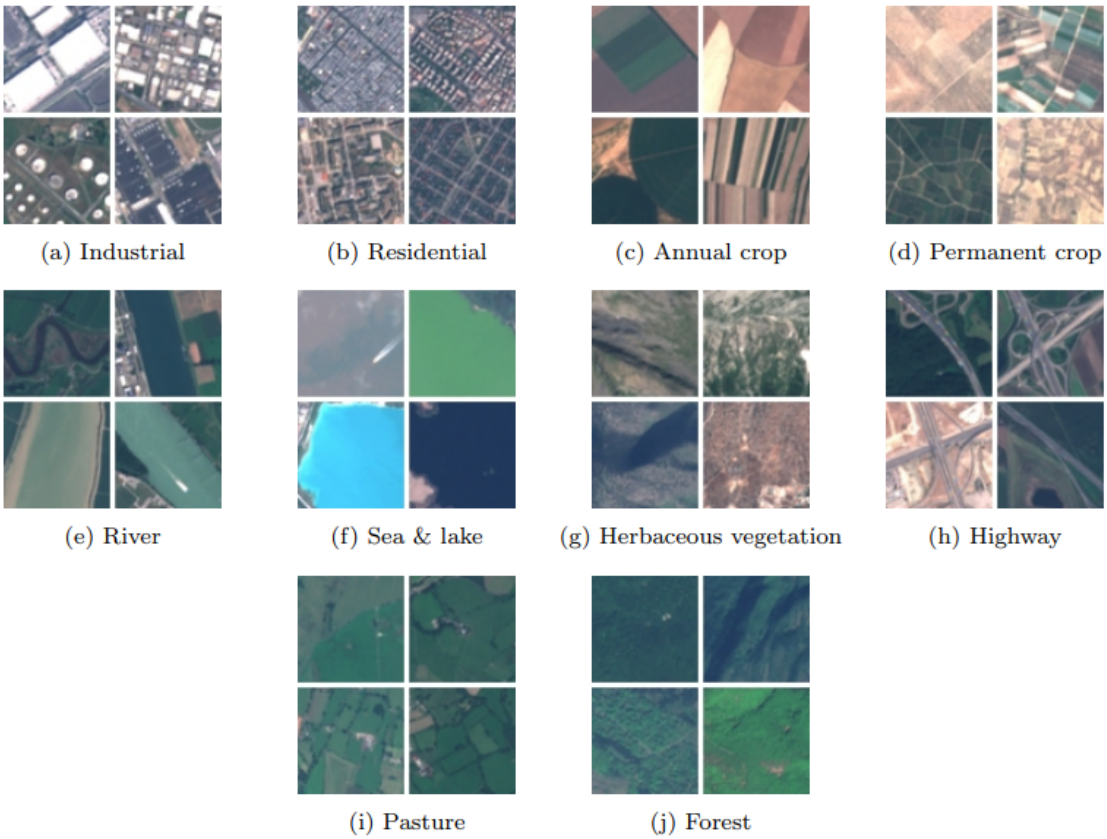

Figure 3.1: Color-corrected samples from the EuroSat dataset.

3.1.3 Kochia dataset

The last dataset is an in-greenhouse controlled hyperspectral image dataset called "Kochia". It consists of images of the weed kochia (*Bassia scoparia*) that were collected and analyzed by Scherrer *et al.* [91] with the aim of learning to differentiate between three different classes of herbicide-resistance of this weed: 1) herbicide-susceptible, 2) glyphosate-resistant, and 3) dicamba-resistant, where glyphosate and dicamba are two components commonly found in commercial herbicides. It is important to note that the difference between these three classes is imperceptible using standard color digital cameras that collect data for three bands in the visible spectrum of light (red, green, and blue), which justifies the need of using hyperspectral or multispectral imaging systems.

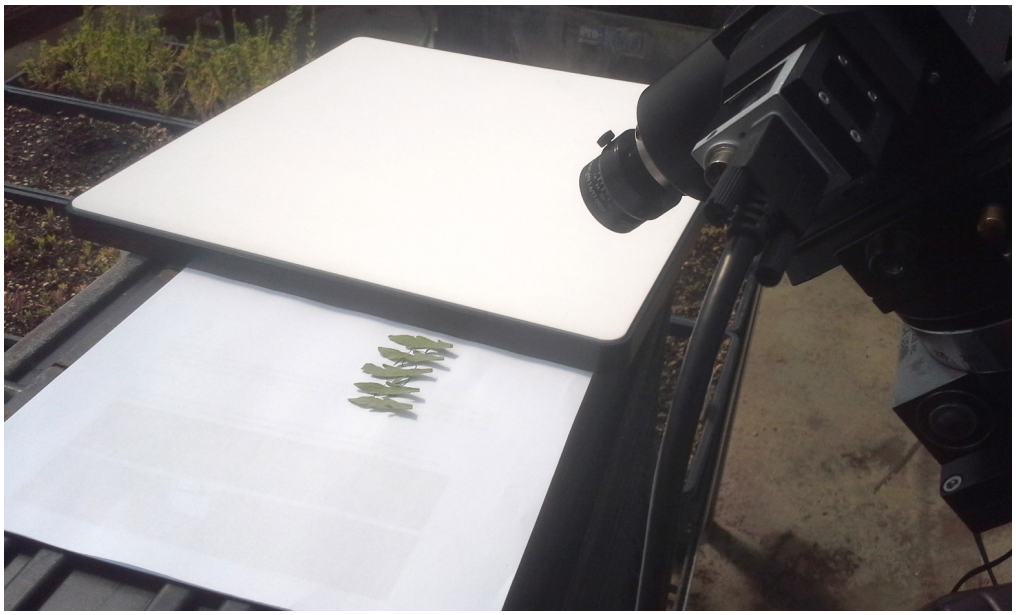

Figure 3.2: A Resonon Pika L hyperspectral imager taking data in a controlled greenhouse setting.

The images were captured using a Resonon Pika L hyperspectral imager (Fig. 3.2) with 300 spectral channels across a spectral range of 387.12 nm to 1,023.50 nm, resulting in a spectral resolution of approximately 2.12 nm. The kochia samples were illuminated using diffuse sunlight in a greenhouse setting. A total of 76 hyperspectral images of kochia with varying ages and spatial resolutions were captured at the Montana State University Southern Agricultural Research Center (SARC). Each image contains three kochia leaves of the same herbicide-resistance class with a height of 900 pixels and width ranging from 700–1,200 pixels.

3.2 Data Pre-processing

The pre-processing step differs depending on the type of task performed: hyperspectral image classification or hyperspectral band selection. That is, the pre-processing step used in Ch. 4 mainly consists of transforming the full spectrum into a new basis to reduce its

dimensionality before passing it to a classifier; whereas, in Ch. 5, it consists of reshaping the data before passing it to a feature selection algorithm that identifies a reduced set of salient spectral bands while discarding the rest. Below, we describe the pre-processing steps needed for both approaches.

3.2.1 Hyperspectral Image Classification Approach

Here, we describe pre-processing techniques that aim to exploit the information provided by the entire spectrum. As we discussed in Sec. 2.1, using all the spectral channels of hyperspectral images represents some limitations given that advanced computational resources to process the data (e.g. GPU clusters) would be needed. Therefore, we apply a dimensionality reduction technique before passing the data to a classifier in order to reduce the number of feature channels while preserving most of the original information.

3.2.1.1 Remote sensing datasets In preparing the datasets for analysis, first, we apply principal component analysis (PCA) on the HSI data cubes in order to reduce the number of spectral bands and remove spectral redundancy (see Sec. 2.3.1.1). In order to preserve spatial locality, we apply PCA at a pixel level. To do this, the original data cube with dimensions $H \times W \times Nb$ (where H is the height, W is the width, and Nb is the number of spectral bands), is reshaped to a $HW \times Nb$ matrix. In this reshaped matrix, each row is a vector with Nb values that represents a pixel of the original data cube. Thus, we apply PCA to reduce its dimensions to $HW \times D$ pixels, as explained in Sec. 2.3.1.1, where D is the desired spectral depth ($D < Nb$). Then, we reshape the modified matrix back as a data cube of $H \times W \times D$ pixels. As a result, for the Indian Pines, Pavia University, and Salinas datasets, we reduce the number of channel inputs to 30 by selecting the top 30 principal components, retaining 99.285%, 99.966%, and 99.99% of the variance, respectively. On the other hand, the EuroSAT dataset already has only nine spectral bands, so it is left without modification.

Since the IP, PU, and SA datasets consist of one big single image, we have to divide them into small patches so that each patch represents one class. Thus, we extract square patches using a 25×25 pixel window around each pixel. Furthermore, we only collect patches around those pixels with an assigned label. By doing so, the new IP dataset consists of 10,249 patches; the new PU dataset of 42776 patches; and the new SA dataset of 54,129 patches. The final shape of a sample from the IP, PU, and SA datasets is $25 \times 25 \times 30$ pixels; however, since the first layer of our proposed architecture requires a 4-D input, we reshape each sample to $25 \times 25 \times 30 \times 1$ pixels. In other words, instead of considering the input as a stack of 30 images of 25×25 pixels, we consider it as one data cube of $25 \times 25 \times 30$ pixels. For the case of the EuroSAT, we reshape each sample to $64 \times 64 \times 9 \times 1$ pixels.

3.2.1.2 Kochia Dataset Each of the 76 collected images in the Kochia dataset has a height of 900 pixels and a width between 700 and 1200 pixels. They contain three kochia leaves of the same herbicide-resistance class and a Spectralon panel. The white band shown on the left side of each image in Fig. 3.3 corresponds to the Spectralon panel, which is a 99% reflective material commonly used as a Lambertian calibration reference. These images were captured by the Pika L hyperspectral imager in raw digital numbers, meaning the image data required pre-processing prior to analysis. For our experiments on the Kochia dataset, we convert the raw digital numbers to reflectance values using the Spectralon panel as a reflectance reference.

The first step of the reflectance correction was to select the pixels in the image that contain the Spectralon reflectance target. Next, we averaged the values of all pixels within the selected region at each spectral band, leaving us with a single, averaged digital number for each spectral band $n \in [1, N]$ (where N represents the total number of spectral bands) denoted as $DN_{target}^{(n)}$. Consequently, we obtained a digital number representing 99% of the reflected light for each spectral band. Finally, we calculated the spectral reflectance at the

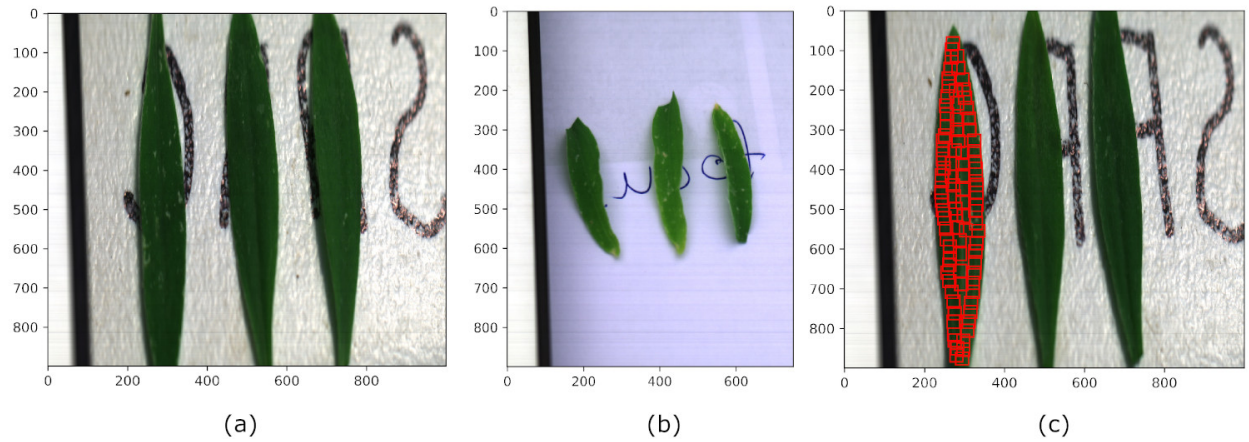

Figure 3.3: Kochia leaves at **(a)** 8 weeks and **(b)** 2 weeks. **(c)** An example of the extraction of 25×25 pixel patches.

i-th pixel of the n-th spectral band, $\rho_i^{(n)}$, as follows:

$$\rho_i^{(n)} = \left(\frac{DN_{scene_i}^{(n)} - DN_{dark}^{(n)}}{DN_{target}^{(n)} - DN_{dark}^{(n)}} \right) \rho_{target},$$

where $DN_{scene_i}^{(n)}$ represents the digital number value captured at the i-th pixel of the n-th spectral band, $DN_{dark}^{(n)}$ represents the dark current or background signal generated through sporadic electron generation in the imager's sensor, and ρ_{target} represents the reflectivity of the reflectance target.

From the calibrated images, we extracted 6,316 25×25 pixel overlapping patches (Fig. 3.3c). Then, as with the remote sensing datasets, we applied PCA to reduce the number of spectral bands from 300 to 100 with an explained variance of 99.565%. The difference with the previous datasets is that this dataset has dimensions $6316 \times 25 \times 25 \times 300$, so we reshaped it to $3,947,500 \times 300$ pixels; then, we applied PCA to reduce it to $3,947,500 \times 100$ pixels; and we reshaped it to $6316 \times 25 \times 25 \times 100$ pixels. Finally, similar to what we did with the remote sensing datasets, we reshaped each sample to $25 \times 25 \times 100 \times 1$ pixels.

3.2.2 Hyperspectral Band Selection Approach

Even if we apply a feature extraction technique like PCA to alleviate the need for powerful processing hardware, we still have to face the prohibitive cost of storing the original data. Therefore, it may be more convenient to identify the most important spectral wavelengths from the original spectrum and discard the rest, as we will discuss in Ch. 5. Band selection methods also allow us to design multispectral sensors with a reduced set of spectral bands using the set of identified important wavelengths.

Here, we describe the pre-processing steps we apply to the hyperspectral datasets before passing them to a band selection algorithm that will identify the most relevant wavelengths without transforming the data. In this approach, we only use the Kochia, Indian Pines, and Salinas datasets for experimentation.

3.2.2.1 Remote Sensing Datasets As we described in Sec. 3.2.1.1, the IP and SA datasets consist of single large images, so we have to divide them into small patches so that each patch represents one class. In this case, we extract 5×5 pixel patches around each pixel and we do not apply any feature extraction technique. Hence, the size of the preprocessed IP dataset is $10,249 \times 5 \times 5 \times 200 \times 1$ while that of the preprocessed SA dataset is $54,129 \times 5 \times 5 \times 204 \times 1$.

Note that in the previous approach we extracted 25×25 pixel patches instead of 5×5 pixel patches; however, during the experiments described in Ch. 5, we noticed that the classification performance was extremely high ($> 99.5\%$) independently of the wavelengths that were selected by the different band selection methods. For that reason, we decided to "increase the difficulty" of our classification tasks by reducing the size of the image patches, as we noticed that the lower spatial information a CNN classifier has, the lower its classification performance is. By doing so, we were able to appreciate how different band selection methods behave under difficult scenarios.

3.2.2.2 Kochia Dataset We use the 6,316 25×25 pixel patches that we extracted from each of the 76 kochia images, as we described in Sec. 3.2.1.2. Furthermore, we reduced the number of spectral bands within each patch from 300 to 150 by averaging adjacent pairs of bands, which can be interpreted as $2\times$ spectral binning, where the resulting spectral resolution of each channel was modified from approximately 2.12 nm to 4.24 nm. In other words, the final shape of this preprocessed dataset is $6316 \times 25 \times 25 \times 150 \times 1$ pixels.

Given that one of the goals of this work is to aid in the design of multispectral imaging systems, we note that decreasing the overall spectral resolution is unlikely to affect our results, as it is unlikely that optical filters with a bandwidth less than 20 nm will be considered. Hence, this process gives us an upfront reduction in dimensionality that greatly reduces the potential overfitting impact in our following analysis without overly constraining the design parameters of a multispectral imager. In addition, averaging consecutive bands has a smoothing effect that helps to combat possible noise in some spectral bands.

CHAPTER FOUR

REDUCED-COST HYPERSPECTRAL IMAGE CLASSIFICATION

In this chapter, we use the concepts introduced in Sec. 2.2 to design a convolutional neural network architecture for hyperspectral image classification called Hyper3DNet. This network was designed to meet comparable state-of-the-art classification performance while increasing computational efficiency. We begin with an overview of related work, outline the specific network architecture, discuss the classification method in detail, and provide experimental results.

4.1 Related Work

Many of the classic HSI classification methods have focused on pixelwise classification in the spectral domain, not taking advantage of the spatial information. Typically, these methods are based on support vector machines (SVMs) [69, 75], k-nearest-neighbors [90], random forests [29], or feedforward neural networks [91]. Because of the high dimensionality, other methods use a feature engineering step before training a classifier [45]. This can be done using feature extraction or feature selection approaches: the former apply linear or non-linear transformations to extract specific features from the original data [108, 121], while the latter select the most useful individual features (i.e. spectral bands) of the data without transforming it [111, 113]. Some approaches based on SVMs using a composition of kernels [16], 3-D wavelet filters [9], 3-D Gabor filters [44] or conditional random fields [57] show improved classification performance, taking into account both spatial and spectral information. Their main drawback, however, is that they require hand-crafted spatial features.

Recently, deep learning techniques are being used because of their capability of learning

hierarchical feature representations. Convolutional neural networks (CNNs) are commonly used with grayscale or RGB images as inputs, which is why some remote sensing applications based on RGB satellite imagery can rely on relatively standard network architectures [65]; however, when the number of spectral bands (or channels) of the data input increases, more specialized network architectures are needed [37]. In order to exploit the spectral and the spatial information jointly, many approaches are based on 3-D convolutional neural networks, as suggested by Pan *et al.* [78] and Chen *et al.* [11]. Similarly, Zhong et al. [126] designed a spectral-spatial residual network (SSRN) architecture that consisted of two main sections: spectral feature learning and spatial feature learning. The first one reduces the number of original spectral bands and applies spectral kernels that affect only the spectral domain, while the second one uses spatial kernels that learn deep spatial representations.

Due to the fact that some spectral bands can be considered as noise or simply not relevant for a certain classification task, Fang et al. [15] introduced a spectral-wise attention mechanism that emphasizes the most informative spectral features. Ma et al. [64] applied the same attention mechanism but used a double-branch 3D-CNN to extract the spectral and spatial features, respectively. Similarly, Gao et al. [19] proposed a feature boosting and suppression method (FBS) in a form of an attention mechanism designed to dynamically amplify and suppress output channels from convolutional layers within a CNN.

Since 2-D CNNs cannot learn channel relationship information and 3-D CNNs are more computationally complex, Roy et al. [89] proposed a hybrid-CNN model that deals with these two shortcomings. In addition to this, before training the model, they remove the spectral redundancy and reduce the dimensionality of the raw data using principal component analysis (PCA). On the other hand, other approaches apply a sequence-based methodology using recurrent neural networks (RNNs) [73], a combination of CNNs and RNNs [118], or convolutional long-short term memory (LSTM) [95].

4.2 The Hyper3DNet Architecture

We present an HSI classification method using a novel 3D-2D CNN architecture called Hyper3DNet. Since a hyperspectral image is basically a set of hundreds of 2-D images, we consider one HSI as a single data cube. As a consequence, the general input shape of our network is $W \times W \times D \times 1$; that is, one data cube with a width and height of W pixels and a spectral depth of D, as explained in Sec. 3.2.1.

Fig. 4.1 shows our network architecture and its two main modules: a 3-D feature extractor and a 2-D spatial encoder. The feature extractor consists of a 4-layer densely connected block, where each layer is composed of a $3 \times 3 \times 7$ 3-D convolution layer of eight filters, denoted as "CONV3D", a batch normalization layer, denoted as "BN", and a rectified linear unit activation layer, denoted as "ReLU". Notice that the difference between the diagram shown in Fig. 2.5 and the feature extractor shown in the left part of Fig. 4.1 is that the latter uses 3-D convolutions, which is why the concatenation ("CONCAT") is carried out along the fourth dimension so that the output of the feature extractor is a stack of 32 data cubes.

The second part of the Hyper3DNet architecture is the spatial encoder, which is shown in the right part of Fig. 4.1. Note that the input tensor coming from the 3-D feature extractor section has to be reshaped into a 3-D tensor before performing any 2-D convolution operation. Our spatial encoder takes a 3-D input and gradually compresses it into an encoded feature vector representation. This is done using four 3×3 separable convolutions of 128 filters, denoted as "SEP CONV", followed by a batch normalization layer and a rectified linear unit activation layer. The spatial dimensionality is reduced using a stride of 2 in the last three convolution operations. Finally, we reshape our output into a 1-D tensor and use a fully connected layer with softmax activation, denoted as "FC", to obtain the final classification result.

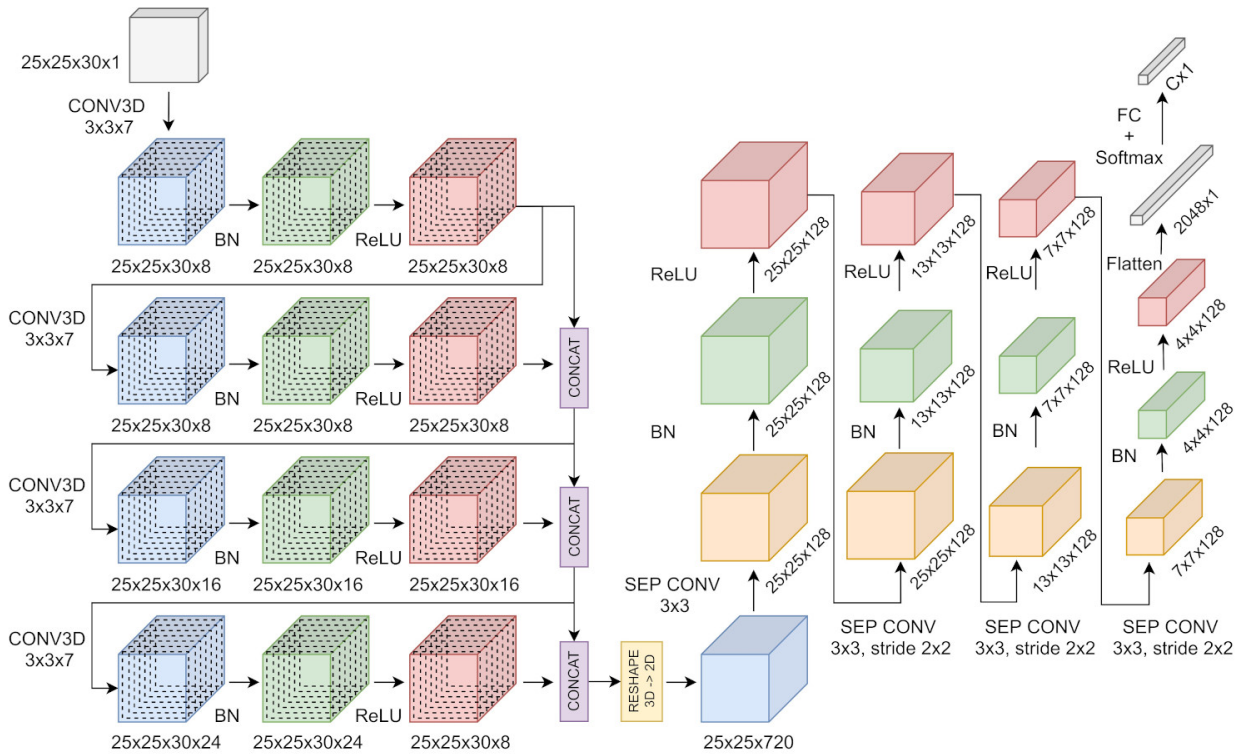

Figure 4.1: Hyper3DNet architecture with a $25 \times 25 \times 30 \times 1$ HSI input. The network includes a 3-D feature extractor (left) and a 2-D spatial encoder (right).

4.3 Results

All experiments are conducted in Python 3.6 on a PC equipped with an Intel Xeon CPU E5-2603 v4 at 1.70GHz, 128GB RAM, and two NVIDIA GeForce GTX 1080 Ti GPUs. The proposed CNN was trained using the Adam optimizer [47] with a learning rate of 0.0001, a momentum term β_1 of 0.9, a momentum term β_2 of 0.999, and an epsilon value of 10E-8. We used a grid search to determine the best mini-batch sizes for each dataset. Hence, the mini-batch sizes chosen for IP, PU, SA, EuroSAT, and Kochia datasets are 4, 4, 4, 8, and 128, respectively. Due to the fact that the last layer of our model is a softmax activation, the categorical cross-entropy function was chosen as our loss function.

We used 10-fold stratified cross-validation to train and evaluate all the networks. That

is, we randomly divided our datasets into 10 equally sized folds; then, the training process was repeated 10 times using each random fold as an independent validation set. Furthermore, stratification indicates that each fold has the same proportion of samples of a given class. For each iteration, we trained our network for 50 epochs on Indian Pines and the Kochia dataset; 40, on Pavia University and Salinas; and 150, on EuroSAT. The maximum number of epochs for each dataset was determined empirically and all compared models reached convergence in fewer epochs.

Furthermore, we show in Fig. 4.2 the mean spectral reflectance curve of all the pixels of the Kochia dataset. Here, higher reflectance values are obtained for the last bands of the spectrum (i.e. $750 - 1023\,\text{nm}$). If we were to directly feed this data into our networks, the last bands would have more contribution than the initial bands. In other words, the learning process would be dominated by those bands that inherently present higher reflectance values. Therefore, we apply z-score normalization onto each spectral band so that all bands present mean equal to zero and standard deviation equal to one. Note that z-score normalization was applied onto each spectral band of each training set while the exact same scaling was applied to their corresponding validation set.

In order to analyze the behavior of our Hyper3DNet model statistically, we calculated four metrics on the validation sets: accuracy (OA), precision ($Prec$), recall (Rec), and F1 score. OA indicates the fraction of correctly predicted observations among total observations. $Prec$ indicates the macro-average precision; that is, the average precision for all the classes. The precision for class c is the fraction of correctly predicted positive observations (TP_c) among total predicted positive observations, which is defined as the sum of correctly predicted positive observations and incorrectly predicted positive observations (FP_c). Rec indicates the macro-average recall. The recall for class c is the fraction of correctly predicted positive observations among total actual positive observations, which is defined as the sum of correctly predicted positive observations and incorrectly predicted negative observations (FN_c). The

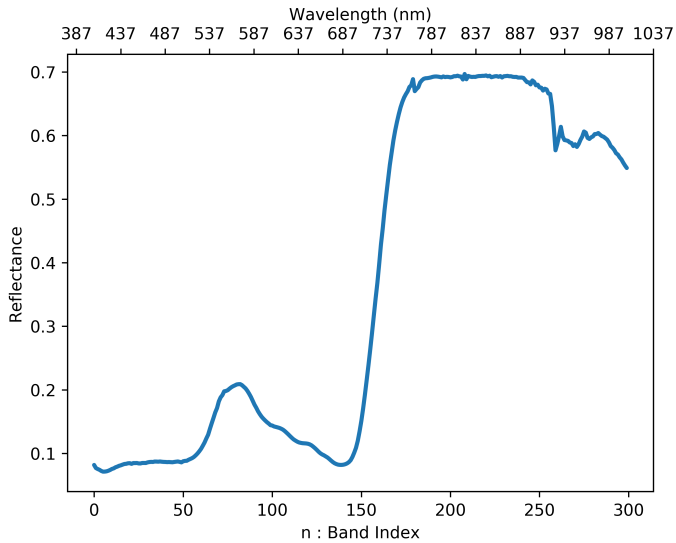

Figure 4.2: Mean spectral reflectance curve of all the pixels of the Kochia dataset.

F1 score is the harmonic mean of $Prec$ and Rec. The equations for all the metrics are given below:

$$OA = \frac{\#correctly\ classified\ observations}{\#total\ observations},$$

$$Prec = \frac{1}{\#classes} \sum_{c=1}^{\#classes} \frac{TP_c}{TP_c + FP_c},$$

$$Rec = \frac{1}{\#classes} \sum_{c=1}^{\#classes} \frac{TP_c}{TP_c + FN_c},$$

$$F1 = 2 \cdot \frac{Prec \cdot Rec}{Prec + Rec}.$$

The source code and datasets are available online[1].

[1]Codebase: `https://github.com/GiorgioMorales/hyper3dnet`.

4.3.1 Compared Methods

The results of Hyper3DNet were compared to those of three supervised methods: ResNet50 [32], SpectrumNet [94], and HybridSN [89]. ResNet50 was selected for this comparison because it constitutes a deep convolutional network that has achieved excellent generalization performance on many image classification tasks. In a similar way, HybridSN was chosen because it outperforms most of the state-of-the-art methods for HSI classification. It is worth mentioning that, for the case of the EuroSAT dataset, we applied zero-padding in the first three 3-D convolution blocks of HybridSN in order to avoid dimensionality inconsistencies. Finally, SpectrumNet was also considered for this comparison for it is a small and computationally efficient CNN, and it is also capable of being trained to process the spectral-spatial input space from multi-spectral satellite imagery.

Although shown to be a high-performance method, ResNet50 turns into a computationally inefficient method when it is used with such input dimensions as those of the Kochia dataset (i.e., it requires 23,912,394 parameters). Consequently, for the special case of the Kochia dataset, we decided to include the method proposed by Scherrer *et al.* [91] to the list of compared methods instead of ResNet50. This supervised method consists of a fully connected feedforward neural network with two hidden layers and 500 units in each layer (denoted simply as KochiaFC). Given that this network processes 1-D vectors, we modified our existing dataset as follows: First, we extracted a 10×10 sub-patch from the center of each patch of the Kochia dataset to make sure that these sub-patches did not overlap; secondly, we took all of the pixels of these sub-patches and put them into a new 1-D vector, discarding those that had a Normalized Difference Vegetation Index (NDVI) less than 0.6.

The KochiaFC network was not used to train the other datasets because it performs pixel-wise classification; thus, it would not be able to capture the underlying spatial information of a 2-D window. For instance, take a 64×64 pixel sample of class "*residential building*" from the EuroSAT dataset; given only the spectrum of one pixel

of the window, KochiaFC will not classify it confidently into the correct class, for that pixel may be located inside a vegetation region (e.g. a tree or a garden) so that the probability of classifying it as "*forest*" or "*pasture*" could be high. Moreover, even if we select a pixel located on top of a building, the model will not have enough information to differentiate between class "*residential building*" and class "*industrial building*".

Furthermore, in order to determine if the difference in performance scores is statistically significant, we performed a paired t-test between Hyper3DNet and the other networks with respect to the F1 scores. For this, we made sure that the datasets were split in the same way (i.e., using a fixed random seed) so that, for each fold, both compared methods saw the same training and validation sets. In this case, the null hypothesis is that the samples of F1 scores were drawn from the same distribution at significance level α. For these experiments, we used $\alpha = 0.05$. In other words, we rejected the null hypothesis if the calculated p-value was smaller than $\alpha = 0.05$.

Table 4.1 shows the number of parameters and the number of floating-point operations needed (FLOPs) to classify a patch for each of the compared methods on the IP, SU, SA, and EuroSAT datasets. The minimum number of parameters and MFLOPS are highlighted in bold font. Table 4.2 shows the same type of comparison for the special case of the Kochia dataset, where we used KochiaFC instead of ResNet50. During testing, the memory consumed by a model is used to store the outputs of intermediate layers as well as the parameters of the model. In addition, the number of parameters is related to the complexity of the model and the amount of data needed to train the model effectively. On the other hand, the number of FLOPS is related to the speed of processing the model.

4.3.2 Results on the IP, PU, and SA datasets

The average classification performance and corresponding standard deviations on the Indian Pines, Pavia University, and Salinas datasets are reported in Table 4.3, Table 4.5, and

Dataset	IP, SA		PU		EuroSAT	
Method	**# Param.**	**MFLOPS**	**# Param.**	**MFLOPS**	**# Param.**	**MFLOPS**
ResNET50	23,705,168	199.56	23,637,705	199.53	23,573,898	337.15
HybridSN	5,122,176	105.44	5,121,273	105.44	15,965,562	**191.04**
SpectrumNet	741,040	**30.10**	737,449	**30.07**	729,898	205.61
Hyper3DNet	**244,328**	89.85	**228,897**	89.82	**200,322**	193.63

Table 4.1: Comparison of number of trainable parameters and MFLOPS of the different networks trained on the IP, PU, SA, and EuroSAT datasets.

Method	# Param.	MFLOPS
KochiaFC	**402,503**	**0.8**
HybridSN	6,410,739	311.35
SpectrumNet	1,479,907	28.44
Hyper3DNet	523,483	213.73

Table 4.2: Comparison of number of trainable parameters and MFLOPS of the different networks trained on the Kochia dataset.

Table 4.4, respectively. We also constructed a box plot, shown in Fig. 4.3, to represent the distribution of the resulting F1 scores. Here, the line through the center of each box indicates the median value of the F1 scores. The edges of boxes are the 25th and 75th percentiles. Whiskers extend to the maximum and minimum points. Outlier points are those past the end of the whiskers. The classification maps of different methods for the IP, PU, and SA datasets are shown in Fig. 4.4, Fig. 4.5, and Fig. 4.6, respectively. These classification maps were generated using the validation fold with the lowest accuracy performance for each one of the datasets.

Using the paired t-test, we found that the improvements of Hyper3DNet over HybridSN and ResNet50 are not statistically significant; that is, the resulting p-values are greater than $\alpha = 0.05$. When compared to the F1 scores of SpectrumNet, we achieved p-values

Method	OA	Prec	Rec	F1
SpectrumNet	98.75 ± 0.39	95.95 ± 1.33	98.37 ± 0.92	96.54 ± 1.10
ResNET50	99.84 ± 0.12	99.51 ± 0.63	99.50 ± 0.65	99.47 ± 0.56
HybridSN	99.98 ± 0.04	99.98 ± 0.04	99.93 ± 0.19	99.95 ± 0.11
Hyper3DNet	99.94 ± 0.08	99.84 ± 0.26	99.89 ± 0.19	99.86 ± 0.19

Table 4.3: Metrics comparison on the IP dataset.

Method	OA	Prec	Rec	F1
SpectrumNet	99.57 ± 0.23	99.21 ± 0.45	98.97 ± 0.58	99.08 ± 0.51
ResNET50	99.79 ± 0.27	99.83 ± 0.10	99.60 ± 0.51	99.71 ± 0.30
HybridSN	99.98 ± 0.04	99.98 ± 0.04	99.93 ± 0.19	99.95 ± 0.11
Hyper3DNet	99.99 ± 0.01	99.99 ± 0.02	99.98 ± 0.04	99.98 ± 0.03

Table 4.4: Metrics comparison on the PU dataset.

Method	OA	Prec	Rec	F1
SpectrumNet	99.72 ± 0.23	99.48 ± 0.42	99.44 ± 0.50	99.45 ± 0.46
ResNET50	99.72 ± 0.09	99.63 ± 0.19	99.66 ± 0.11	99.64 ± 0.13
HybridSN	99.95 ± 0.04	99.93 ± 0.04	99.92 ± 0.05	99.93 ± 0.05
Hyper3DNet	99.96 ± 0.04	99.94 ± 0.06	99.94 ± 0.05	99.94 ± 0.06

Table 4.5: Metrics comparison on the SA dataset.

of 8.3E-5, 5.9E-4, and 3.7E-3 for Indian Pines, Pavia University, and Salinas datasets, respectively, which means we have a statistical basis to reject the null hypothesis. However, from Table 4.1, we observe that Hyper3DNet requires fewer trainable parameters than the compared methods.

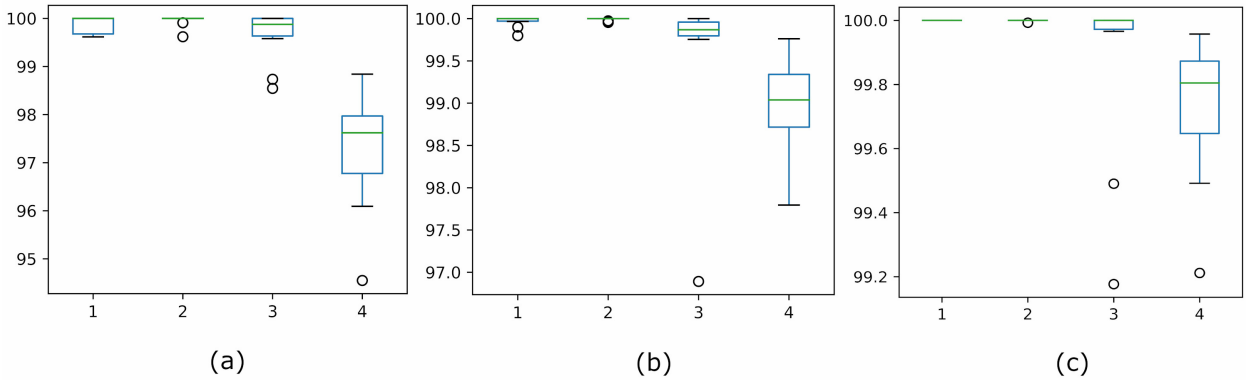

Figure 4.3: Box plot of F1 scores on the **(a)** Indian Pines, **(b)** Pavia University, and **(c)** Salinas datasets. Numbers in the abscissa corresponding to 1) Hyper3DNet; 2) HybridSN; 3) ResNet50; 4) SpectrumNet.

Method	**OA**	**Prec**	**Rec**	**F1**
SpectrumNet	96.08 ± 0.58	96.08 ± 0.49	95.87 ± 0.57	95.94 ± 0.56
ResNET50	94.48 ± 0.62	94.49 ± 0.52	94.15 ± 0.72	94.24 ± 0.66
HybridSN	98.84 ± 0.37	98.85 ± 0.38	98.74 ± 0.41	98.78 ± 0.39
Hyper3DNet	99.24 ± 0.15	99.24 ± 0.15	99.19 ± 0.17	99.21 ± 0.16

Table 4.6: Metrics comparison on the EuroSAT dataset.

4.3.3 Results on the EuroSAT dataset

Results on the EuroSAT dataset are reported in Table 4.6. Fig. 4.7 shows a box plot of the distribution of the resulting F1 scores. We also performed paired t-tests between Hyper3DNet and HybridSN, ResNet50 and SpectrumNet, finding that the improvements of Hyper3DNet on F1 scores are statistically significant with p-values 1.9E-3, 4.2E-9, and 9.3E-8, respectively. In addition, Table 4.1 shows that Hyper3DNet requires fewer trainable parameters than the compared methods.

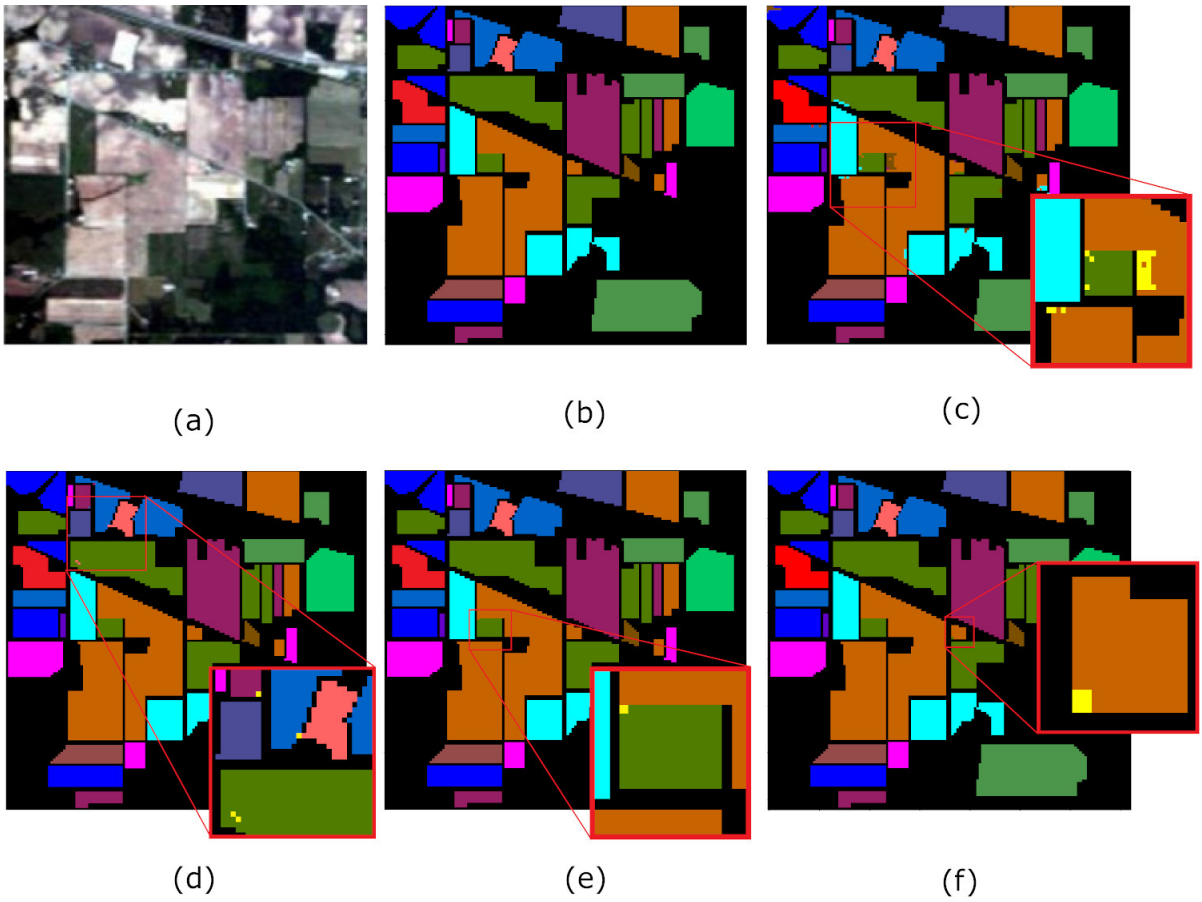

Figure 4.4: Classification maps of the Indian Pines dataset. **(a)** RGB image. **(b)** Ground-truth labels. **(c)** SpectrumNet. **(d)** ResNet50. **(e)** HybridSN. **(f)** Hyper3DNet. Zoomed images show some misclassified pixels in yellow.

4.3.4 Results on the Kochia dataset

Table 4.7 shows the results on Kochia dataset and Fig. 4.8 depicts the distribution of the F1 scores achieved by all the compared methods. Given that this dataset consists of only three classes, we show the four resulting confusion matrices in Fig. 4.9. Here, the numbers in the abscissa correspond to 0) Dicamba-resistant; 1) Glyphosate-resisant; 2) Herbicide-susceptible.

We trained KochiaFC using our own configuration and the results are included here; however, we should note this is not a fair comparison since we did not use exactly the

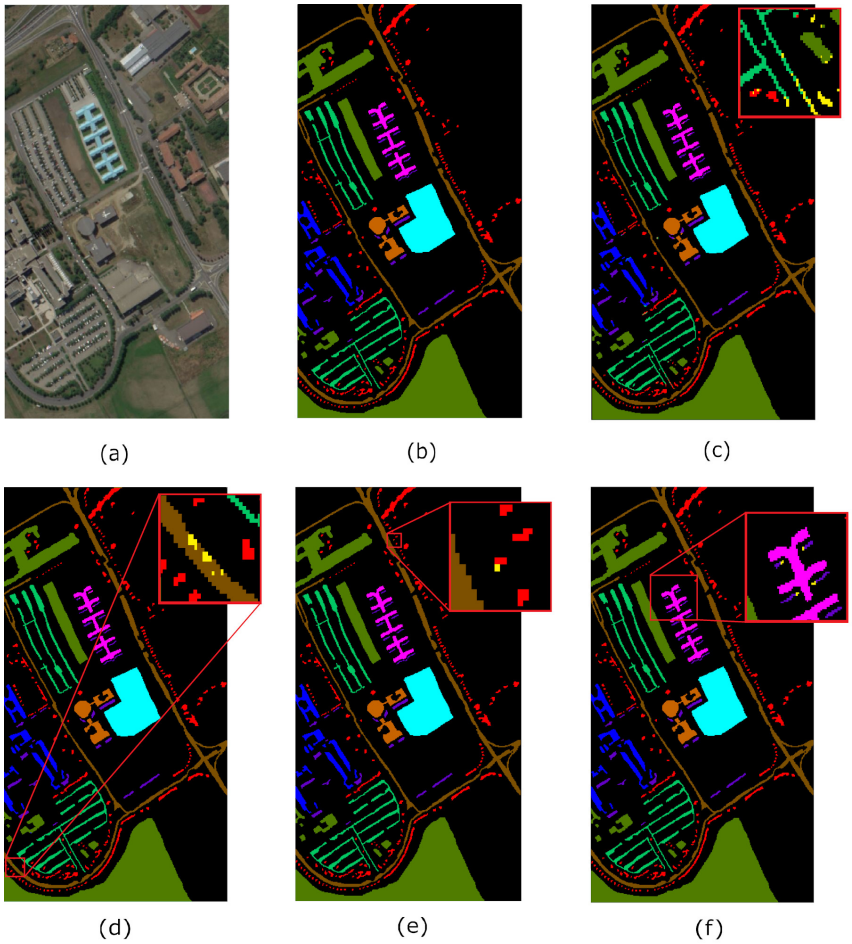

Figure 4.5: Classification maps of the Pavia University dataset. **(a)** RGB image. **(b)** Ground-truth labels. **(c)** SpectrumNet. **(d)** ResNet50. **(e)** HybridSN. **(f)** Hyper3DNet. Zoomed images show some misclassified pixels in yellow.

Method	**OA**	**Prec**	**Rec**	**F1**
KochiaFC	92.89 ± 0.29	93.09 ± 0.30	92.82 ± 0.40	92.95 ± 0.29
HybridSN	98.09 ± 0.62	98.18 ± 0.61	98.28 ± 0.56	98.22 ± 0.58
SpectrumNet	97.59 ± 0.55	97.81 ± 0.59	97.77 ± 0.46	97.78 ± 0.51
Hyper3DNet	99.55 ± 0.22	99.62 ± 0.19	99.57 ± 0.23	99.59 ± 0.21

Table 4.7: Metrics comparison on the Kochia dataset.

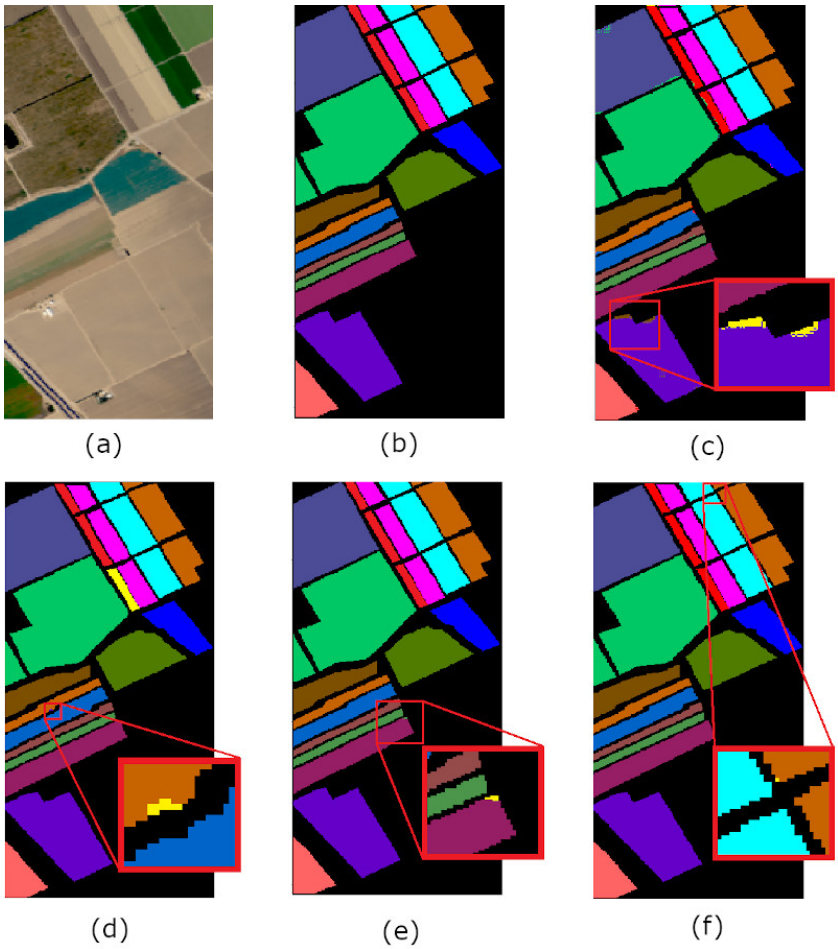

Figure 4.6: Classification maps of the Salinas dataset. **(a)** RGB image. **(b)** Ground-truth labels. **(c)** SpectrumNet. **(d)** ResNet50. **(e)** HybridSN. **(f)** Hyper3DNet. Zoomed images show some misclassified pixels in yellow.

same dataset or input dimensions as in the other methods. This is because KochiaFC is a feedforward network that performs pixel-wise classification and it required us to adapt our dataset before training it, as explained in Sec. 4.3.1. Therefore, the t-test is performed between Hyper3DNet and HybridSN, and Hyper3DNet and SpectrumNet, achieving p-values of 1.60E-05 and 1.40E-05, respectively, thus indicating significantly better performance by Hyper3DNet.

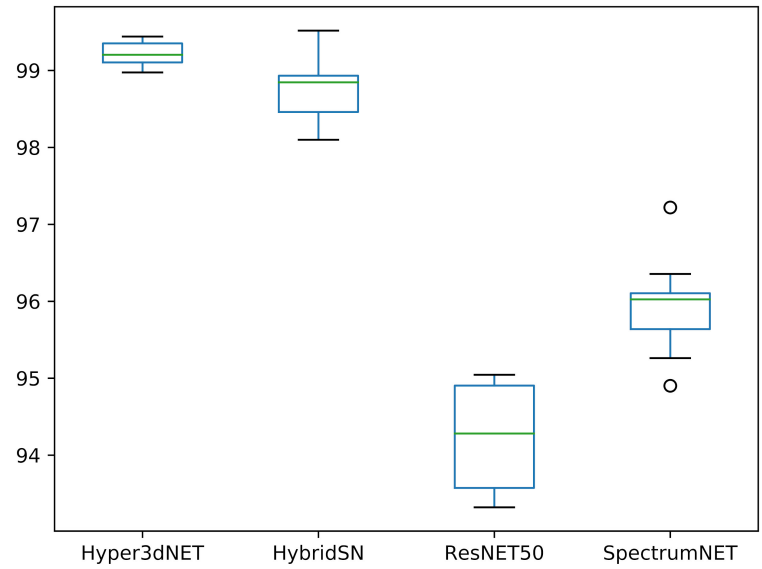

Figure 4.7: Box plot of F1 scores of different methods on the EuroSAT dataset.

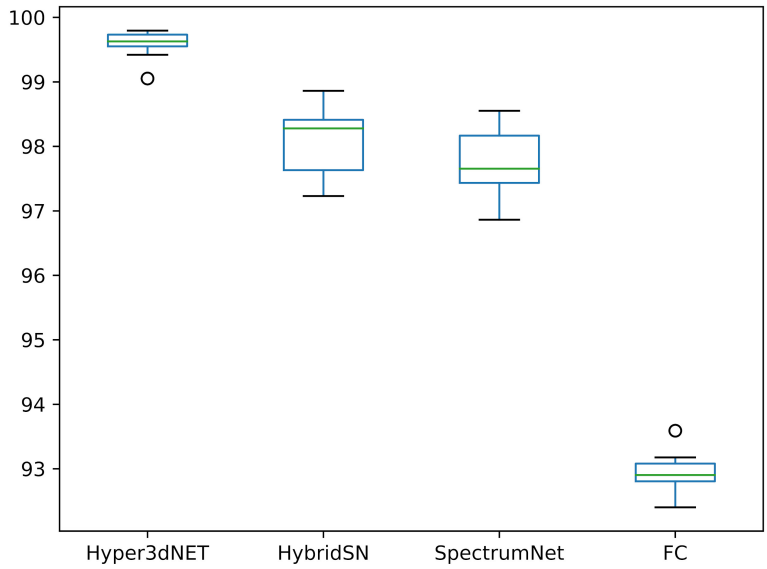

Figure 4.8: Box plot of F1 scores of different methods on the Kochia dataset.

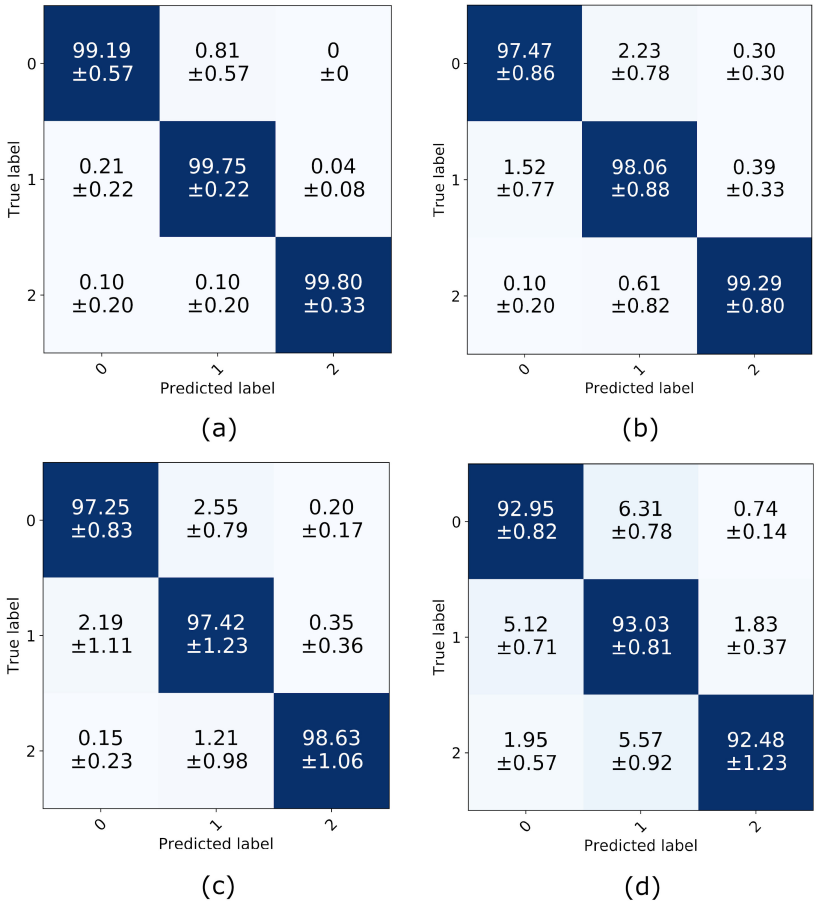

Figure 4.9: Confusion matrices of different networks on the Kochia dataset. **(a)** Hyper3DNet. **(b)** HybridSN. **(c)** SpectrumNet. **(d)** KochiaFC.

4.3.5 Discussion

From Table 4.3, Table 4.5, Table 4.4, Table 4.6, and Table 4.7, we conclude that our network achieves the highest performance scores regardless of which dataset is used. Nevertheless, we calculated that there is no statistical significance in the difference of performance of Hyper3DNet, HybridSN, and ResNet50 when using Indian Pines, Pavia University, and Salinas datasets. The reason for this is that these datasets are easy to learn, so we should focus on the other two datasets to better understand our network's performance. We can claim, however, that we achieve comparable state-of-the-art results for

these three datasets with a substantially reduced-complexity model, given that it requires fewer parameters. As a result, our model is less prone to overfitting and requires less data to train its parameters. Furthermore, Table 4.6 and Table 4.7 demonstrate important performance improvements of Hyper3DNet over the other methods, which can also be verified in Fig. 4.7 and Fig. 4.8. In fact, the t-tests we performed indicate that the improvements on the observed F1 scores on EuroSAT and Kochia datasets are statistically significant.

From Table 4.1 and Table 4.2, we note that Hyper3DNet presents the minimum number of parameters in all the cases, except for KochiaFC, without detriment to its performance, which is always the highest. It should be noted, however, that despite Hyper3DNet being a much more complex architecture than KochiaFC (two fully connected layers), they differ in number of parameters by just 120,980. Furthermore, as stated previously, having a small number of parameters is important due to the fact that they are loaded into the memory during test time, so that reducing this number implies fewer memory accesses and an improvement in the power efficiency [116]. For instance, the parameters of the Hyper3DNet model trained on EuroSAT require 2,494 KB while those of ResNet50 require 277,216 KB.

Also from Table 4.1, we observe that our method is second in terms of MFLOPS after HybridSN in the case of EuroSAT dataset, and after SpectrumNet in the case of the remaining datasets. This is explained by the fact that SpectrumNet reduces the size of the input tensor using a 2-D convolution block with a stride of 2×2 and then uses a max-pooling layer after the fourth and eighth convolutional blocks. Those operations reduce the size of the outputs of the intermediate layers of the network substantially and, as a result, the number of multiply-and-accumulate operations; however, this drastic reduction causes a loss of useful spatial information that affects the performance of the network. On the other hand, when the window size of the input tensor is higher (i.e., 64×64 for EuroSAT), the spatial reduction is less drastic, thus increasing the number of MFLOPS over that of Hyper3DNet.

For both Hyper3DNet and HybridSN, the 3-D convolutions require less than 4 and 1

MFLOPS, respectively, because of the small number of channels or filters used (up to 24 for Hyper3DNet, see Fig. 4.1). The real bottleneck is located after the *reshape* layer used in both networks to reshape 3-D tensors into 2-D tensors. Take the example of Hyper3DNet shown in Fig. 4.1. After the reshaping operation, we get an output size $64 \times 64 \times 960$ (considering we are working with EuroSAT) and then we apply a 3×3 separable convolution operation of 128 filters. This operation requires 161.61 MFLOPS, which represents 83.4% of the total number of operations for the network. Similarly, the convolution operation carried out after the reshaping operation of HybridSN takes 159.48 MFLOPS, which also represents 83.4% of the total number of operations. Nevertheless, while our network requires 2.61 more MFLOPS than HybridSN for this dataset, we can argue it is a more efficient architecture because it needs 15,765,240 fewer parameters, not to mention its superior performance.

4.4 Conclusions

In this chapter, we presented a novel deep neural network, called Hyper3DNet, for addressing the problem of HSI classification. Many previous approaches have focused solely on improving the performance metrics, while not placing enough importance on the computational cost. In contrast, our network architecture has been designed explicitly to reduce the number of trainable parameters and computational operations. The experimental results show that Hyper3DNet consistently achieves the highest classification accuracy within a variety of HSI scenarios, demonstrating that the reduced complexity of the model does not affect its performance and makes it less prone to overfitting. In Chapter 5, we propose a simplified version of Hyper3DNet called Hyper3DNetLite designed for multispectral image classification.

CHAPTER FIVE

HYPERSPECTRAL BAND SELECTION

In this chapter, we use the concepts introduced in Sec. 2.3 to design novel dimensionality reduction methods in the context of hyperspectral image classification. In particular, we propose a filter-based method called Inter-Band Redundancy Analysis (IBRA). In addition, we propose a band selection method that uses IBRA to obtain a reduced set of candidate bands; then, it applies a wrapper-based approach called Greedy Spectral Selection (GSS) to the results of IBRA to select the desired number of bands. We also propose a feature extraction framework that consists of two main steps: first, it reduces the total number of bands using IBRA; then, it can use any feature extraction method to obtain the desired number of feature channels.

The remainder of this chapter is structured as follows: In Section 5.1, we provide a brief review of previous related work done with feature extraction and feature selection methods in the context of hyperspectral image classification. Section 5.2 describes our IBRA and GSS algorithms and explains how they are used for dimensionality reduction. Section 5.3 provides and discusses experimental results.

5.1 Related Work

Dimensionality reduction techniques rely on feature extraction or feature selection approaches; the former apply linear or non-linear transformations to extract specific features from the original data, while the latter select the most useful subset of the features of the data without transforming them.

Feature extraction methods can be subdivided into two classes: unsupervised and supervised. Among unsupervised methods, principal component analysis (PCA) and its

variants (e.g. folded-PCA and kernel PCA) are some of the most commonly used methods to remove spectral redundancy and reduce the dimensionality of the raw data [108]. Nevertheless, theoretical arguments and experimental evidence of the ineffectiveness of PCA for hyperspectral feature extraction have been presented [83]. According to this, PCA should not be used as a pre-processing step to solve small-sample-size problems as there is evidence that class separation deteriorates after PCA transformation. Other unsupervised methods are based on independent component analysis (ICA) and its variants (e.g. kernel ICA and Random Fourier Feature-ICA) [42]. Alternatively, supervised methods such as Fisher's linear discriminant analysis (LDA) [14, 100] or partial least squares discriminant analysis (PLS-DA) [84] are used widely for hyperspectral feature extraction.

On the other hand, feature selection methods select a subset of spectral bands without modifying the data or projecting it into a new basis. In other words, the aim of a hyperspectral band selection method is to identify which spectral wavelengths are most responsive or relevant for a particular classification task without modifying the data. Additionally, identifying a reduced subset of relevant spectral bands allows for a better understanding of the optical properties of the materials and provides information that is useful when designing cheaper task-specific multispectral imagers.

Given the potential benefits of feature selection, several methods have been proposed for hyperspectral image classification. In this context, some feature extraction methods can be adapted for filter-based feature selection. For example, Chang *et al.* [8] presented a joint band-prioritization method based on PCA where the spectral bands are ranked according to their variances. PLS-DA can also be adapted into a feature selection scheme using the resulting regression coefficients from full spectra to identify the most influential wavelengths [3, 18].

Some filter-based methods analyze the statistics of the data and heuristically estimate a scoring index for each feature. For instance, Preet *et al.* [85] employed a filter-based feature-

selection algorithm based on minimizing a tight bound on the Vapnik-Chervonenkis (VC) dimension [109]. Ranking-based methods are one of the most common filter-based feature selection methods; these methods estimate the importance of each spectral band using such metrics as the variance inflation factor (VIF) [10] in order to select the top-ranked bands. We will use the idea of calculating the VIF value to measure collinearity, but the spectral bands will not be ranked based on this simple measure alone.

Another example of a ranking-based method is the fast density-peak-based clustering (FDPC) method proposed by Rodriguez *et al.* [88]. This method ranks each band according to its ability to become a cluster center, which is measured using two factors: local density and distance from points of higher density. These factors are calculated using a similarity matrix based on Euclidean distance. Similarly, Xu *et al.* [119] constructed a similarity matrix based on structural similarity (SSIM) that is used to calculate two measures called average similarity and significant dissimilarity to evaluate the ability of each band to become a cluster center. Then it ranks the bands according to the product of these two measures so that the top-ranked bands are selected.

Other methods estimate the relevance of each band as part of an optimization approach. For instance, Medjahed and Ouali [68] formulated the feature selection problem as a combinatorial optimization problem where feature relevance is measured by mutual information. Furthermore, Wang *et al.* [114] proposed an optimal clustering framework (OCF) that searches for an optimal band partition. Here, the clustering strategy is to evaluate the contribution of each cluster and then sum the contributions as a measure of the whole clustering result. After obtaining an optimal partition, the method ranks each band within a cluster according to a selected measure (e.g., information entropy) and selects those with higher rank values. In subsequent work, Wang *et al.* [112] proposed a fast neighborhood grouping method for hyperspectral band selection (FNGBS) that uses a neighborhood band grouping approach to partition the data into several groups based on Euclidean distance.

After partitioning the data, the method ranks the bands within each group according to the product of local density and information entropy in order to obtain the most relevant and informative band for each cluster.

Recently, wrapper-based feature selection approaches based on genetic algorithms have gained more attention [17, 55, 110]. For instance, a recent method for bandwidth selection is known as Histogram Assisted Genetic Algorithm for Reduction in Dimensionality (HAGRID) [111]. This method maintains a population of index vectors identifying a specific number of wavelengths and fits a Gaussian mixture model to the converged population to identify the main wavelengths with their associated filter bandwidths.

Alternatively, some embedded-based approaches have been proposed in the context of deep learning. Taherkhani *et al.* [104] proposed regularizing the convolutional filters of the first layer of a convolutional neural network (CNN) using a group Least Absolute Shrinkage and Selection Operator (LASSO) algorithm in order to sparsify the redundant spectral bands. Similar attempts, although they are not explicit feature selection methods, have been carried out in works such as [15], and [19], where a spectral-wise attention mechanism in the form of a fully-connected layer is applied to the inner convolutional layers of the network with the objective of emphasizing informative spectral features and suppressing less useful spectral features. Furthermore, the use of saliency maps have gained popularity due to their ability to estimate which features (i.e., pixels) are more relevant for a model [77, 79]. For example, Nagasubramanian *et al.* [74] adapted the concept of spatial saliency maps to an HSI classification scheme in order to estimate the relative importance of each wavelength given a specific class.

5.2 Proposed Methods

5.2.1 Convolutional Neural Network Architecture

All the classifiers used in this chapter are CNNs that use a modified version of the Hyper3DNet architecture (see Chapter 4). This CNN architecture is referred to as Hyper3DNet-Lite. The difference with the original architecture is that its 3-D feature extractor consists of two 3-D convolutional layers instead of a densely connected block with four layers; additionally, its 2-D spatial encoder has three layers instead of four. The Hyper3DNet-Lite architecture used for the Kochia dataset is detailed in Table 5.1, where N denotes the number of spectral bands in the input, "SepConv2D" denotes a 2-D separable convolutional layer, and "ReLU" denotes a rectified linear unit activation layer (where $ReLU(x) = \max(0, x)$). The only difference with the network used to process the IP and SA datasets is that, since the input image is smaller (5×5 pixels), the stride used in the last two "SepConv2D" layers is $(1, 1)$ instead of $(2, 2)$ to avoid dimensionality inconsistencies.

Another difference of our Hyper3DNet-Lite with respect to the original Hyper3DNet is that, after the last "SepConv2D" layer, we use a "GlobalAveragePooling" operation instead of a "Flatten" operation. For example, for the case of the network shown in Table 5.1, we transform our 3-D tensor with dimensions $7 \times 7 \times 256$ pixels into a 1-D vector with 256 elements by averaging the 49 values from each channel. Conversely, if we were to use a flattening operation, we would obtain a 1-D vector with 12,544 elements. This relaxes the need of using an intermediate fully connected layer to reduce the dimensionality of the tensor before the final classification layer; thus, we avoid using an excessive number of trainable parameters.

The simplified architecture of the Hyper3DNet-Lite network becomes especially suitable for datasets that use just a few spectral bands. The reason for this is that these datasets often do not require models with a high level of complexity to process them, unlike those datasets

Layer Name	**Kernel Size**	**Stride Size**	**Output Size**
Input	—	—	(25, 25, N, 1)
Conv3D + ReLU	(3, 3, 3)	(1, 1, 1)	(25, 25, N, 16)
Conv3D + ReLU	(3, 3, 3)	(1, 1, 1)	(25, 25, N, 16)
Reshape	—	—	(25, 25, 16N)
SepConv2D + ReLU	(3, 3)	(1, 1)	(25, 25, 320)
SepConv2D + ReLU	(3, 3)	(2, 2)	(13, 13, 256)
SepConv2D + ReLU	(3, 3)	(2, 2)	(7, 7, 256)
GlobalAveragePooling	—	—	256
Dense + Softmax	—	—	# classes

Table 5.1: Hyper3DNet-Lite architecture for the Kochia dataset.

that use all the available spectral bands that benefit from deeper models (e.g.,Hyper3DNet) to extract more complex features and achieve better performance. In other words, if we used a complex architecture such as Hyper3DNet to train a classifier on a dataset with a few spectral bands (e.g., $\sim 5 - 10$), we would have a model that is too complex for the given classification task. Thus, we would have an over-parameterization problem, which results in our models being prone to overfitting.

5.2.2 Inter-Band Redundancy Analysis

Our first contribution consists of having developed a method called Inter-Band Redundancy Analysis (IBRA) that selects a subset of representative spectral bands from the original data cube aiming to reduce the inter-band redundancy. IBRA is a filter-based selection method that iteratively calculates the Variance Inflation Factor (VIF) [41] between pairs of bands in order to determine how correlated they are; that is, to verify the presence of collinearity between them.

Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ be a hyperspectral dataset with N spectral bands where each $\mathbf{x}_i = \{x_i^{(1)}, \ldots, x_i^{(T)}\}$ consists of T samples and $x_i^{(t)}$ represents the reflectance value of the i-th band in the t-th sample. To calculate the VIF value between two bands, $\mathbf{x}_i$ and $\mathbf{x}_j$, we build an ordinary least squares (OLS) regression model that takes one of the bands as the independent variable ($\mathbf{x}_i$) and the other as the dependant variable ($\mathbf{x}_j$). Then, we use the R-squared (coefficient of determination) value from the resulting model, $R^2_{x_1,x_2}$, defined as follows:

$$R^2_{x_i,x_j} = 1 - \frac{\sum_t (x_j^{(t)} - \hat{x}_j^{(t)})^2}{\sum_t (x_j^{(t)} - \bar{x}_j)^2},$$

where $\hat{x}_j^{(t)}$ represents the estimated value of $x_j^{(t)}$ using $x_i^{(t)}$ and $\bar{x}_j$ is the mean of $\mathbf{x}_j$. Then, the VIF value is calculated as:

$$VIF(\mathbf{x}_i, \mathbf{x}_j) = 1/(1 - R^2_{\mathbf{x}_i,\mathbf{x}_j}).$$

VIF values range from 1 upwards. The higher the VIF value, the more the risk of collinearity. If two bands are colinear, they explain the same variance within the dataset and can be considered to be redundant. We will consider VIF values greater than a threshold θ as representing the presence of collinearity between two spectral bands. In the literature, the recommended values of θ are between 5 and 10 [6, 27], so we test different values of $\theta \in [5, 12]$ to observe how the classification performance is affected by this threshold and to choose the best θ for a given classification task.

Some methods, such as that proposed by Castaldi *et al.* [10], use the VIF metric to perform a stepwise selection process where the bands that show a VIF value greater than a threshold θ (i.e., bands that show a high risk of multicollinearity) are removed from the selection. Our approach is novel and distinct in that we use the VIF metric as part of a pre-selection step, assessing the collinearity degree between each band and its local neighbors iteratively in order to find independent salient bands that are suitable cluster centers.

Algorithm 5.2 Calculating the interband redundancy

```
1: function INTERBANDREDUNDANCY(θ)
2:     d_left ← []
3:     d_right ← []
4:     table ← zeros(N, N) // creates an N × N matrix of zeros
5:     for all band ∈ (0, N) do
6:         // Check left side
7:         t ← 1
8:         vif ← ∞
9:         while (vif > θ) ∧ ((band − dt) > 0) do
10:            if table[band, band − t] = 0 then
11:                table[band, band − t] = getVIF(band, band − t)
12:                table[band − t, band] = table[band, band − t]
13:            vif = table[band, band − t]
14:            t ← t + 1
15:        d_left ← [d_left − 1]
16:        // Check right side
17:        t ← 1
18:        vif ← ∞
19:        while (vif > θ) ∧ ((band − dt) < N) do
20:            if table[band, band + t] = 0 then
21:                table[band, band + t] = getVIF(band, band + t)
22:                table[band + t, band] = table[band, band + t]
23:            vif = table[band, band + t]
24:            t ← t + 1
25:        d_right ← [d_right − 1]
26:    d ← |d_left − d_right|
27:    preselection ← getLocalMinima(d)
28:    return d, preselection
```

Our IBRA method first analyzes how each spectral band $\mathbf{x}_n$ ($n \in [1, N]$) is correlated to is left and right neighbors. Thus, we calculate the number of bands $d_{left}(\mathbf{x}_n)$ located at the left side of $\mathbf{x}_n$ that are similar to it (i.e., with a VIF value greater than a threshold θ). Similarly, we calculate the number of bands $d_{right}(\mathbf{x}_n)$ located at the right side of $\mathbf{x}_n$ that are similar to it (see Algorithm 5.2). From this, we assumed that all the bands within the interval $[\mathbf{x}_{n-d_{left}(\mathbf{x}_n)+1}, \mathbf{x}_{n+d_{right}(\mathbf{x}_n)-1}]$ are similar to $\mathbf{x}_n$ and form a possible cluster.

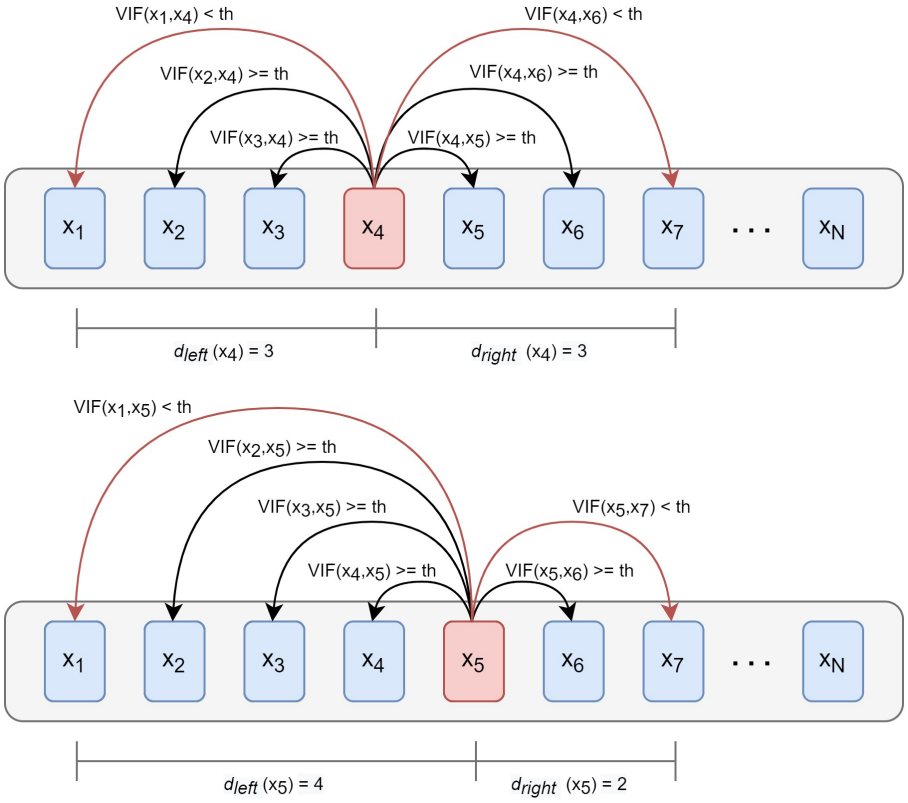

Figure 5.1: Example of the calculation of distances d_{left} and d_{right} for bands $\mathbf{x}_4$ and $\mathbf{x}_5$ used in the IBRA process. Note that the value of $\text{VIF}(\mathbf{x}_4, \mathbf{x}_5)$ is calculated when analyzing the variable $\mathbf{x}_4$ so that this value is simply reused in the next iteration.

We consider that a suitable cluster center should represent the bands that are on the left side of the cluster in a similar way that it represents those on the right side; therefore, we calculate the difference $d(\mathbf{x}_n) = |d_{left}(\mathbf{x}_n) - d_{right}(\mathbf{x}_n)|$ to quantify the suitability of band $\mathbf{x}_n$ to become a cluster center, where low values of $d(\mathbf{x}_n)$ indicate better suitability. Take Fig. 5.1 as an example. Here $\mathbf{x}_4$ ($d(\mathbf{x}_4) = 0$) is more suitable than $\mathbf{x}_5$ ($d(\mathbf{x}_5) = 2$) to be considered a bandwidth center because it is similar to both the left and right sides, so it better represents the bands within the interval $[\mathbf{x}_2, \mathbf{x}_6]$. Recall that there exists a strong correlation between bands with close wavelengths [114], which means that the more distant two bands are, the more different their spectral information is likely to be.

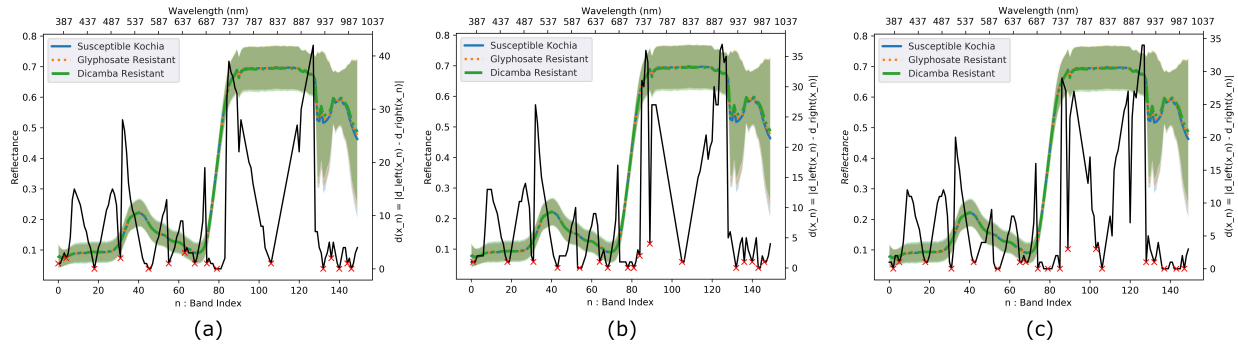

Figure 5.2: Spectral response and Spectral index vs. distance d plots for the Kochia dataset using different VIF thresholds. **(a)** $\theta = 8$ **(b)** $\theta = 10$. **(c)** $\theta = 12$. Local minima in the three graphs are indicated with an '$\times$'.

Next, we analyze how the differences $d(\mathbf{x}_n)$ are distributed across the spectrum. This distribution is used to find clusters with similar bands and their corresponding cluster centers. Fig. 5.2 shows the distribution of the variable $d(\mathbf{x}_n)$ for the Kochia dataset given different thresholds θ. From this, we observe that each distribution consists of a series of "V" patterns. In this context, a local minimum—the center of a "V" pattern—represents a salient band that explains the variance within the dataset in a way similar to its neighbors. Even though all the bands within a "V" pattern are similar, the band at the leftmost position is more similar to those bands on the left side of the cluster. Similarly, the band at the rightmost position of the "V" is more similar to those bands on the right side of the cluster, while the band corresponding to the local minimum is similar to both sides, acting as a centroid. In general, we keep the bands corresponding to a local minimum in the n vs. $d(\mathbf{x}_n)$ plot and remove the rest since they are assumed to be redundant.

The entire IBRA process is described in Algorithm 5.2. Here, `getVIF`(a, b) calculates the VIF value between bands $\mathbf{x}_a$ and $\mathbf{x}_b$. Note that `getVIF`(a, b) = `getVIF`(b, a); therefore, in order to avoid re-calculations and make the algorithm more efficient, we construct a symmetric matrix `table` that stores the pre-computed VIF values between pairs of bands. Furthermore, `getLocalMinima`(d) is a function that retrieves the position of the local minima

of the vector d. Hence, the algorithm returns a vector called `preselection`, which is a set of numbers $S_c \in \mathbb{Z}^{N'}$ (where $N' \ll N$) representing the indices of the pre-selected spectral bands. It is worth mentioning that the number of pre-selected bands N' is not known *a priori* and depends on each dataset and the selected VIF threshold θ.

5.2.3 Band Selection Using Pre-Selected Bands

Our second contribution is the development of a two-step hybrid band selection method: First, it uses IBRA to obtain a set of independent candidate bands from the original spectrum, as we showed in Sec. 5.2.2. Then, it employs a wrapper-based method to select the best combination of bands given the desired number of bands k (whose indices are denoted by $S_f \in \mathbb{Z}^k$) from the set of N' available candidate bands pre-selected by IBRA (whose indices are denoted by $S_c \in \mathbb{Z}^{N'}$) instead of the entire set of spectral bands, which greatly reduces the search complexity. We call this process Greedy Spectral Selection (GSS).

The GSS process starts by ranking each candidate band $\mathbf{x}_c$, where $c \in S_c$, according to some criterion. In this work, we use information entropy to calculate an initial relevance score for each spectral band. The three datasets used in this work were acquired using a bit depth of 14 bits, allowing us to interpret a spectral band as a discrete random variable. The entropy of band $\mathbf{x}_c$, $H(\mathbf{x}_c)$, conveys the average level of information inherent in $\mathbf{x}_c$ and is defined as follows:

$$H(\mathbf{x}_c) = -\sum_{z \in \Omega_{\mathbf{x}_c}} P(z) \log P(z),$$

where $P(z)$ is the probability mass function of random variable z, and $\Omega_{\mathbf{x}_c}$ is the space that encompasses all the possible values that can occur in the spectral band $\mathbf{x}_c$.

Here, it is important to highlight the difference of our method with respect to other band selection methods that use information entropy as a ranking measure, such as those proposed by Wang *et al.* [112, 114]. In particular, those methods use an optimization approach to group the data into k clusters of similar bands. Then they use information entropy to find

the most representative band from each cluster (i.e., the band with the largest amount of information) to obtain a final selection of k bands. Our approach is distinct in two aspects: First, when using IBRA, we already picked the most representative band from each cluster (defined as a local minimum in the n vs. $d(\mathbf{x}_n)$ plot, as shown in Fig. 5.2). Then we rank the N' pre-selected bands based on information entropy in order to select a subset S_f of k bands. Thus, initially, S_f consists of the indices of the top-k candidate bands of S_c with the greatest entropy. The second aspect is that, while other methods use information entropy to obtain a final selection of bands directly, we use it to obtain an initial selection that will be refined further using a greedy algorithm based on a multicollinearity analysis.

We note that collinearity does not exist between pairs of candidate bands obtained by the IBRA pre-selection process because the calculated VIF value between two cluster centers must be lower than a given threshold θ. However, it is possible that three or more bands are highly correlated—a phenomenon known as multicollinearity [6]. In that case, we check for the presence of multicollinearity within the k selected spectral bands the same way as described in Sec. 5.2.2 with collinearity; that is, taking one band at a time, considering it as the dependent variable while considering the rest of the $k-1$ bands as the independent variables, and calculating its corresponding VIF value.

With this, after obtaining an initial set of k selected bands based on information entropy, GSS employs the following greedy algorithm (see Algorithm 5.3). First, we calculate the average classification performance when using the current k selected bands. We chose to use the F1 score to describe the performance of the classifier because it conveys the balance between the precision and the recall. To do this, we perform a 5×2-fold cross-validation using a Hyper3DNetLite network (see Sec. 5.2.1) and calculate the average F1 score obtained on the 10 validation folds. Then, after calculating the VIF value for each of the currently selected bands, we remove the one with the greatest VIF value and consider the next available band with the greatest entropy. With this new subset of k bands, we perform a new 5×2-fold

Algorithm 5.3 Greedy spectral selection

```
1:  function selectBands(S_c, k)
2:      H ← getEntropy(S_c)
3:      S_c.sort(key = H)
4:      S_f ← S_c[1 : k]
5:      S_c ← S_c[k + 1 : end]
6:      F1 ← trainSelection(S_f)
7:      bestS_f ← S_f
8:      while length(c) > 0 do
9:          listVIF ← getVIFMulti(S_f)
10:         index = getMax(listVIF)
11:         S_f[index : end] ← S_f[index + 1 : end]
12:         S_f.append(S_c[1])
13:         S_c ← S_c[2 : end]
14:         newF1 ← trainSelection(S_f)
15:         if newF > F1 then
16:             bestS_f ← S_f
17:     return bestS_f
```

cross-validation process to determine if the average classification performance has improved. If so, we consider the current selection of k bands as the current best selection and update the indices of S_f. We repeat this process until there are no more available bands. Finally, we select the combination of bands that showed the best classification performance.

The type of classifier we used for the GSS process to evaluate the appropriateness of a given selection of spectral bands (line 6 in Algorithm 5.3) is a CNN (Hyper3DNetLite). Preliminarily, we experimented with other types of classifiers to use in the GSS process; specifically, support vector machines, random forests, and feedforward neural networks. However, these approaches showed slow convergence rates and poor classification performance in comparison to our CNN, despite the CNN being a more complex model. For this reason, we continued to use CNNs over the other types of classifiers.

Fig. 5.3 depicts the process of selecting $k = 6$ bands for the Kochia dataset from the set of 17 band candidates obtained using IBRA with a VIF threshold of 10 (see Fig. 5.2.b). In

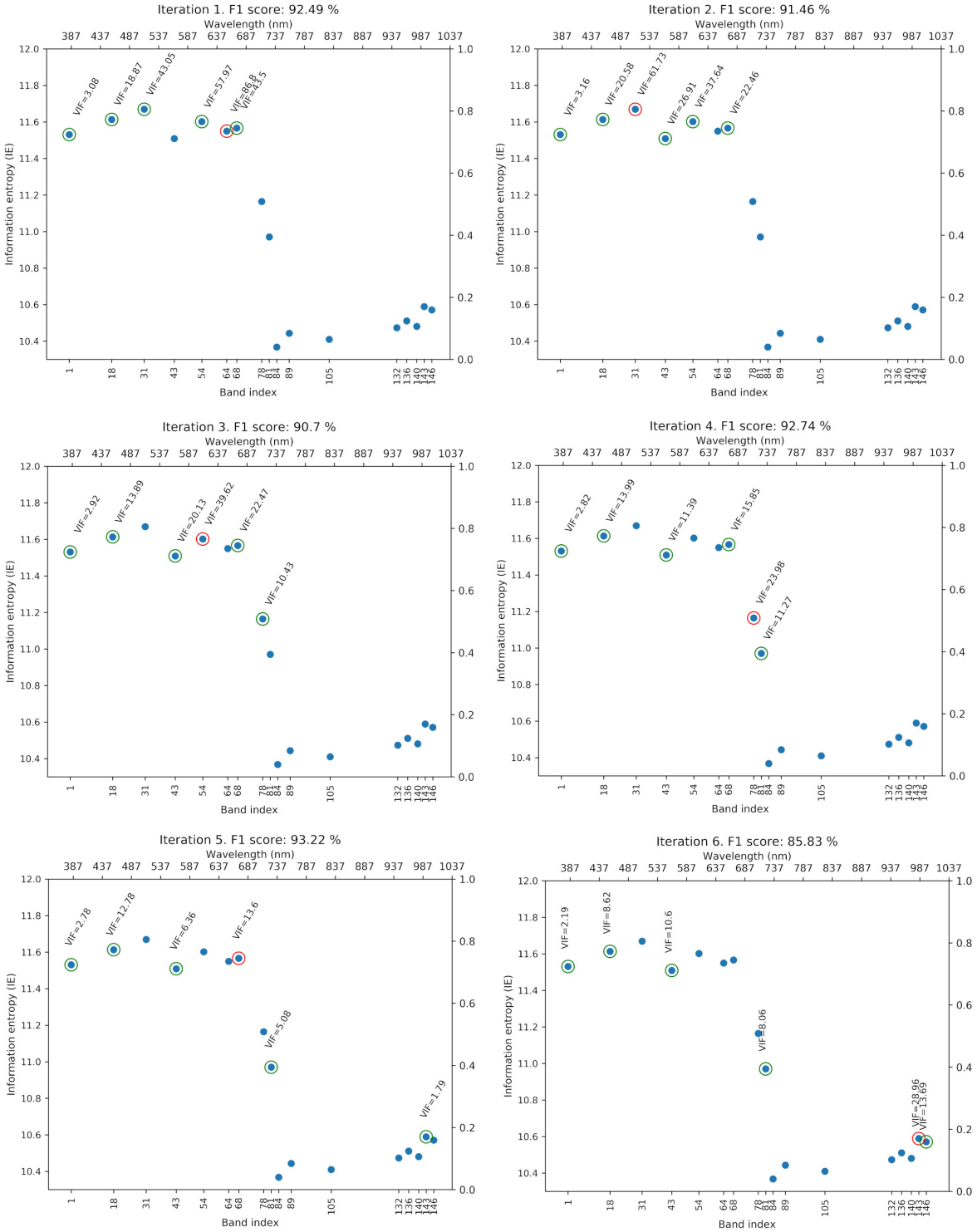

Figure 5.3: Demonstration of the Greedy Spectral Selection (GSS) process for the Kochia dataset using a VIF threshold of 10 and $k = 6$.

the first iteration of GSS, the first six bands selected are those with the highest information entropy values; for example, those with indices $[1, 18, 31, 54, 64, 68]$. With this selection of bands, we perform 5×2-fold cross-validation and obtain an average F1 score of 92.49%. Then, we determine that the band with index 64 has the highest VIF value (i.e., 86.8), which means that its variance can be explained well using the other five bands. For this reason, this band is removed from the selection and the next available band (i.e., the band with index 43) is considered in the second iteration. We only show the results obtained after six iterations because the following iterations could not achieve better classification performance scores. Therefore, the best selected bands are found at the fifth iteration and their indices are $[1, 18, 43, 68, 81, 143]$.

The GSS algorithm is presented in Algorithm 5.3. Note that it selects the indices of the best k bands from a list of candidate band indices S_c. Function `getEntropy(`S_c`)` returns the entropy value of each of the candidate bands with indices S_c. Then, S_c`.sort(key=`H`)` sorts the elements of S_c with respect to their corresponding entropy values in non-increasing order. Function `getVIFMulti(`S_f`)` returns a list with the VIF value for each band in S_f. Next `getMax(`l`)` returns the position where the maximum value in a list l was found. Finally, the performance of the currently selected bands is evaluated using `trainSelection(`S_f`)`, which returns the average F1 score obtained using the bands with indices S_f and 5×2-fold cross-validation.

5.2.4 Multispectral Imager Design

A spectral band is determined by its wavelength center and its spectral resolution; that is, an imager captures the reflected light in a given band around its center wavelength within a range determined by its spectral resolution. The method described in Sec. 5.2.3 is used to select the most relevant spectral bands from the original hyperspectral data cube and, as a consequence, their wavelengths. Knowing which wavelengths are more useful for

a given application allows us to discard the information captured at different wavelengths, thus reducing storage requirements. However, this knowledge also allows for the design of a compact multispectral imager that would be used instead of a full hyperspectral imager. Hence, to test our hypothesis that we can use our method to provide the specifications needed to design a multispectral imager, we assess classification performance with a simulated multispectral imager by using the original hyperspectral data cube and simulate applying optical filters to capture data from an imager that would use our selected wavelengths.

The bandwidths of the simulated multispectral filters are set to be equivalent to bandwidths encompassed by five bands, which is equivalent to 21 nm for the case of the Kochia dataset, and 50 nm for the case of the IP and SA datasets. These are commonly available optical filter bandwidths. To obtain the reflectance measured by the simulated multispectral sensor, we generate k Gaussian distributions, one for each of the selected bands in S_f, taking the spectral wavelengths selected by a band selection method as the centroids. The spread of the generated Gaussian curves is proportional to the filter bandwidth. In particular, the full-width at half-maximum (FWHM) of the curve is set to be equal to the bandwidth [111]. The FWHM is defined based on the standard deviation of the curve σ as $\text{FWHM} = 2\sigma\sqrt{2\ln 2}$.

Hence, the Gaussian distribution generated for the i-th band ($i \in S_f$) is defined as:

$$g_i(w) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{(w - p(i))^2}{2\sigma^2}\right\},$$

where $w \in \mathbb{R}$, and $p(i)$ denotes the wavelength of the i-th band. Then the simulated multispectral reflectance measurement is obtained by multiplying the original hyperspectral data cube by the corresponding Gaussian distribution generated for each band, then integrating under the resulting response curve to get a single reflectance value. Consider that the reflectance of a single data point is determined by the function $r(w)$ ($w \in [0, N]$),

the simulated reflectance measurement of the i-th band, $s(i)$, is calculated as follows:

$$s(i) = \int_{i-\mathrm{FWHM}/2}^{i+\mathrm{FWHM}/2} r(w) g_i(w)\, dw.$$

This process is repeated for each of the k Gaussian distributions. Fig. 5.4 shows an example of a multispectral filter simulation using the fourth band as the filter centroid.

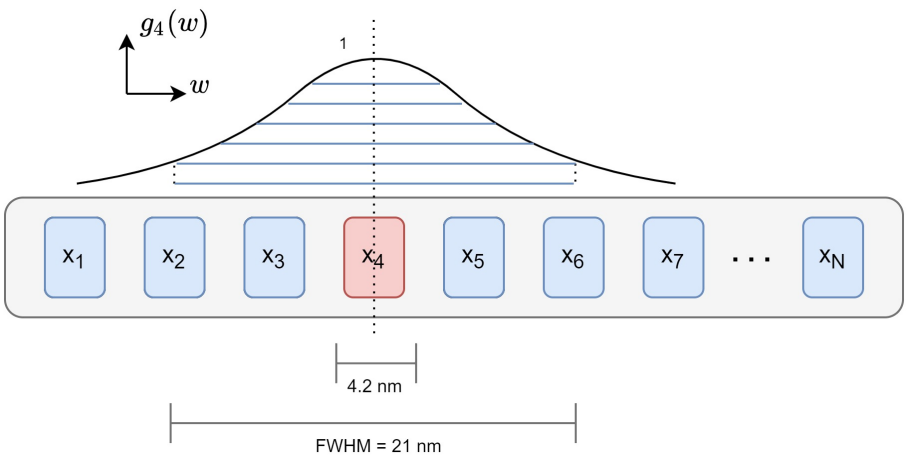

Figure 5.4: Example of a multispectral filter simulation.

5.2.5 Feature Extraction Framework

Feature extraction approaches project the N original spectral bands into a new representation basis of N'' features (where $N'' < N$). As such, transformed features can no longer be interpreted as spectral bands. Therefore, we lose capability to design multispectral filters as explained in Sec. 5.2.4. However, in some cases, feature extraction methods may present some performance advantages over feature selection methods. For example, projecting the data into a new basis with reduced dimensionality based on PCA helps to remove redundancy while trying to explain the original variance of the dataset. Conversely, feature selection methods simply discard the information of the bands that were not selected.

Therefore, a classifier trained on N'' transformed features that explain most of the original variance of the dataset might perform better than a classifier trained on N'' selected bands, considering that the discarded bands did not contain trivial information.

We propose generalizing our method described above into a two-stage feature extraction framework. First, we apply IBRA to remove redundant bands; then, we can use any feature extraction method to obtain the desired number of feature channels k. In this work, we demonstrate the pipeline using two types of feature extraction methods: unsupervised and supervised. We chose PCA as the unsupervised method and denote the resulting instance as IBRA-PCA. In addition, we use PLS-DA as the supervised method; here, we use the class labels of the dataset to maximize the class separation within the reduced space. This instance is denoted as IBRA-PLS-DA.

5.3 Results

In this section, we present the experimental results for each step of our proposed methods and analyze their performance in comparison to other dimensionality reduction methods. For the sake of consistency and fairness, we used the same configuration for all the networks trained in our experiments, including those used during the GSS process; that is, we used the same network architecture (i.e., Hyper3DNet-Lite), optimizer, and batch-size. While this strategy does not guarantee the best possible results, it allows us to compare the behavior of different band selection methods under the same conditions. All CNNs were trained using the Adadelta optimizer [122], which is a gradient descent method based on an adaptive learning rate so that there is no need to select a global learning rate manually. The mini-batch size was set to 128. The last layer of the CNNs used a softmax activation function, and we employed a categorical cross-entropy loss function.

We applied 5×2-fold stratified cross-validation to train and evaluate all networks. That is, we randomly divided our datasets into two equally sized folds so that one fold works

as the training set and the other as the validation set, train again with the roles of these folds reversed, and then repeat this process four more times. Furthermore, stratification indicates that each fold has the same proportion of samples of a given class. Note that z-score normalization (mean equal to 0 and standard deviation equal to 1) was applied onto each spectral band of each training set while the exact same scaling was applied to their corresponding validation set. In order to analyze the behavior of our models, we calculated four metrics by aggregating the results on all ten of the validation sets: overall classification accuracy (OA), macro-average precision ($Prec$), macro-average recall (Rec), and macro-average F1 score. The source code and datasets are available online[1].

5.3.1 Training with Pre-Selected Bands Alone

Previously (Fig. 5.2) we showed some examples of applying the pre-selection method using IBRA on the Kochia dataset using three different VIF thresholds (12, 10, and 8), which reduced our search space from 150 bands to 19, 17, and 16 bands, respectively. Table 5.2 gives the number of pre-selected bands for Kochia, IP, and SA datasets when using IBRA with a VIF threshold of 10 ($\theta = 10$); it also gives the average performance for the four metrics and corresponding standard deviations using the Hyper3DNet-Lite network when training on the full hyperspectral spectrum and only the pre-selected bands. The number of parameters required to train each network is reported in the last column.

5.3.2 Greedy Spectral Selection

From Fig. 5.2, we notice that different VIF thresholds can lead to slightly different selections of candidate bands. In general, the lower the VIF threshold, the more distinct two bands have to be to be considered non-collinear. Exactly how large this threshold needs to be is not clear. If it is too large, we may end up including too many candidate bands

[1]Codebase: `https://github.com/GiorgioMorales/HSI-BandSelection.git`.

Dataset	# Bands	OA	Prec	Rec	F1	# Param.
Kochia	150	98.46 ± 0.29	98.66 ± 0.26	98.55 ± 0.31	98.60 ± 0.28	561,475
	17	97.05 ± 0.47	97.25 ± 0.45	97.17 ± 0.46	97.21 ± 0.44	258,035
Indian Pines	200	99.42 ± 0.18	99.32 ± 0.29	99.47 ± 0.28	99.39 ± 0.27	1,274,464
	31	99.49 ± 0.14	99.38 ± 0.34	99.56 ± 0.19	99.47 ± 0.23	338,880
Salinas	204	99.92 ± 0.04	99.93 ± 0.05	99.94 ± 0.03	99.94 ± 0.04	1,296,608
	14	99.43 ± 0.13	99.75 ± 0.08	99.72 ± 0.08	99.73 ± 0.08	244,768

Table 5.2: Performance with and without IBRA preselection ($\theta = 10$).

that do not represent suitable bandwidth centers. On the other hand, if it is too small, we may discard bands that could have been useful for the classification task. For that reason, we applied IBRA using different VIF thresholds $\theta \in [5, 12]$ so that we would obtain a set of pre-selected bands for each of them. Then, we applied GSS on each set of candidates to select the desired number of bands k. Finally, we selected the classifier that achieved the best classification performance based on the mean F1-score obtained after a 5×2-fold cross-validation.

For the Kochia dataset, we considered six, eight, and ten bands in order to evaluate the trade-off between the number of bands and classification performance. For the IP and SA datasets, we selected only five bands. This is because we found that the ease of training these datasets when using $k > 5$ yielded high-performance metrics ($> 99\%$ accuracy) no matter which feature selection method is chosen. That is, the lack of sufficiently distinct

VIF	Selected bands (nm)	OA	Prec	Rec	F1
12	[395.5, 463.3, 565.1, 700.8, 722.0 , 993.3]	92.44 ± 0.71	92.76 ± 0.80	92.79 ± 0.67	92.76 ± 0.72
11	[395.5, 408.2, 463.3, 586.3, 662.6, 700.8]	90.74 ± 1.05	91.56 ± 0.97	91.54 ± 1.06	91.54 ± 1.01
10	**[391.2, 463.3, 569.3, 675.3, 730.4, 993.3]**	92.69 ± 0.53	93.24 ± 0.52	93.08 ± 0.49	**93.15 ± 0.49**
9	[391.2, 463.3, 569.3, 700.8, 730.4, 993.3]	92.40 ± 0.63	92.67 ± 0.63	92.77 ± 0.59	92.71 ± 0.59
8	[387.0, 404.0 , 463.3, 577.8, 700.8, 722.0]	92.58 ± 0.63	93.05 ± 0.65	93.08 ± 0.57	93.06 ± 0.59
7	[387.0, 404.0, 463.3, 569.3, 700.8, 722.0]	92.07 ± 0.89	92.52 ± 0.89	92.55 ± 0.79	92.53 ± 0.83
6	[387.0, 404.0 , 463.3, 586.3, 700.8, 717.7]	92.00 ± 0.61	92.57 ± 0.54	92.52 ± 0.64	92.53 ± 0.57
5	[387.0, 463.3, 586.3, 645.6, 700.8, 722.0]	91.03 ± 1.04	91.79 ± 1.14	91.75 ± 0.91	91.76 ± 1.01

Table 5.3: Greedy Spectral Selection on the Kochia dataset ($k = 6$).

results prevented us from being able to evaluate the existence of any performance difference among the various methods, which is why selecting more than five bands from these datasets was not considered appropriate for this work. In addition, for each dataset, we experimented with different dataset sizes (i.e. 100%, 75%, 50%, and 25%) to evaluate how consistent the band selection results as data availability changes.

For the Kochia dataset, when selecting six bands ($k = 6$), the best results were obtained using a VIF threshold of ten ($\theta = 10$), and the wavelengths of the selected bands (in nm) were $[391.2, 463.3, 569.3, 675.3, 730.4, 993.3]$ for each of the four dataset size variations. Equivalently, the indices of the selected bands were $[1, 18, 43, 68, 81, 143]$. Table 5.3 shows

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[395.5, 408.2 , 463.3, 565.1, 675.3, 700.8, 722.0, 993.3]	93.96 ± 0.68	94.47 ± 0.65	94.34 ± 0.68	94.39 ± 0.66
11	[395.5, 408.2 , 463.3, 565.1, 675.3, 700.8, 726.2, 993.3]	94.18 ± 0.70	94.74 ± 0.65	94.47 ± 0.67	94.59 ± 0.66
10	[391.2, 463.3, 569.3, 675.3, 730.4, 963.6, 993.3, 1006.0]	93.95 ± 0.62	94.48 ± 0.56	94.13 ± 0.68	94.29 ± 0.61
9	[391.2, 463.3, 569.3, 671.1, 700.8, 730.4, 963.6, 993.3]	94.33 ± 0.41	94.81 ± 0.48	94.60 ± 0.47	94.69 ± 0.47
8	**[387.0, 404.0, 463.3, 518.4, 577.8 , 654.1, 675.32, 700.8]**	94.79 ± 0.65	95.28 ± 0.55	95.20 ± 0.67	**95.23 ± 0.61**
7	[387.0, 404.0, 463.3, 569.3, 675.3, 700.8, 722.0, 1006.0]	94.48 ± 0.77	95.05 ± 0.72	94.85 ± 0.73	94.93 ± 0.73
6	[387.0, 404.0, 463.3, 586.3, 679.6, 700.8, 730.4, 1001.8]	94.65 ± 0.45	95.21 ± 0.38	94.89 ± 0.48	95.04 ± 0.42
5	[387.0, 463.3, 586.3, 645.6, 700.8, 722.0, 980.6, 1001.8]	93.68 ± 0.75	94.25 ± 0.65	93.95 ± 0.68	94.08 ± 0.66

Table 5.4: Greedy Spectral Selection on the Kochia dataset ($k = 8$).

the performance using IBRA and GSS on the full Kochia dataset for different VIF thresholds ($\theta \in [5, 12]$) when selecting six bands.

Table 5.4 shows the performance using IBRA and GSS on the full Kochia dataset for different VIF thresholds when selecting eight bands ($k = 8$). In that case, the best results were obtained using a VIF threshold of $\theta = 8$ when using 100%, 50%, and 25% of the dataset, and $\theta = 6$ when using 75% of the dataset (see Appendix A.1). In the four cases, the difference between the classification performance obtained using $\theta = 8$, $\theta = 7$, and $\theta = 6$ was not statistically significant, so we consider the three options equally as good. Therefore,

VIF	Selected bands (nm)	OA	Prec	Rec	F1
12	[395.5, 408.2 , 463.3, 518.4, 565.1, 616.0, 675.3, 700.8, 722.0, 993.3]	96.31 ± 0.69	96.57 ± 0.55	96.49 ± 0.73	96.53 ± 0.64
11	[395.5, 408.2, 463.3, 565.1, 662.6, 675.3, 700.8, 713.5, 726.2, 993.3]	96.18 ± 0.41	96.48 ± 0.29	96.31 ± 0.46	96.39 ± 0.36
10	[391.2, 463.3, 518.4, 569.3, 658.4, 675.3, 717.7, 730.4, 993.3, 1006.]	95.83 ± 0.36	96.10 ± 0.38	96.06 ± 0.32	96.08 ± 0.34
9	[391.2, 463.3, 518.4, 569.3, 616.0, 671.1, 700.8, 717.7, 730.4, 993.3]	96.16 ± 0.56	96.48 ± 0.50	96.37 ± 0.54	96.42 ± 0.52
8	[387.0, 404.0, 463.3, 518.4, 577.8, 654.1, 675.3, 700.8, 722.0, 1006.0]	96.47 ± 0.38	96.79 ± 0.36	96.66 ± 0.37	96.72 ± 0.35
7	**[387.0, 404.0, 463.3, 518.4, 569.3, 654.1, 675.3, 700.8, 722.0, 1006.0]**	96.69 ± 0.35	96.92 ± 0.38	96.95 ± 0.34	**96.93 ± 0.35**
6	[387.0, 404.0, 463.3, 586.3, 649.9, 679.6, 700.8, 717.7, 730.4, 1001.8]	95.91 ± 0.50	96.34 ± 0.44	96.12 ± 0.47	96.22 ± 0.45
5	[387.0, 463.3, 586.3, 645.6, 700.8, 722.0, 832.2, 946.7, 980.6, 1001.8]	95.06 ± 0.54	95.44 ± 0.52	95.33 ± 0.56	95.38 ± 0.53

Table 5.5: Greedy Spectral Selection on the Kochia dataset ($k = 10$).

in this work, we simply chose the bands selected using $\theta = 8$ since it achieves slightly higher performance metrics in three out of the four dataset variations. Hence, the wavelengths of the selected bands were $[387.0, 404.0, 463.3, 518.4, 577.8, 654.1,\ 675.32, 700.8]$ (nm) with indices $[0, 4, 18, 31, 45, 63, 68,\ 74]$.

Table 5.5 shows the performance using IBRA and GSS on the full Kochia dataset for different VIF thresholds when selecting ten bands ($k = 10$). In that case, the best results were obtained using a VIF threshold of $\theta = 7$ when using 100% and 50% of the dataset, and $\theta = 8$ when using 75% and 25% of the dataset. The wavelengths of the bands selected

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[498.3, 646.7, 706.2, 754.4, 1023.7]	97.96 ± 0.33	98.21 ± 0.43	98.32 ± 0.33	98.25 ± 0.35
11	[557.5, 587.2, 706.2, 754.4, 1023.7]	97.55 ± 0.29	98.05 ± 0.29	97.95 ± 0.29	97.98 ± 0.22
10	**[498.3, 626.9, 706.2, 754.4, 1023.7]**	98.08 ± 0.43	98.26 ± 0.42	98.39 ± 0.43	**98.32 ± 0.39**
9	[557.5, 626.9, 706.2, 821.8, 1023.7]	98.28 ± 0.35	98.24 ± 0.47	98.06 ± 0.59	98.11 ± 0.43
8	[607.0, 646.7, 706.2, 821.8, 1023.7]	98.04 ± 0.30	98.19 ± 0.46	98.06 ± 0.46	98.10 ± 0.35
7	[567.4, 587.2, 706.2, 821.8, 1023.7]	98.01 ± 0.24	98.29 ± 0.18	98.36 ± 0.42	98.31 ± 0.26
6	[577.3, 706.2, 821.8, 918.0, 1023.7]	97.06 ± 0.49	97.49 ± 0.58	97.63 ± 0.48	97.53 ± 0.48
5	[577.3, 715.8, 812.2, 918.0, 1023.7]	96.73 ± 0.55	97.46 ± 0.53	97.00 ± 0.61	97.19 ± 0.49

Table 5.6: Greedy Spectral Selection on the Indian Pines dataset.

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[532.1, 731.8, 950.5, 1368.9, 2224.7]	98.70 ± 0.11	99.40 ± 0.05	99.48 ± 0.04	99.44 ± 0.04
11	[557.5, 731.8, 950.5, 1368.9, 2224.7]	98.68 ± 0.11	99.36 ± 0.10	99.45 ± 0.05	99.40 ± 0.07
10	[557.5, 731.8, 950.5, 1359.4, 2224.7]	98.62 ± 0.14	99.38 ± 0.07	99.45 ± 0.04	99.41 ± 0.05
9	[557.5, 731.8, 950.5, 1359.4, 2224.7]	98.62 ± 0.14	99.38 ± 0.07	99.45 ± 0.04	99.41 ± 0.05
8	**[731.8, 950.5, 1159.7, 1254.8, 2224.7]**	99.05 ± 0.05	99.56 ± 0.04	99.55 ± 0.03	**99.56 ± 0.03**
7	[731.8, 950.5, 1159.7, 1254.8, 2215.2]	99.02 ± 0.10	99.54 ± 0.07	99.53 ± 0.06	99.54 ± 0.06
6	[587.2, 950.5, 1245.3, 1368.9, 2215.2]	99.02 ± 0.05	99.54 ± 0.04	99.56 ± 0.02	99.55 ± 0.03
5	[577.3, 950.5, 1245.3, 1368.9, 2215.2]	98.99 ± 0.06	99.52 ± 0.09	99.56 ± 0.05	99.54 ± 0.07

Table 5.7: Greedy Spectral Selection on the Salinas dataset.

for $\theta = 7$ were $[387.0, 404.0, 463.3, 518.4,\ 569.3, 654.1, 675.3, 700.8, 722.0, 1006.0]$ (nm) with indices $[0, 4, 18, 31, 43, 63, 68, 74, 79, 146]$ and the only difference with respect to the bands selected for $\theta = 8$ was the selection of the wavelength 577.8 nm that corresponds to the band index 45 instead of 569.3 nm, which corresponds to the band index 43. The bold entries in these tables represent the best VIF threshold and average F1 score.

For the IP dataset, the best results were obtained using a VIF threshold of ten ($\theta = 10$) and the wavelength of the selected bands were $[498.3, 626.9, 706.2, 821.8, 1023.7]$ (i.e., with band indices $[11, 25, 34, 39, 67]$). These bands are selected when using the four dataset size variations except when we used 50% of the dataset, where the best VIF threshold was nine ($\theta = 9$) and the wavelength 557.5 nm (band index 46) was selected instead of the wavelength 498.3 nm (band index 39), as shown in Appendix A.2. Table 5.6 shows the performance using IBRA and GSS on the full IP dataset.

For the SA dataset, the best classification performance was obtained using a VIF threshold of $\theta = 8$ when using 100% and 75% of the dataset. The wavelength of the selected bands were $[731.83, 950.54, 1159.73, 1254.82,\ 2044.73]$ (nm), that correspond to the indices $[37, 60, 82,\ 92, 175]$ of the corrected SA dataset after discarding the 20 water absorption bands (see Sec. 3.1.1). When we reduced the dataset size, different VIF thresholds were selected. Specifically, when we used 50% of the dataset, the best VIF threshold was six ($\theta = 6$) and we selected the wavelengths 570.18 nm (index 20) instead of 731.83 nm (index 37); 1245.31 nm (index 91) instead of 1254.82 nm (index 92); and 2215.22 nm (index 174) instead of 2224.73 nm (index 175) (see Appendix A.3). Furthermore, when we used 25% of the dataset, the best VIF threshold was seven ($\theta = 7$) and we selected the wavelength 2215.22 nm (index 174) instead of 2224.73 nm (index 175). Similarly to what happened with the Kochia dataset when selecting eight bands, the difference in classification performance obtained using $\theta = 8$, $\theta = 7$, and $\theta = 6$ is not statistically significant, so we consider the three options equally as good; however, in this work, we chose to use the bands selected

using $\theta = 8$, as it yielded slightly better performance metrics in two out the four dataset variations. Table 5.7 shows the performance using IBRA and GSS on the full SA dataset.

5.3.3 Comparative Results

We compared our IBRA-GSS method to four other band selection methods: OCF [114], HAGRID [111], SR-SSIM [119], FNGBS [112], and PLS-DA [66]. For OCF, we used the normalized cut criterion as the objective function along with information entropy as the ranking method, given that this setting showed the best performance. For HAGRID, we used a grid search to choose the following hyperparameters: a crossover rate of 0.25, a mutation rate of 0.05, a tournament size of 5, a population size of 1,000, and 300 generations. For SR-SSIM, we used a window size of 3 for the SSIM index algorithm to construct a hyperspectral similarity matrix, as it achieved the best performance. For FNGBS, to calculate the number of shared neighbor elements between two bands, we considered the nearest neighbor of two elements. Finally, we adapted PLS-DA for feature selection by sorting the coefficients obtained by PLS-DA in order to select the top-k bands. We also compared our IBRA-GSS method to our IBRA-PCA and IBRA-PLS-DA methods to analyze the behavior of feature extraction methods in front of a feature selection method.

To assess the effectiveness of the dimensionality reduction methods, we compared the performance of eight Hyper3DNet-Lite CNNs, each trained on the features selected or extracted by the eight methods. This comparison was carried out using the same network architecture, hyperparameters, and other configurations for all of the methods. Finally, to determine if the difference in performance scores was statistically significant, we performed a paired t-test and a paired permutation test using the F1 scores at the $\alpha = 0.05$ level.

The method comparison is shown in Tables 5.8–5.10 for the Kochia dataset, Table 5.11 for the IP dataset, and Table 5.12 for the SA dataset, with the best performing metrics highlighted in bold. Here, the first row of each method represents the results obtained after

Method	OA	Prec	Rec	F1
FNGBS	84.32 ± 1.78	84.85 ± 1.77	84.37 ± 1.72	84.59 ± 1.73
	86.98 ± 0.84	87.35 ± 0.80	86.91 ± 0.91	87.10 ± 0.83
PLS-DA	84.77 ± 1.83	85.15 ± 1.89	84.69 ± 1.82	84.89 ± 1.82
	88.41 ± 0.79	88.85 ± 0.62	88.37 ± 0.96	88.59 ± 0.78
SR-SSIM	86.42 ± 1.07	87.66 ± 1.02	87.31 ± 1.06	87.47 ± 1.03
	88.73 ± 0.90	89.55 ± 0.81	89.30 ± 0.99	89.41 ± 0.89
OCF	90.48 ± 0.57	90.92 ± 0.62	90.81 ± 0.44	90.86 ± 0.49
	92.42 ± 0.67	92.75 ± 0.66	92.66 ± 0.66	92.70 ± 0.65
HAGRID	91.71 ± 0.83	92.25 ± 0.78	92.17 ± 0.84	92.20 ± 0.80
	92.48 ± 0.62	92.91 ± 0.53	92.89 ± 0.58	92.89 ± 0.54
IBRA-PCA	83.04 ± 2.23	83.44 ± 2.16	82.96 ± 2.40	83.17 ± 2.29
	91.31 ± 0.74	91.81 ± 0.67	91.18 ± 0.71	91.46 ± 0.68
IBRA-PLS-DA	90.81 ± 0.96	91.68 ± 1.04	91.33 ± 0.86	91.48 ± 0.94
	92.29 ± 0.63	92.98 ± 0.58	92.57 ± 0.72	92.76 ± 0.65
IBRA-GSS	**92.69 ± 0.53**	**93.24 ± 0.52**	**93.08 ± 0.50**	**93.15 ± 0.49**
	93.32 ± 0.68	**93.80 ± 0.64**	**93.74 ± 0.66**	**93.76 ± 0.64**

Table 5.8: Feature selection method comparison — Kochia ($k = 6$).

training a model using the original selected bands (identified as "original band selection"), while the second row represents the results obtained after using simulated filters that take the position of the selected bands as their central wavelengths (identified as "multispectral filter simulation"), as explained in Sec. 5.2.4. The simulated filters used for the Kochia dataset were 20 nm while for the IP and SA datasets were 50 nm. Note that the feature channels extracted by IBRA-PCA and IBRA-PLS-DA cannot be interpreted as spectral bands, thus we cannot apply the "multispectral filter simulation" approach directly on them. Instead, we applied a Gaussian transformation on the IBRA-selected bands to simulate filters that take the position of the pre-selected bands as their central wavelengths; then, we used PCA or PLS-DA on the transformed bands to obtain the desired number of feature channels. In Appendixes B.1, B.2, and B.3 we show the method comparison for the Kochia, IP, and SA datasets, respectively, using 75%, 50%, and 25% of the dataset.

Additionally, Table 5.13 shows the resulting p-values of the paired t-tests and paired

Method	OA	Prec	Rec	F1
FNGBS	92.88 ± 0.55	93.63 ± 0.55	93.10 ± 0.46	93.34 ± 0.50
	93.12 ± 0.32	93.69 ± 0.28	93.13 ± 0.46	93.39 ± 0.34
PLS-DA	90.53 ± 1.65	91.70 ± 1.37	91.01 ± 1.72	91.32 ± 1.57
	91.92 ± 0.81	92.69 ± 0.71	92.21 ± 0.87	92.43 ± 0.79
SR-SSIM	91.01 ± 1.37	91.97 ± 1.28	91.54 ± 1.35	91.75 ± 1.31
	92.31 ± 0.83	93.00 ± 0.81	92.73 ± 0.89	92.86 ± 0.83
OCF	93.05 ± 0.90	93.86 ± 0.80	93.43 ± 0.84	93.62 ± 0.83
	92.89 ± 0.71	93.45 ± 0.64	93.21 ± 0.67	93.32 ± 0.65
HAGRID	91.59 ± 1.25	92.52 ± 1.05	91.88 ± 1.24	92.17 ± 1.16
	92.82 ± 0.79	93.41 ± 0.69	92.92 ± 0.91	93.15 ± 0.80
IBRA-PCA	83.75 ± 2.40	84.33 ± 2.48	83.77 ± 2.43	84.02 ± 2.45
	95.04 ± 0.70	95.35 ± 0.60	95.21 ± 0.81	95.27 ± 0.70
IBRA-PLS-DA	95.46 ± 0.53	95.91 ± 0.46	95.63 ± 0.49	95.76 ± 0.47
	95.42 ± 0.60	95.85 ± 0.52	95.62 ± 0.57	95.72 ± 0.53
IBRA-GSS	**94.79 ± 0.65**	**95.28 ± 0.55**	**95.20 ± 0.67**	**95.23 ± 0.61**
	94.08 ± 0.60	**94.67 ± 0.57**	**94.54 ± 0.54**	**94.60 ± 0.54**

Table 5.9: Feature selection method comparison — Kochia ($k = 8$).

Method	OA	Prec	Rec	F1
FNGBS	93.78 ± 0.77	94.17 ± 0.84	93.99 ± 0.73	94.08 ± 0.78
	94.19 ± 0.47	94.54 ± 0.47	94.27 ± 0.51	94.39 ± 0.48
PLS-DA	94.36 ± 0.51	94.86 ± 0.55	94.67 ± 0.47	94.76 ± 0.49
	95.10 ± 0.68	95.44 ± 0.59	95.18 ± 0.67	95.30 ± 0.63
OCF	94.87 ± 0.51	95.23 ± 0.52	95.11 ± 0.46	95.16 ± 0.47
	94.62 ± 0.73	95.00 ± 0.65	94.80 ± 0.64	94.89 ± 0.64
SR-SSIM	93.84 ± 0.62	94.50 ± 0.57	94.27 ± 0.64	94.37 ± 0.59
	94.36 ± 0.51	94.86 ± 0.55	94.67 ± 0.47	94.76 ± 0.50
HAGRID	94.50 ± 0.81	94.81 ± 0.78	94.69 ± 0.72	94.74 ± 0.74
	95.14 ± 0.51	95.49 ± 0.48	95.18 ± 0.51	95.33 ± 0.47
IBRA-PCA	95.69 ± 0.44	96.13 ± 0.44	95.92 ± 0.38	96.02 ± 0.39
	96.74 ± 0.30	96.96 ± 0.29	96.85 ± 0.28	96.91 ± 0.27
IBRA-PLS-DA	96.51 ± 0.46	96.78 ± 0.43	96.67 ± 0.44	96.72 ± 0.42
	96.82 ± 0.50	**97.08 ± 0.44**	**96.86 ± 0.54**	**96.97 ± 0.48**
IBRA-GSS	**96.69 ± 0.35**	**96.92 ± 0.38**	**96.95 ± 0.34**	**96.93 ± 0.35**
	96.21 ± 0.49	96.51 ± 0.45	96.40 ± 0.44	96.45 ± 0.44

Table 5.10: Feature selection method comparison — Kochia ($k = 10$).

Method	OA	Prec	Rec	F1
PLS-DA	96.68 ± 0.86	96.83 ± 0.99	95.62 ± 0.94	96.11 ± 0.74
	97.17 ± 0.60	97.30 ± 0.79	96.66 ± 1.03	96.90 ± 0.84
OCF	96.68 ± 0.56	97.34 ± 0.76	96.34 ± 0.98	96.77 ± 0.82
	97.02 ± 0.58	97.73 ± 0.51	97.14 ± 0.63	97.39 ± 0.48
HAGRID	96.74 ± 0.54	97.06 ± 0.75	96.34 ± 1.03	96.65 ± 0.88
	97.03 ± 0.75	97.24 ± 0.86	96.72 ± 1.45	96.91 ± 1.21
SR-SSIM	97.28 ± 0.37	97.74 ± 0.44	97.34 ± 0.80	97.49 ± 0.57
	96.83 ± 0.77	97.14 ± 1.02	96.82 ± 1.95	96.86 ± 1.47
FNGBS	97.49 ± 0.34	97.86 ± 0.36	97.64 ± 0.71	97.72 ± 0.5
	97.34 ± 0.65	97.94 ± 0.52	97.75 ± 0.46	97.82 ± 0.44
IBRA-PCA	94.70 ± 0.65	96.12 ± 0.71	94.45 ± 1.52	95.10 ± 1.02
	96.99 ± 0.97	97.84 ± 0.77	97.60 ± 0.39	97.70 ± 0.55
IBRA-PLS-DA	96.11 ± 1.37	97.10 ± 1.04	96.60 ± 0.98	96.70 ± 1.01
	97.19 ± 0.67	97.77 ± 0.67	97.67 ± 1.03	97.64 ± 0.95
IBRA-GSS	**98.08 ± 0.43**	**98.26 ± 0.42**	**98.39 ± 0.43**	**98.32 ± 0.39**
	98.24 ± 0.39	**98.56 ± 0.38**	**98.43 ± 0.42**	**98.48 ± 0.36**

Table 5.11: Feature selection method comparison — IP ($k = 5$).

permutation tests performed between IBRA-GSS and the other compared methods when selecting six, eight, and ten bands from the Kochia dataset. Similarly, Table 5.14 shows the results of the hypothesis tests between IBRA-GSS and the other methods when selecting five bands from the IP and SA datasets. In both cases, the upward-pointing arrow (↑) indicates that IBRA-GSS performed significantly better (i.e., p-value < 0.05), the downward-pointing arrow (↓) indicates that the compared method performed significantly better than IBRA-GSS, and the equal symbol (=) indicates that the difference between IBRA-GSS and the compared method is not statistically significant (i.e., p-value ≥ 0.05). According to both the t-test and the permutation test, our IBRA-GSS method performed significantly better than the other five band selection methods in each of the cases; however, it did not perform better than IBRA-PLS-DA nor IBRA-PCA (except for the "original band selection" case) when selecting eight or ten bands from the Kochia dataset.

We also evaluated how the distribution of the F1 scores resulting from using different

Method	OA	Prec	Rec	F1
PLS-DA	97.89 ± 0.18	98.99 ± 0.09	99.10 ± 0.10	99.04 ± 0.09
	98.46 ± 0.10	99.24 ± 0.04	99.33 ± 0.03	99.28 ± 0.03
OCF	98.37 ± 0.14	99.20 ± 0.07	99.29 ± 0.07	99.24 ± 0.07
	98.51 ± 0.12	99.27 ± 0.08	99.34 ± 0.06	99.30 ± 0.07
HAGRID	98.47 ± 0.11	99.26 ± 0.09	99.36 ± 0.05	99.31 ± 0.06
	98.51 ± 0.12	99.29 ± 0.08	99.37 ± 0.05	99.33 ± 0.06
SR-SSIM	97.89 ± 0.22	99.11 ± 0.11	99.07 ± 0.09	99.09 ± 0.09
	98.14 ± 0.21	99.18 ± 0.12	99.18 ± 0.09	99.18 ± 0.10
FNGBS	98.44 ± 0.14	99.34 ± 0.08	99.37 ± 0.06	99.35 ± 0.07
	98.51 ± 0.10	99.37 ± 0.05	99.39 ± 0.04	99.38 ± 0.04
IBRA-PCA	96.10 ± 0.78	98.23 ± 0.34	98.14 ± 0.49	98.16 ± 0.45
	98.49 ± 0.16	99.39 ± 0.07	99.39 ± 0.07	99.39 ± 0.06
IBRA-PLS-DA	96.26 ± 0.21	98.33 ± 0.14	98.36 ± 0.11	98.34 ± 0.10
	98.14 ± 0.20	99.25 ± 0.08	99.23 ± 0.09	99.24 ± 0.08
IBRA-GSS	**99.05 ± 0.05**	**99.56 ± 0.04**	**99.55 ± 0.03**	**99.56 ± 0.03**
	99.07 ± 0.09	**99.53 ± 0.06**	**99.55 ± 0.03**	**99.54 ± 0.04**

Table 5.12: Feature selection method comparison — SA ($k = 5$).

dimensionality reduction techniques changes as k changes. To do this, we considered a population that consisted of all the F1 scores obtained by all the methods (i.e., a sample size of 80) on the full Kochia dataset. From this, we show in Table 5.15 that the overall mean F1 score increases as k increases, while the standard deviation decreases; that is, the F1 scores are more spread out when k is small. We also performed a one-way analysis of variance (ANOVA). In this case, each group consists of the ten F1 scores obtained by each method. From Table 5.13, we already know that such statistically significant differences exist, so, as expected, the F statistics obtained for the three values of k are high and their corresponding p-values are quite small (p-value $\ll 0.05$), as shown in Table 5.15. However, notice that the F statistic decreases and the p-value increases as k increases. In other words, as k increases it becomes less likely that choosing a specific dimensionality reduction technique will cause a significant difference in the resulting average F1 score. This is not surprising since, by increasing k, we are effectively reinserting what was considered to be overlapping

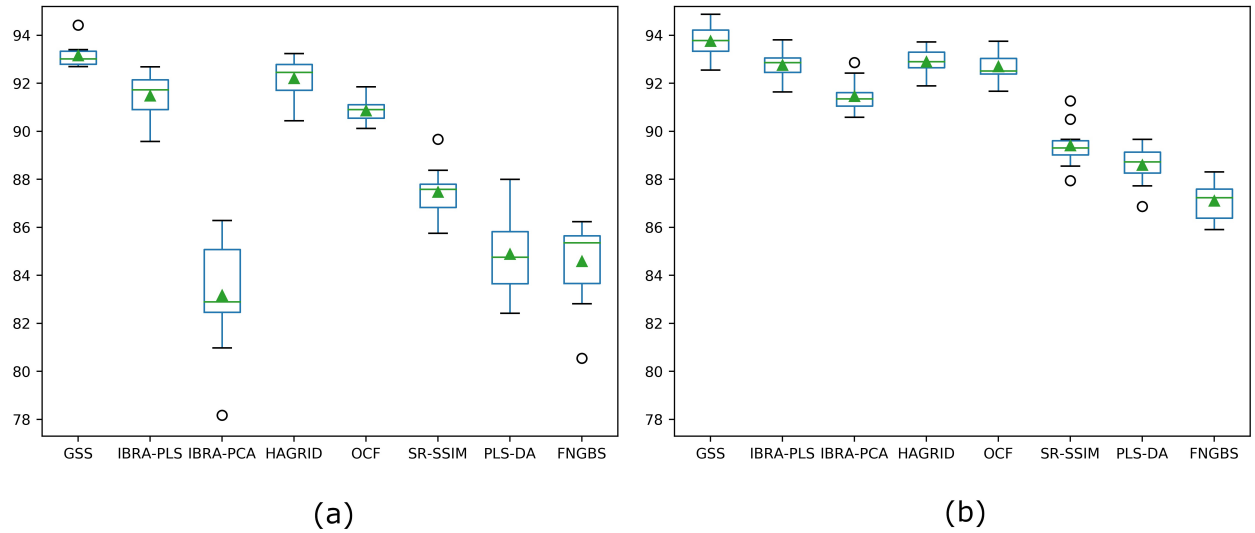

Figure 5.5: Box plot of F1 scores of different band selection methods on the Kochia dataset with $k = 6$. **(a)** Original band selection. **(b)** Multispectral filter simulation.

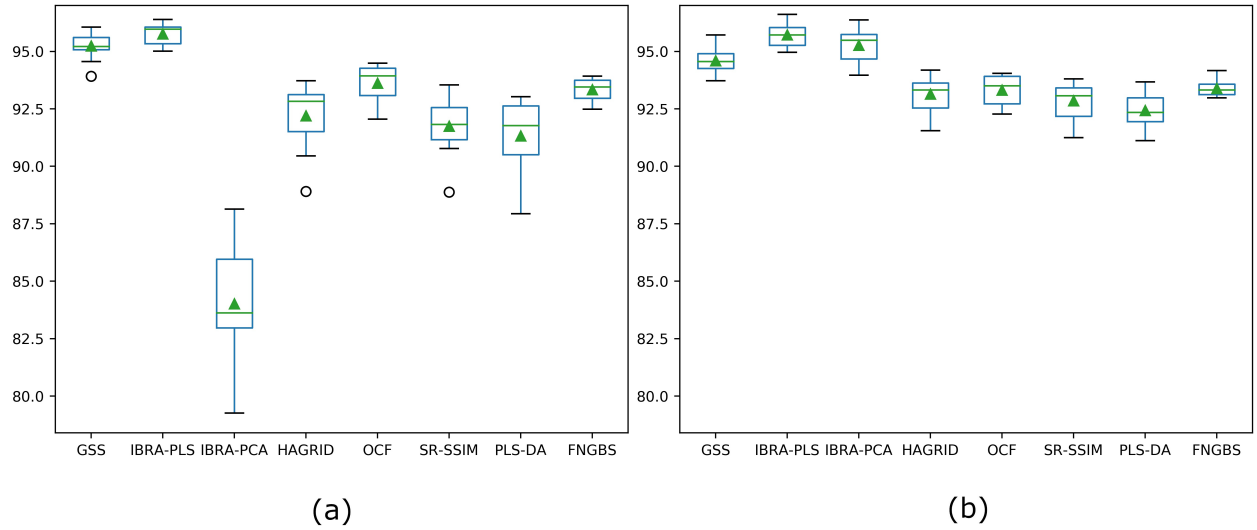

Figure 5.6: Box plot of F1 scores of different band selection methods on the Kochia dataset with $k = 8$. **(a)** Original band selection. **(b)** Multispectral filter simulation.

information.

Fig. 5.5, Fig. 5.6, and Fig. 5.7 depict the distribution of the F1 scores achieved by all the compared methods when selecting six, eight, and ten bands from the Kochia dataset,

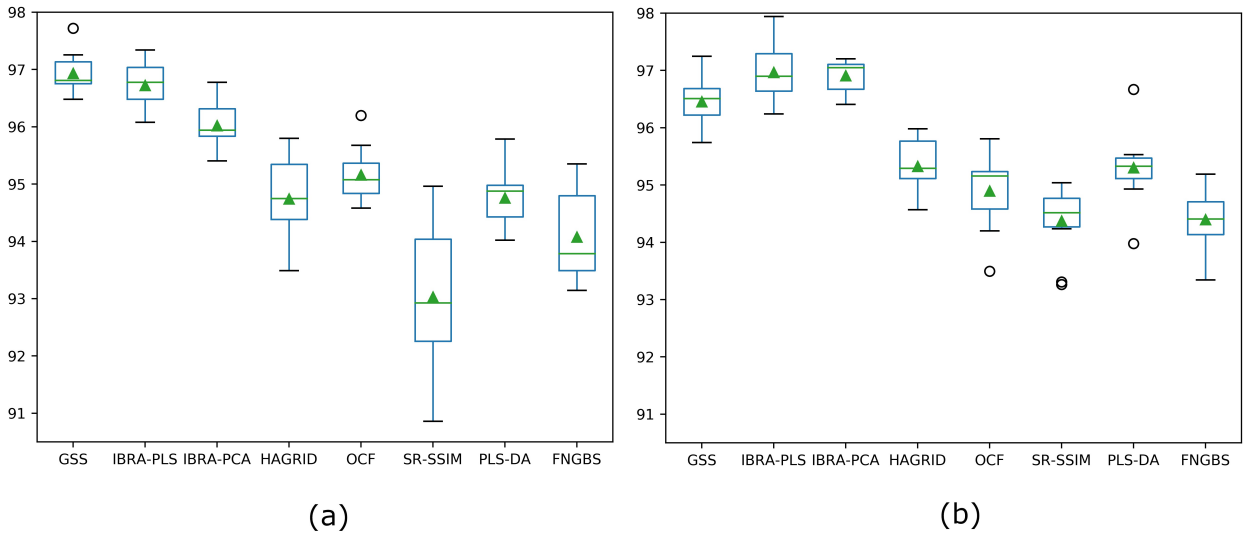

Figure 5.7: Box plot of F1 scores of different band selection methods on the Kochia dataset with $k = 10$. **(a)** Original band selection. **(b)** Multispectral filter simulation.

respectively. Here, the line through the center of each box indicates the median value of the F1 scores. The edges of boxes are the 25th and 75th percentiles. Whiskers extend to the maximum and minimum points. Outlier points are those past the end of the whiskers. Similarly, Fig. 5.8 and Fig. 5.9 show box plots of the distribution of the resulting F1 scores for the IP and SA datasets.

We also tested the four dataset size variations; for example, Fig. 5.10 depicts the performance metrics comparison of the eight methods for the Kochia dataset using four dataset size variations and selecting six, eight, and ten bands, respectively. For the three tested datasets, we found that the improvement in performance of our IBRA-GSS method over the other feature selection methods was still statistically significant even when the dataset size was reduced, except when selecting eight bands from the Kochia dataset and using 25% of the dataset (Fig. 5.10.d).

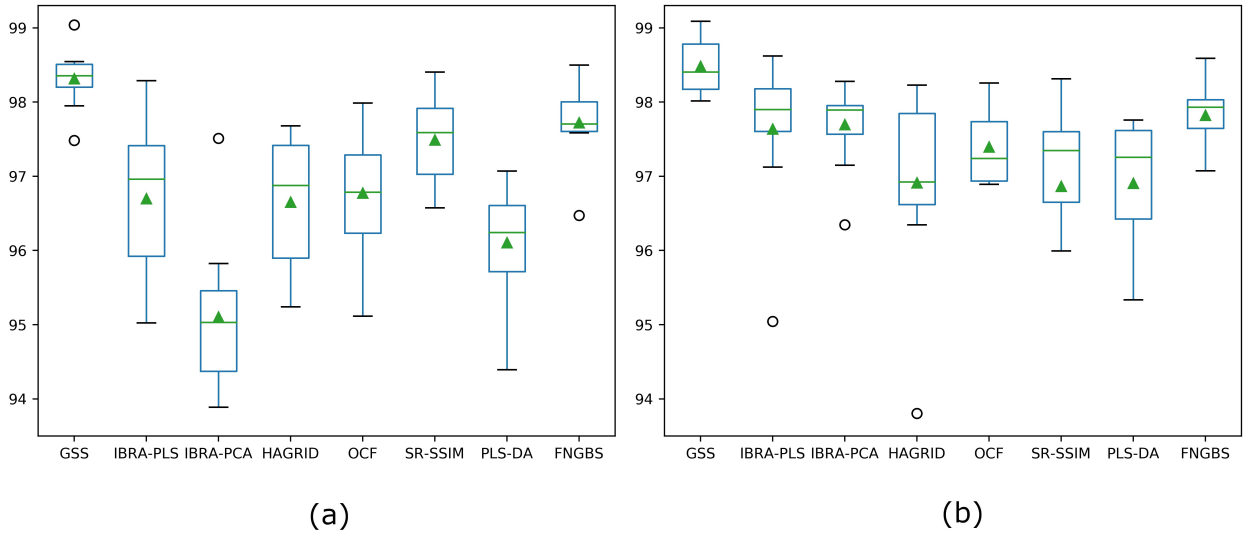

Figure 5.8: Box plot of F1 scores of different band selection methods on the Indian Pines dataset with $k = 5$. **(a)** Original band selection. **(b)** Multispectral filter simulation.

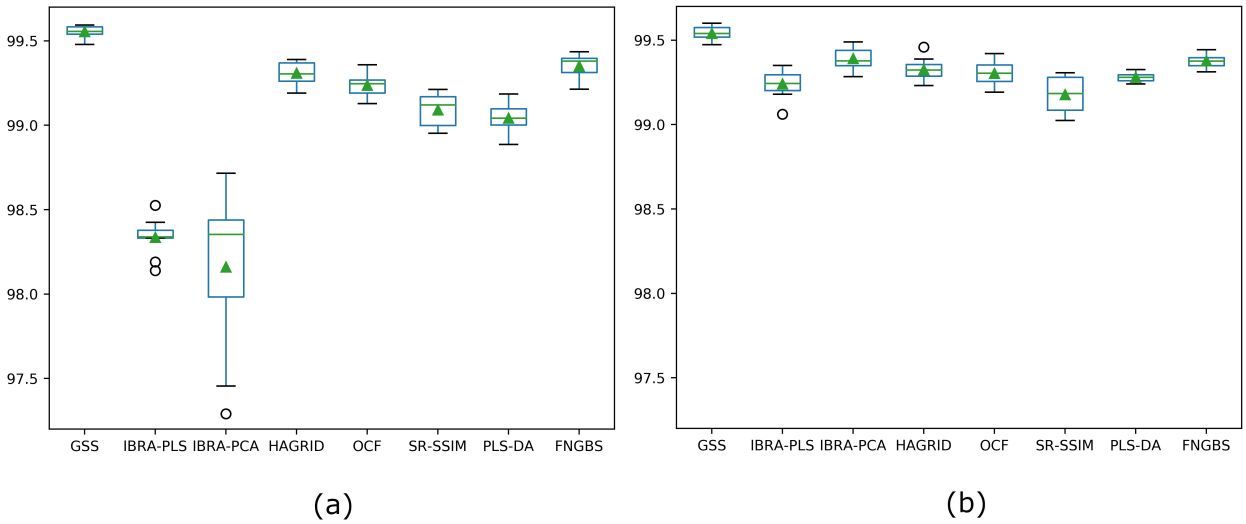

Figure 5.9: Box plot of F1 scores of different band selection methods on the Salinas dataset with $k = 5$. **(a)** Original band selection. **(b)** Multispectral filter simulation.

5.3.4 Discussion

Our IBRA method is used to cluster the entire spectrum into sets of similar (or collinear) spectral bands and to identify their cluster centers directly. Furthermore, we can interpret

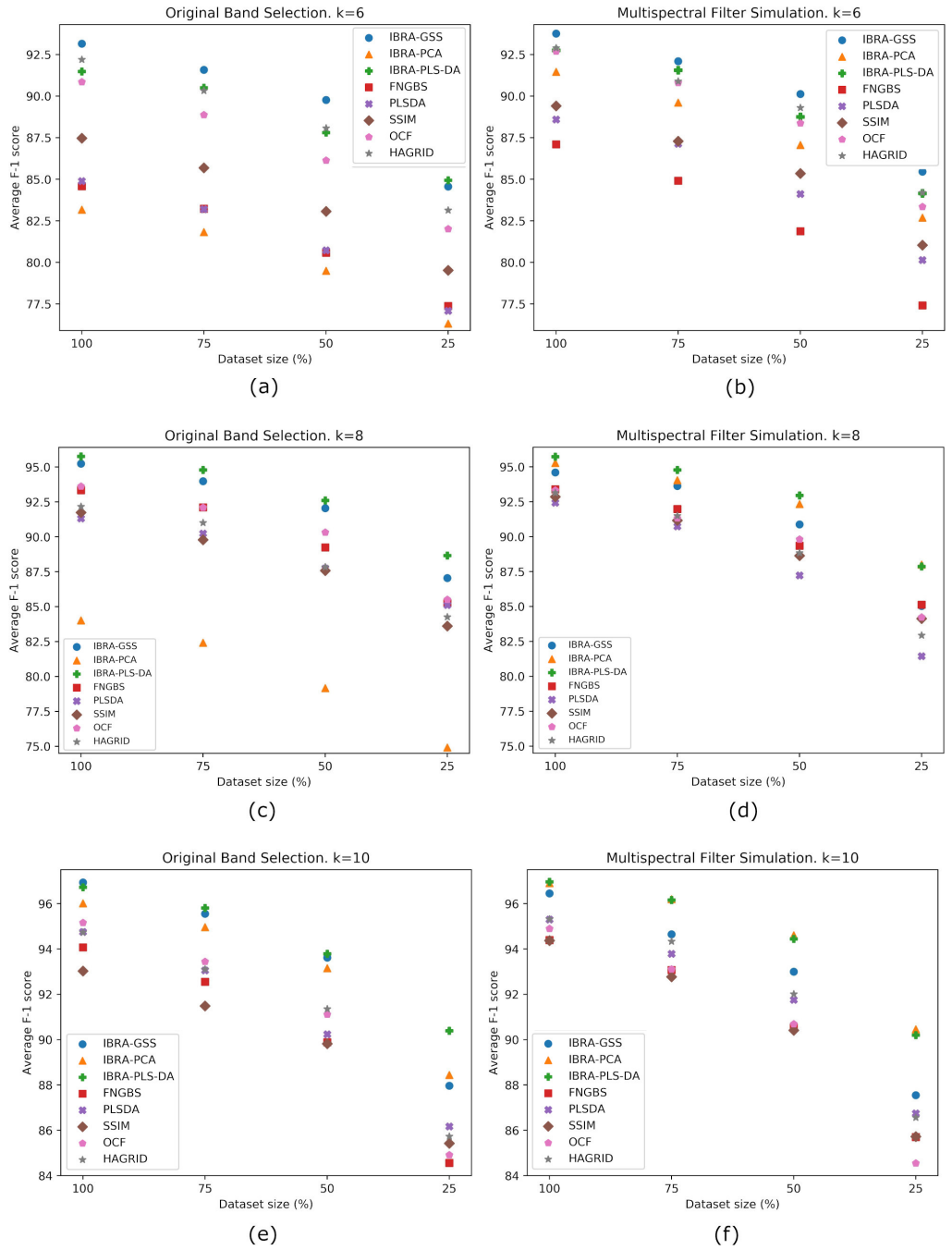

Figure 5.10: Average F1 scores obtained by each dimensionality reduction method for the Kochia dataset across different dataset sizes. **(a)** Original band selection using $k = 6$. **(b)** Multispectral filter simulation using $k = 6$. **(c)** Original band selection using $k = 8$. **(d)** Multispectral filter simulation using $k = 8$. **(e)** Original band selection using $k = 10$. **(f)** Multispectral filter simulation using $k = 10$.

Compared Method	**Kochia**, $k = 6$		**Kochia**, $k = 8$		**Kochia**, $k = 10$	
	***t*-test**	**Perm.**	***t*-test**	**Perm.**	***t*-test**	**Perm.**
FNGBS	7.4E-8 (↑)	2E-3 (↑)	4.2E-7 (↑)	1.7E-3 (↑)	1E-6 (↑)	2E-3 (↑)
	1.8E-9 (↑)	2E-3 (↑)	5.9E-5 (↑)	1.7E-3 (↑)	3.7E-8 (↑)	2E-3 (↑)
PLS-DA	6.5E-8 (↑)	2.6E-3 (↑)	2.1E-5 (↑)	2.6E-3 (↑)	1.4E-7 (↑)	2.6E-3 (↑)
	1.2E-9 (↑)	2.6E-3 (↑)	3.9E-3 (↑)	2.6E-3 (↑)	3.6E-5 (↑)	2.6E-3 (↑)
SR-SSIM	1E-7 (↑)	2E-3 (↑)	2.6E-6 (↑)	2E-3 (↑)	9.9E-6 (↑)	2E-3 (↑)
	1E-7 (↑)	2E-3 (↑)	4.2E-6 (↑)	2E-3 (↑)	9.9E-6 (↑)	2E-3 (↑)
OCF	3E-7 (↑)	1.9E-3 (↑)	6.3E-4 (↑)	3.5E-3 (↑)	8E-7 (↑)	1.9E-3 (↑)
	2E-3 (↑)	5.4E-3 (↑)	1.2E-5 (↑)	1.9E-3 (↑)	1.2E-5 (↑)	1.9E-3 (↑)
HAGRID	9.1E-4 (↑)	1.6E-3 (↑)	5.3E-5 (↑)	1.6E-3 (↑)	2.1E-6 (↑)	1.6E-3 (↑)
	2.8E-4 (↑)	3.3E-3 (↑)	5.5E-4 (↑)	1.6E-3 (↑)	1.6E-4 (↑)	1.6E-3 (↑)
IBRA-PCA	3.8E-7 (↑)	1.8E-3 (↑)	1.9E-7 (↑)	1.8E-3 (↑)	9E-5 (↑)	1.8E-3 (↑)
	2.9E-5 (↑)	1.8E-3 (↑)	**1.6E-2** (↓)	**2.3E-2** (↓)	**5.6E-2** (=)	**5.9E-2** (=)
IBRA-PLS-DA	1.8E-4 (↑)	1.7E-3 (↑)	**2.1E-2** (↓)	**1.2E-2** (↓)	**0.18** (=)	**0.18** (=)
	7E-3 (↑)	0.012 (↑)	**5.1E-6** (↓)	**1.7E-3** (↓)	**1.3E-2** (↓)	**1.9E-2** (↓)

Table 5.13: Hypothesis tests — Kochia dataset. p-values obtained comparing the classification performance of IBRA-GSS to that of the other methods.

Compared Method	**IP**, $k = 5$		**SA**, $k = 5$	
	***t*-test**	**Perm.**	***t*-test**	**Perm.**
FNGBS	1E-2 (↑)	5.7E-3 (↑)	4.1E-6 (↑)	2E-3 (↑)
	1.2E-2 (↑)	0.014 (↑)	2.8E-8 (↑)	1.7E-3 (↑)
PLS-DA	1E-5 (↑)	2.6E-3 (↑)	3.4E-8 (↑)	2.6E-3 (↑)
	5.2E-5 (↑)	2.6E-3 (↑)	1.4E-7 (↑)	1.8E-3 (↑)
SR-SSIM	2.5E-3 (↑)	3.6E-3 (↑)	1.9E-7 (↑)	2E-3 (↑)
	2.5E-3 (↑)	3.6E-3 (↑)	1.9E-7 (↑)	2E-3 (↑)
OCF	3E-7 (↑)	1.9E-3 (↑)	3.3E-7 (↑)	1.9E-3 (↑)
	4.7E-4 (↑)	1.9E-3 (↑)	4.2E-5 (↑)	1.6E-3 (↑)
HAGRID	3E-5 (↑)	1.6E-3 (↑)	2.8E-6 (↑)	1.6E-3 (↑)
	2.5E-3 (↑)	3.4E-3 (↑)	3.6E-6 (↑)	1.9E-3 (↑)
IBRA-PCA	7.9E-7 (↑)	1.8E-3 (↑)	5.4E-6 (↑)	1.8E-3 (↑)
	1E-2 (↑)	1.8E-3 (↑)	5.3E-5 (↑)	2.6E-3 (↑)
IBRA-PLS-DA	2.7E-4 (↑)	1.7E-3 (↑)	2.9E-11 (↑)	1.7E-3 (↑)
	2.5E-2 (↑)	1.3E-2 (↑)	1.4E-5 (↑)	1.7E-3 (↑)

Table 5.14: Hypothesis tests — IP and SA datasets. p-values obtained comparing the classification performance of IBRA-GSS to that of the other methods.

Metric \ # Bands	$k = 6$	$k = 8$	$k = 10$
Mean	88.48	92.15	95.18
	91.08	93.84	95.58
Std	3.93	3.68	1.43
	2.38	1.31	1.13
F **statistic**	75.99	69.29	34.05
	102.65	29.1	38.98
p**-value**	1.1E-30	1.9E-29	2E-20
	7.2E-35	1.3E-18	4.8E-22

Table 5.15: Dispersion analysis of the F1 scores obtained using all the compared dimensionality reduction methods on the full Kochia dataset.

these bands as representing a set of influencing spectral bands with wavelengths that are suitable for the design of multispectral filters. These pre-selected bands explain the variance of their neighbors (i.e., spectral bands that belong to the same cluster) in the original spectrum with a VIF value greater than a threshold $\theta \in [5, 12]$; therefore, keeping them and removing the other bands of each cluster allows us to avoid spectral bands that do not contain useful information for performing classification. That is, our method effectively identifies those spectral bands that carry information for performing classification while discarding redundant spectral bands.

Results shown in Table 5.2 demonstrate that it is possible for a model trained on the subset of spectral bands determined by IBRA, to achieve high accuracy values ($\sim$ 97–99%) while discarding most of the bands, similar to those obtained when using the full spectrum. For example, for the case of the Salinas dataset when using a VIF threshold of 10, we reduced the number of bands from 204 to 14, discarding more than 93% of the bands while decreasing the average F1 score less than 0.2 percentage points, which was also shown not to be significantly different. Thus, we were able to train a CNN classifier that required 80%

fewer trainable parameters than a model trained on the full spectrum yet without detriment to its performance.

Our GSS method uses information entropy as a ranking criterion to identify which bands incorporate more information among the bands pre-selected by IBRA. However, if we need to select at most k bands, the subset of bands with the greatest information entropy values may not be the best selection. For instance, for the Kochia dataset, for $k = 6$, the wavelengths of the bands with the highest entropy values were $[391.2, 463.3, 518.4,\ 616.0, 658.4, 675.3]$ (nm); however, line 9 in Algorithm 5.3 detected strong multicollinearity between bands with wavelengths 616.0, 658.4, and 675.3 nm in the first iteration (see Fig. 5.3). Rather than using redundant bands, our method selects a more diverse subset of bands if this helps to improve the classification performance.

It is worth noting that, according to the results described in Sec. 5.2.3, there does not necessarily have to be a single optimal VIF threshold θ to find the best set of spectral bands for a given classification task. Instead, more than one value of θ can lead to the same or similar selection of bands. For example, from Table 5.7, the only difference that we obtained using $\theta = 8$ with respect to using $\theta = 7$ is that we selected the wavelength 2224.7 nm (band index 175) instead of 2215.2 nm (band index 174). Considering that these two bands are contiguous, they explain practically the same variance, so both options are equally as good, which can be verified by observing that their resulting classification performance was very similar. There were other cases where, given two VIF thresholds, the GSS method selected bands that are located in different parts of the spectrum, and yet both options are considered equally as good as there was no statistical difference between their classification performance, as observed in Table 5.4 when using $\theta = 6$ and $\theta = 8$.

From Tables 5.8–5.12, we see that our IBRA-GSS method performed better than the other five band selection methods on the three datasets. The results remained consistent even when considering different dataset sizes, except in one case when using the "multispectral

filter simulation" approach on the Kochia dataset. In fact, Fig. 5.10 shows that the gap in performance between our method and the others became more noticeable when the dataset size was reduced. In addition, Table 5.15 shows that the average F1 scores obtained by each method in both approaches are more spread out when selecting six bands than when selecting eight or ten bands (which can be verified in Fig. 5.10). This confirms that the fewer spectral bands we select, the harder the band selection task will be; however, multispectral imagers generally become more practical computationally as the number of spectral channels becomes smaller. Thus, the fact that IBRA-GSS consistently outperformed the others when selecting a reduced set of spectral bands makes it the most suitable band selection method for multispectral imager design, at least for the data sets analyzed and the algorithms studied.

With our IBRA-GSS method, the classification performance resulting from the "original band selection" approach was very similar to the performance with the "multispectral filter simulation" approach, unlike some of the compared methods. This can be explained by the way we selected the first band candidates using IBRA. That is, a spectral band corresponding to a local minimum in the plot of spectral index vs. distance $d(x)$ (Fig.5.2) acts as a centroid due to its similarity to the spectral bands located on either side. Therefore, if we take this local minimum as the central wavelength of a multispectral filter, generate a Gaussian distribution around it by considering a standard bandwidth, and integrate it under the response curve, then we obtain reflectance values similar to those of the central band. Given that the simulated multispectral filter and the original spectral band present similar information, their classification performance was similar. This is convenient for a multispectral sensor design, as we would like the central wavelength of a filter to be the most representative.

Regarding our tested feature extraction methods, we observe from Table 5.13 that IBRA-PCA and IBRA-PLS-DA were able to obtain better or competitive results than those obtained by our IBRA-GSS band selection method when using eight ($k = 8$) or

ten ($k = 10$) feature channels from the Kochia dataset. Nevertheless, they could not obtain better performance than IBRA-GSS or other band selection methods when extracting fewer channels. This is due to the fact that PCA tries to explain the total variance of the original features (i.e., the IBRA-selected bands) and PLS-DA tries to explain the maximum variance of the response variable (i.e., maximizes inter-class separability) so that the fewer the number of dimensions we choose in both cases, the worse this explanation is. Conversely, when k is small, our GSS method does not apply a poor transformation on the spectral bands. Instead, it seeks to find a set of original spectral bands that maximize class separability. For the specific case of PCA, we noticed that the performance obtained using the "original band selection" was significantly lower than that obtained using the "multispectral filter simulation" approach. This is because reflectance measures captured by narrow hyperspectral bands tend to introduce a considerable number of outliers and PCA is an outlier sensitive feature extraction technique. On the other hand, the Gaussian transformation applied to simulate multispectral filters has a smoothing effect that reduces outlier noise; thus, improving the classification performance dramatically.

5.4 Conclusion

In this chapter, we focused on the proposal of filter-based (IBRA) and hybrid (IBRA-GSS) band selection methods in the context of hyperspectral image classification. Experimental results show that our IBRA method alone is able to discard up to 93% of the spectral bands and achieve similar classification performance to that obtained using the full spectrum. Therefore, IBRA successfully removes redundant spectral bands while maintaining the most relevant spectral information for a given classification task.

Furthermore, we presented a hybrid method for selecting salient wavelengths from a hyperspectral data cube based on two main steps: A pre-selection step that uses IBRA to identify a subset of independent spectral bands and a wrapper-based greedy selection

step based on information entropy (GSS). Experimental results showed our IBRA-GSS band selection method generally outperformed other commonly-employed feature selection methods on the Kochia, Indian Pines, and Salinas datasets.

We also presented a novel feature extraction framework that consists of applying any feature extraction technique to the results obtained by IBRA. By doing so, the reduced set of transformed feature channels try to explain most of the variance of the original data or the maximum variance of the response variable, depending on the type of feature extraction method selected. Our experiments showed that the results obtained by this framework outperformed the results obtained by all the compared band selection methods when the number of selected bands is sufficiently large (e.g. ten bands).

Finally, we showed that the inter-band redundancy method does not only reduce the search space considerably, but it also provides potential filter centers that are suitable for the design of multispectral sensors. Hence, another outcome of this work is the aid in the design of compact multispectral imagers that will assist in applications such as automatically identifying herbicide-resistance biotypes of the weed kochia.

CHAPTER SIX

CONCLUSION

Data captured by an imaging system is often processed to make observations and classifications about the world around us. The spatial and spectral content of the images obtained is key to analyzing the data, with spectral content playing a central role. Thus, hyperspectral imaging systems have become a reliable source of information due to their high spatial and spectral resolution that allows us to characterize objects with a great level of detail. For this reason, hyperspectral imagery is widely used for environmental monitoring, agricultural applications, biomedicine, among other critical applications.

Hyperspectral images are mostly used for supervised image classification tasks, as they facilitate the differentiation of objects that cannot be easily distinguished visually or that otherwise require laboratory analysis to be accurately classified (e.g., classification of a herbicide-resistant weed). Traditional machine learning methods focus on the analysis of the spectral domain exclusively; that is, they ignore the spatial information that surrounds a specific pixel. Conversely, modern deep learning techniques are able to extract spatial and spectral features conjunctively in order to improve classification performance metrics. Nevertheless, this type of processing is challenging due to the increased volume of data and the consequent computational burden.

In response to the need for a hyperspectral image classifier that prioritizes not only the classification performance but also the computational efficiency when storing and processing the data, we presented a convolutional neural network (CNN) architecture called Hyper3DNet. This novel architecture consists of two main components: a 3-D feature extractor designed to extract deep spectral features, and a 2-D spatial encoder. Results showed that Hyper3DNet achieved comparable state-of-the-art results when testing

it on hyperspectral and multispectral datasets. What is more, it required fewer trainable parameters and computational operations in comparison to other approaches, yet without detriment to its performance. Therefore, we conclude that it is possible to design convolutional neural networks that extract complex spectral and spatial features that do not involve excessive computational cost and do not compromise the classification performance.

Even if high classification performance metrics are achieved using reduced-cost convolutional networks trained on the full spectrum, we note that it is possible to reduce the computational burden even more if we only use the most relevant spectral information for a given classification task. This is because the dense spectral information collected by a hyperspectral imager is not required for every application. Therefore, the inherent high computational cost of processing hyperspectral images can be avoided using dimensionality reduction techniques.

In that sense, feature extraction techniques are used to transform the data and reduce its dimensionality, which improves the computational efficiency when processing HSIs. Feature selection techniques, on the other hand, help to determine the most relevant wavelengths for a given application, which allows for the design of compact multispectral imagers that could be used in place of hyperspectral imagers. This approach leads to significant economic savings, as fewer specialized storage and processing devices are required. It also leads to clarity and time-savings when analyzing data.

The narrow spectral bands utilized by hyperspectral imaging systems enable us to obtain images with high spectral resolution. This is particularly useful when trying to differentiate between objects from different classes based on subtle spectral changes on specific regions of the spectrum. Nevertheless, other regions of the spectrum that do not need this level of detail will be captured with the same fine spectral resolution, causing redundancy of the data. Keeping redundant features entails a waste of computational resources, as fewer features could be stored and processed without losing important information. Besides, a

highly redundant dataset might hinder the performance of many machine learning classifiers due to the Hughes phenomenon.

With these issues in mind, we presented IBRA, a filter-based selection method designed to remove redundant spectral bands. IBRA performs multicollinearity analysis between each spectral band and its neighbors, allowing us to find clusters of similar spectral bands and select the most representative band from each cluster. Experimental results showed that IBRA is able to discard more than 90% of the spectral bands such that a classification model trained on the selected features selected achieves similar performance to that of a model that was trained using the full spectrum. This demonstrates that IBRA successfully removes spectral redundancy and reduces the search space for further feature selection.

Furthermore, our IBRA-GSS method was presented as a hybrid selection method that allows for the selection of a specific number of spectral bands k. In particular, GSS is a wrapper-based selection method that is applied to the bands selected by IBRA. It is a greedy selection algorithm based on information entropy that uses a Hyper3DNetLite network to evaluate the classification performance of each selection of bands. IBRA-GSS was shown to outperform the other competing band selection methods on the three tested datasets even when we reduced the dataset sizes. We also observed that more than one VIF threshold in the IBRA step can lead to the same or similar selection of bands that are found to be equally as effective. This behavior was found to be consistent when the dataset size was reduced.

The Hyper3DNetLite network was presented as an alternative to Hyper3DNet when dealing with multispectral image classification. The architecture of Hyper3DNetLite is shallower than that of Hyper3DNet (i.e., it consists of fewer convolutional layers) so it requires fewer trainable parameters. This is specially suitable for multispectral datasets that use a reduced set of spectral bands. Thus, by using low-complexity models, such as Hyper3DNetLite, we avoid over-parameterization, which, in turn, prevents the model from overfitting. Experimental results show that Hyper3DNetLite achieved high classification

accuracy on datasets with a few spectral bands ($\sim 5-10$), which demonstrates that its low complexity did not harm classification performance and it lead to achieving classification metrics similar to those of models that used full spectrum information.

We also showed how to use our IBRA method as part of a novel feature selection framework. Two variants of this framework were tested, namely IBRA-PCA and IBRA-PLS-DA, which were able to obtain better results than all of the band selection methods when k was sufficiently large. In that sense, we demonstrated that the dimensionality reduction task becomes harder when k becomes smaller. Thus, given that our IBRA-GSS method consistently outperformed the other dimensionality reduction techniques when selecting a reduced set of spectral bands, we conclude it is the most suitable band selection method for the design of compact multispectral imagers among all the compared methods. This is also supported by the fact that our multispectral filter simulations showed that IBRA effectively provides representative optical filter centers that are suitable for the design of multispectral sensors.

To summarize, the main contributions presented in this thesis include the following:

- We presented Hyper3DNet, a low-cost convolutional neural network for hyperspectral image classification. We showed that it achieves better or comparable classification performance to other methods while requiring fewer computational complexity.

- We presented Hyper3DNetLite, a simplified version of Hyper3DNet designed for multispectral image classification. Results showed that, in spite of its low complexity, it achieved classification performance similar to that of models that used full spectrum information.

- We presented IBRA, a filter-based band selection method. We demonstrated IBRA helps to remove redundant spectral bands and provides representative optical filter centers that are suitable for the design of multispectral sensors.

- We presented IBRA-GSS as a hybrid-selection method that allows for the selection of a user-defined number of spectral bands. IBRA-GSS was shown to outperform the other competing band selection methods on the three tested datasets.

- We showed that IBRA can be used as part of a feature extraction framework. Results showed that this framework was able to obtain better results than all of the band selection methods when the number of feature channels was sufficiently large.

In the future, we plan to develop multispectral sensors based on the wavelengths recommended by our IBRA-GSS method. This would allow us to study implementation issues that may not be noticed using our multispectral filter simulation. Having compact multispectral sensors would also enable us to easily acquire data from drones and other platforms of opportunity for cost-effective sensing of herbicide-resistant weeds and many other applications.

REFERENCES CITED

[1] A.P. French A. Lowe, N. Harrison. Hyperspectral image analysis techniques for the detection and classification of the early onset of plant disease and stress. *Plant Methods*, 13(80), 2017.

[2] T. Adão, J. Hruška, L. Pádua, J. Bessa, E. Peres, R. Morais, and J. Sousa. Hyperspectral imaging: A review on UAV-based sensors, data processing and applications for agriculture and forestry. *Remote Sensing*, 9(11):1110, Oct 2017.

[3] D. F. Barbin, D.-W. Sun, and C. Su. NIR hyperspectral imaging as non-destructive evaluation tool for the recognition of fresh and frozen–thawed *porcine longissimus dorsi* muscles. *Innovative Food Science and Emerging Technologies*, 18:226–236, 2013.

[4] M. F. Baumgardner, L. L. Biehl, and D. A. Landgrebe. 220 band AVIRIS hyperspectral image data set: June 12, 1992 Indian Pine test site 3. `https://purr.purdue.edu/publications/1947/1`, Sep 2015. Accessed on 09.27.2021.

[5] R. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, Princeton, NJ, 1961.

[6] D.A. Belsley, E. Kuh, R.E. Welsch, and R.O. Wells. *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*. Wiley Series in Probability and Statistics - Applied Probability and Statistics Section Series. Wiley, 1980.

[7] M. Borengasser, W. Hungate, and R. Watkins. *Hyperspectral Remote Sensing*. CRC Press, 1st edition, 2007.

[8] C.-I. Chang, Q. Du, T.-L. Sun, and M. L. G. Althouse. A joint band prioritization and band-decorrelation approach to band selection for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 37(6):2631–2641, 1999.

[9] X. Cao, L. Xu, D. Meng, Q. Zhao, and Z. Xu. Integration of 3-dimensional discrete wavelet transform and Markov random field for hyperspectral image classification. *Neurocomputing*, 226:90–100, 2017.

[10] F. Castaldi, A. Castrignanò, and R. Casa. A data fusion and spatial data analysis approach for the estimation of wheat grain nitrogen uptake from satellite data. *International Journal of Remote Sensing*, 37(18):4317–4336, 2016.

[11] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi. Deep feature extraction and classification of hyperspectral images based on convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 54(10):6232–6251, Oct 2016.

[12] F. Chollet. Xception: deep learning with depthwise separable convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1800–1807, 2017.

[13] F. Dell'Acqua, P. Gamba, A. Ferrari, J. A. Palmason, J. A. Benediktsson, and K. Arnason. Exploiting spectral and spatial information in hyperspectral urban data with high resolution. *IEEE Geoscience and Remote Sensing Letters*, 1(4):322–326, 2004.

[14] Q. Du and N. Younan. Dimensionality reduction and linear discriminant analysis for hyperspectral image classification. In *Knowledge-Based Intelligent Information and Engineering Systems*, pages 392–399, 2008.

[15] B. Fang, Y. Li, H. Zhang, and J. Chan. Hyperspectral images classification based on dense convolutional networks with spectral-wise attention mechanism. *Remote Sensing*, 11(2):159, 2019.

[16] M. Fauvel, J. Chanussot, and J.A. Benediktsson. A spatial–spectral kernel-based approach for the classification of remote-sensing images. *Pattern Recognition*, 45(1):381–392, 2012.

[17] Lei Feng, Susu Zhu, Fei Liu, Yong He, Yidan Bao, and Chu Zhang. Hyperspectral imaging for seed quality and safety inspection: a review. *Plant Methods*, 15(91):2631–2641, 2019.

[18] A. G. Frenich, D. Jouan-Rimbaud, D. L. Massart, S. Kuttatharmmakul, M. Martínez Galera, and J. L. Martínez Vidal. Wavelength selection method for multicomponent spectrophotometric determinations using partial least squares. *Analyst*, 120:2787–2792, 1995.

[19] X. Gao, Y. Zhao, L. Dudziak, R. Mullins, and X. Cheng-Zhong. Dynamic channel pruning: Feature boosting and suppression. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, 2019.

[20] M. J. Giardino. A history of NASA remote sensing contributions to archaeology. *Journal of Archaeological Science*, 38(9):2003–2009, 2011.

[21] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15, pages 315–323, 2011.

[22] A. Goetz, G. Vane, J. E. Solomon, and B. N. Rock. Imaging spectrometry for earth remote sensing. *Science*, 228(4704):1147–1153, 1985.

[23] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[24] A.A. Gowen, C.P. O'Donnell, P.J. Cullen, G. Downey, and J.M. Frias. Hyperspectral imaging – an emerging process analytical tool for food quality and safety control. *Trends in Food Science & Technology*, 18(12):590–598, 2007.

[25] A. G. Gualtieri, S. Chettri, R. F. Cromp, and L. F. Johnson. Support vector machine classifiers as applied to AVIRIS data. In *Summaries of the Eighth JPL Airborne Earth Science Workshop*, 1999.

[26] N.A. Hagen and M.W. Kudenov. Review of snapshot spectral imaging technologies. *Optical Engineering*, 52(9):1 – 23, 2013.

[27] J. Hair, R. Anderson, and B. Babin. *Multivariate Data Analysis*. Prentice Hall, 7th edition, 2009.

[28] M. Halicek, J. V. Little, X. Wang, A. Y. Chen, and B. Fei. Optical biopsy of head and neck cancer using hyperspectral imaging and convolutional neural networks. *Journal of Biomedical Optics*, 24(3):1 – 9, 2019.

[29] J. Ham, Yangchi Chen, M. M. Crawford, and J. Ghosh. Investigation of the random forest framework for classification of hyperspectral data. *IEEE Transactions on Geoscience and Remote Sensing*, 43(3):492–501, March 2005.

[30] T. Haraguchi, D.-Q. Ding, A. Yamamoto, T. Kaneda, T. Koujin, and Y. Hiraoka. Multiple-color fluorescene imaging of chromosomes and microtubules in living cells. *Cell Structure and Function*, 24(5):291–298, 1999.

[31] T. Haraguchi, T. Shimi, T. Koujin, N. Hashiguchi, and Y. Hiraoka. Spectral imaging fluorescence microscopy. *Genes to Cells*, 7(9):881–887, 2002.

[32] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016.

[33] P. Helber, B. Bischke, A. Dengel, and D. Borth. EuroSAT: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, July 2019.

[34] Y. Hiraoka, T. Shimi, and T. Haraguchi. Multispectral imaging fluorescence microscopy for living cells. *Cell Structure and Function*, 27(5):367–374, 2002.

[35] S. J. Hook. NASA 2014 The hyperspectral infrared imager (HyspIRI)–science impact of deploying instruments on separate platforms. White Paper 14-13, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, July 2014.

[36] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. MobileNets: Efficient convolutional neural networks for mobile vision applications, 2017. arXiv:1704.04861.

[37] W. Hu, Y. Huang, L. Wei, F. Zhang, and H. Li. Deep convolutional neural networks for hyperspectral image classification. *Journal of Sensors*, 12:1–12, 2015.

[38] G. Huang, Z. Liu, L. v. d. Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, July 2017.

[39] G. Hughes. On the mean accuracy of statistical pattern recognizers. *IEEE Transactions on Information Theory*, 14(1):55–63, 1968.

[40] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 448–456, Lille, France, 07–09 Jul 2015.

[41] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning with Applications in R*. Springer Science+Business Media, New York, 2013.

[42] C. Jayaprakash, B. B. Damodaran, S. V., and K. P. Soman. Dimensionality reduction of hyperspectral images for classification using randomized independent component analysis. In *5th International Conference on Signal Processing and Integrated Networks (SPIN)*, pages 492–496, 2018.

[43] S. Ji, W. Xu, M. Yang, and K. Yu. 3D convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231, 2013.

[44] S. Jia, L. Shen, and Q. Li. Gabor feature-based collaborative representation for hyperspectral imagery classification. *IEEE Transactions on Geoscience and Remote Sensing*, 53(2):1118–1129, Feb 2015.

[45] X. Jia, B. Kuo, and M. M. Crawford. Feature mining for hyperspectral image classification. *Proceedings of the IEEE*, 101(3):676–697, March 2013.

[46] I.T. Jolliffe. *Principal Component Analysis*. Springer Verlag, 1986.

[47] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations*, San Diego, CA, USA, May 7-9, 2015.

[48] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25, 2012.

[49] B. Kunkel, F. Blechinger, D. Viehmann, H. Van Der Piepen, and R. Doerffer. ROSIS imaging spectrometer and its potential for ocean parameter measurements (airborne and space-borne). *International Journal of Remote Sensing*, 12(4):753–761, 1991.

[50] D. A. Landgrebe. *Signal Theory Methods in Multispectral Remote Sensing*. Wiley, Newark, NJ, 2005.

[51] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

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

[53] J. Li, J. M. Bioucas-Dias, and A. Plaza. Spectral–spatial classification of hyperspectral data using loopy belief propagation and active learning. *IEEE Transactions on Geoscience and Remote Sensing*, 51(2):844–856, Feb 2013.

[54] S. Li, W. Song, L. Fang, Y. Chen, P. Ghamisi, and J. A. Benediktsson. Deep learning for hyperspectral image classification: An overview. *IEEE Transactions on Geoscience and Remote Sensing*, 57(9):6690–6709, Sep. 2019.

[55] S. Li, Z. Xia, and C. Zhang. Sensitive wavelengths selection in identification of *Ophiopogon japonicus* based on near-infrared hyperspectral imaging technology. *International Journal of Analytical Chemistry*, 2017.

[56] X. Li, H. Shen, L. Zhang, H. Zhang, Q. Yuan, and G. Yang. Recovering quantitative remote sensing products contaminated by thick clouds and shadows using multitemporal dictionary learning. *IEEE Transactions on Geoscience and Remote Sensing*, 52(11):7086–7098, 2014.

[57] Y. Liang, X. Zhao, A. J. X. Guo, and F. Zhu. Hyperspectral image classification with deep metric learning and conditional random field. *IEEE Geoscience and Remote Sensing Letters*, 17(6):1042–1046, 2019.

[58] R. Logan. Calibration and characterization of a VNIR hyperspectral imager for produce monitoring. Master's thesis, Department of Electrical and Computer Engineering, Montana State University, 2020.

[59] R. D. Logan, B. Scherrer, J. Senecal, N. S. Walton, A. Peerlinck, J. W. Sheppard, and J. A. Shaw. Assessing produce freshness using hyperspectral imaging and machine learning. *Journal of Applied Remote Sensing*, 15(3):1 – 17, 2021.

[60] B. Lu, P. D. Dao, J. Liu, Y. He, and J. Shang. Recent advances of hyperspectral imaging technology and applications in agriculture. *Remote Sensing*, 12(16), 2020.

[61] G. Lu and B. Fei. Medical hyperspectral imaging: A review. *Journal of Biomedical Optics*, 19(1), 2014.

[62] R. Lu. Multispectral imaging for predicting firmness and soluble solids content of apple fruit. *Postharvest Biology and Technology*, 31(2):147–157, 2004.

[63] R. Lucas, A. Rowlands, O. Niemann, and R. Merton. *Hyperspectral Sensors and Applications*, pages 11–49. Springer Berlin Heidelberg, 2004.

[64] W. Ma, Q. Yang, Y. Wu, W. Zhao, and X. Zhang. Double-branch multi-attention mechanism network for hyperspectral image classification. *Remote Sensing*, 11(11):1307, 2019.

[65] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez. Convolutional neural networks for large-scale remote-sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(2):645–657, 2017.

[66] M. Marker and W. Rayens. Partial least squares for discrimination. *Journal of Chemometrics*, 17:166–173, 2003.

[67] S. A. Mathews. Design and fabrication of a low-cost, multispectral imaging system. *Applied Optics*, 47(28):F71–F76, Oct 2008.

[68] S. A. Medjahed and M. Ouali. Band selection based on optimization approach for hyperspectral image classification. *The Egyptian Journal of Remote Sensing and Space Science*, 21(3):413–418, 2018.

[69] F. Melgani and L. Bruzzone. Classification of hyperspectral remote sensing images with support vector machines. *IEEE Transactions on Geoscience and Remote Sensing*, 42(8):1778–1790, Aug 2004.

[70] G. Morales, J. Sheppard, R. Logan, and J. Shaw. Hyperspectral band selection for multispectral image classification with convolutional networks. In *IEEE International Joint Conference on Neural Networks (IJCNN)*, June 2021.

[71] G. Morales, J. Sheppard, B. Scherrer, and J. Shaw. Reduced-cost hyperspectral convolutional neural networks. *Journal of Applied Remote Sensing*, 14(3):036519, 2020.

[72] G. Morales, J. W. Sheppard, R. D. Logan, and J. A. Shaw. Hyperspectral dimensionality reduction based on inter-band redundancy analysis and greedy spectral selection. *Remote Sensing*, 13(18), 2021.

[73] L. Mou, P. Ghamisi, and X. X. Zhu. Deep recurrent neural networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(7):3639–3655, July 2017.

[74] K. Nagasubramanian, S. Jones, A. K. Singh, S. Sarkar, A. Singh, and B. Ganapathysubramanian. Plant disease identification using explainable 3D deep learning on hyperspectral images. *Plant Methods*, 15(98), 2019.

[75] P. Nugent, J. Shaw, P. Jha, B. Scherrer, A. Donelick, and V. Kumar. Discrimination of herbicide-resistant kochia with hyperspectral imaging. *Journal of Applied Remote Sensing*, 12(1):1 – 10, 2018.

[76] K. Ose, T. Corpetti, and L. Demagistri. Multispectral satellite image processing. In N. Baghdadi and M. Zribi, editors, *Optical Remote Sensing of Land Surface*, pages 57–124. Elsevier, 2016.

[77] J. Pan, C. C. Ferrer, K. McGuinness, N. E. O'Connor, J. Torres, E. Sayrol, and X. Giro i Nieto. SalGAN: Visual saliency prediction with generative adversarial networks, 2017. arXiv:1701.01081.

[78] M.E. Paoletti, J.M. Haut, J. Plaza, and A. Plaza. A new deep convolutional neural network for fast hyperspectral image classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 145:120–147, 2018.

[79] F. Pasa, V. Golkov, F. Pfeiffer, D. Cremers, and D. Pfeiffer. Efficient Deep Network Architectures for Fast Chest X-Ray Tuberculosis Screening and Visualization. *Scientific Reports and Visualization*, 9(1), 2019.

[80] J. Pirgazi, M. Alimoradi, T.E. Abharian, and M.H. Olyaee. An efficient hybrid filter-wrapper metaheuristic-based gene selection method for high dimensional datasets. *Scientific Reports*, 9:18580, 2019.

[81] G Polder, G W A M Van Der Heijden, and I T Young. Spectral Image Analysis For Measuring Ripeness of Tomatoes. *Transactions of the ASAE*, 45(4):1155–1161, 2002.

[82] W. M. Porter and H. T. Enmark. A system overview of the airborne visible/infrared imaging spectrometer (AVIRIS). In G. Vane, editor, *Imaging Spectroscopy II*, volume 0834, pages 22 – 31. SPIE, 1987.

[83] S. Prasad and L. Bruce. Limitations of principal components analysis for hyperspectral target recognition. *IEEE Geoscience and Remote Sensing Letters*, 5(4):625–629, 2008.

[84] S. Prasad and L. Bruce. Partial least squares discriminant analysis: A dimensionality reduction method to classify hyperspectral data. *Statistica Applicata - Italian Journal of Applied Statistics*, 21(2), 2019.

[85] P. Preet, S. S. Batra, and Jayadeva. Feature selection for classification of hyperspectral data by minimizing a tight bound on the VC dimension. `arXiv:1509.081121`, 2015.

[86] H. Pu, L. Lin, and D.-W. Sun. Principles of hyperspectral microscope imaging techniques and their applications in food quality and safety detection: A review. *Comprehensive Reviews in Food Science and Food Safety*, 18(4):853–866, 2019.

[87] P. Rajkumar, N. Wang, G. Elmasry, G. S. V. Raghavan, and Y. Gariepy. Studies on Banana Fruit Quality and Maturity Stages Using Hyperspectral Imaging. *Journal of Food Engineering*, 108:194–200, 2012.

[88] A. Rodriguez and A. Laio. Clustering by fast search and find of density peaks. *Science*, 344(6191):1492–1496, 2014.

[89] S. K. Roy, G. Krishna, S. R. Dubey, and B. B. Chaudhuri. HybridSN: Exploring 3-D-2-D CNN Feature Hierarchy for Hyperspectral Image Classification. *IEEE Geoscience and Remote Sensing Letters*, pages 1–5, 2019.

[90] L. Samaniego, A. Bardossy, and K. Schulz. Supervised classification of remotely sensed imagery using a modified k-NN technique. *IEEE Transactions on Geoscience and Remote Sensing*, 46(7):2112–2125, July 2008.

[91] B. Scherrer, J. Sheppard, P. Jha, and J. Shaw. Hyperspectral imaging and neural networks to classify herbicide-resistant weeds. *Journal of Applied Remote Sensing*, 13(4):1 – 15, 2019.

[92] A. Schneider and H. Feussner. Chapter 5 - diagnostic procedures. In A. Schneider and H. Feussner, editors, *Biomedical Engineering in Gastrointestinal Surgery*, pages 87–220. Academic Press, 2017.

[93] R. A. Schowengerdt. Chapter 9 - thematic classification. In Robert A. Schowengerdt, editor, *Remote Sensing*, pages 387–XXXIII. Academic Press, Burlington, 3rd edition, 2007.

[94] J. J. Senecal, J. W. Sheppard, and J. A. Shaw. Efficient convolutional neural networks for multi-spectral image classification. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, Budapest, Hungary, July 2019.

[95] M. Seydgar, A. Alizadeh Naeini, M. Zhang, W. Li, and M. Satari. 3-D convolution-recurrent networks for spectral-spatial classification of hyperspectral images. *Remote Sensing*, 11(7):883, 2019.

[96] J. Shapey, Y. Xie, E. Nabavi, R. Bradford, S.R. Saeed, S. Ourselin, and T. Vercauteren. Intraoperative multispectral and hyperspectral label-free imaging: A systematic review of in vivo clinical studies. *Journal of Biophotonics*, 12(9):e201800455, 2019.

[97] L. Sifre. *Rigid-Motion Scattering For Image Classification*. PhD thesis, Applied Math Center, Ecole Polytechnique, 2014.

[98] A. Signoroni, M. Savardi, A. Baronio, and S. Benini. Deep learning meets hyperspectral image analysis: A multidisciplinary review. *Journal of Imaging*, 5(5):52, May 2019.

[99] D. Sliney, R.T. Wangemann, J.K. Franks, and M.L. Wolbarsht. Visual sensitivity of the eye to infrared laser radiation. *Journal of the Optical Society of America*, 66(4):339–341, 1976.

[100] M. Sugiyama. Dimensionality reduction of multimodal labeled data by Local Fisher Discriminant Analysis. *Journal of Machine Learning Research*, 8(37):1027–1061, 2007.

[101] W. Sun and Q. Du. Hyperspectral band selection: A review. *IEEE Geoscience and Remote Sensing Magazine*, 7(2):118–139, 2019.

[102] C. Szegedy, Wei L., Yangqing J., P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[103] S. Tadjudin. *Classification of High Dimensional Data with Limited Training Samples*. PhD thesis, School of Electrical and Computer Engineering, Purdue University, 1998.

[104] F. Taherkhani, J. Dawson, and N.M. Nasrabadi. Deep sparse band selection for hyperspectral face recognition, 2019. arXiv:1908.09630.

[105] G. Themelis, J.S. Yoo, and V. Ntziachristos. Multispectral imaging using multiple-bandpass filters. *Optics Letters*, 33(9):1023–1025, May 2008.

[106] Q. Tong, Y. Xue, and L. Zhang. Progress in hyperspectral remote sensing science and technology in china over the past three decades. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(1):70–91, 2014.

[107] J. Tschannerl, J. Ren, J. Zabalza, and S. Marshall. Segmented autoencoders for unsupervised embedded hyperspectral band selection. In *7th European Workshop on Visual Information Processing (EUVIP)*, pages 1–6, 2018.

[108] M. P. Uddin, M. A. Mamun, and M. A. Hossain. Feature extraction for hyperspectral image classification. In *IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*, pages 379–382, 2017.

[109] V. Vapnik and A. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *The Theory of Probability and Its Applications*, 16:264–280, 1971.

[110] C. Wallays, B. Missotten, Jo. De Baerdemaeker, and W. Saeys. Hyperspectral waveband selection for on-line measurement of grain cleanness. *Biosystems Engineering*, 104(1):1–7, 2009.

[111] N. Walton, J. Sheppard, and J. Shaw. Using a genetic algorithm with histogram-based feature selection in hyperspectral image classification. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, page 1364–1372, Prague, Czech Republic, 2019.

[112] Q. Wang, Q. Li, and X. Li. A fast neighborhood grouping method for hyperspectral band selection. *IEEE Transactions on Geoscience and Remote Sensing*, 59(6):5028–5039, 2021.

[113] Q. Wang, J. Lin, and Y. Yuan. Salient band selection for hyperspectral image classification via manifold ranking. *IEEE Transactions on Neural Networks and Learning Systems*, 27(6):1279–1289, June 2016.

[114] Q. Wang, F. Zhang, and X. Li. Optimal clustering framework for hyperspectral band selection. *IEEE Transactions on Geoscience and Remote Sensing*, 56(10):5910–5922, 2018.

[115] H. Wold. Soft modelling by latent variables: The non-linear iterative partial least squares (NIPALS) approach. *Journal of Applied Probability*, 12(S1):117–142, 1975.

[116] B. Wu. *Efficient Deep Neural Networks*. PhD thesis, Electrical Engineering and Computer Sciences, University of California, Berkeley, 2019.

[117] Di Wu and Da-Wen Sun. Advanced applications of hyperspectral imaging technology for food quality and safety analysis and assessment: A review — Part I: Fundamentals. *Innovative Food Science and Emerging Technologies*, 19, 2013.

[118] H. Wu and S. Prasad. Convolutional recurrent neural networks for hyperspectral data classification. *Remote Sensing*, 9(3):298, 2017.

[119] B. Xu, X. Li, W. Hou, Y. Wang, and Y. Wei. A similarity-based ranking method for hyperspectral band selection. Early access in *IEEE Transactions on Geoscience and Remote Sensing*, 2021.

[120] J. Xue and B. Su. Significant Remote Sensing Vegetation Indices: A Review of Developments and Applications. *Journal of Sensors*, 2017.

[121] X. Yu, R. Wang, B. Liu, and A. Yu. Salient feature extraction for hyperspectral image classification. *Remote Sensing Letters*, 10(6):553–562, 2019.

[122] M. Zeiler. ADADELTA: An adaptive learning rate method, 2012. arXiv:1212.5701.

[123] M. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Computer Vision (ECCV)*, pages 818–833, 2014.

[124] L. Zhang, L. Zhang, and B. Du. Deep learning for remote sensing data: A technical tutorial on the state of the art. *IEEE Geoscience and Remote Sensing Magazine*, 4(2):22–40, June 2016.

[125] W. Zhao, Z. Guo, J. Yue, X. Zhang, and L. Luo. On combining multiscale deep learning features for the classification of hyperspectral remote sensing imagery. *International Journal of Remote Sensing*, 36(13):3368–3379, 2015.

[126] Z. Zhong, J. Li, Z. Luo, and M. Chapman. Spectral-spatial residual network for hyperspectral image classification: A 3-D deep learning framework. *IEEE Transactions on Geoscience and Remote Sensing*, 56(2):847–858, Feb 2018.

APPENDICES

APPENDIX A

GSS ON REDUCED DATASETS

A.1 Kochia Dataset

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[395.5, 463.3, 565.1, 700.8, 722.0 , 993.3]	90.71 ± 0.89	91.23 ± 0.77	91.00 ± 0.85	91.09 ± 0.80
11	[395.5, 408.2, 463.3, 586.3, 662.6, 700.8]	89.42 ± 0.91	90.50 ± 0.83	90.27 ± 0.80	90.38 ± 0.80
10	**[391.2, 463.3, 569.3, 675.3, 730.4, 993.3]**	91.08 ± 0.41	91.77 ± 0.40	91.44 ± 0.44	**91.59 ± 0.41**
9	[391.2, 463.3, 569.3, 700.8, 730.4, 993.3]	91.21 ± 0.64	91.63 ± 0.58	91.38 ± 0.74	91.49 ± 0.63
8	[387.0 , 404.0 , 463.3, 577.8, 700.8, 722.0]	90.51 ± 0.97	91.18 ± 0.84	90.99 ± 0.84	91.07 ± 0.82
7	[387.0, 404.0, 463.3, 569.3, 700.8, 722.0]	90.56 ± 0.52	91.43 ± 0.46	90.85 ± 0.57	91.12 ± 0.50
6	[387.0 , 404.0 , 463.3, 586.3, 700.8, 717.7]	90.29 ± 0.64	91.05 ± 0.67	90.79 ± 0.53	90.90 ± 0.58
5	[387.0, 463.3, 586.3, 700.8, 722.0, 1001.8]	89.98 ± 0.93	90.65 ± 0.83	90.35 ± 0.86	90.48 ± 0.82

Table A.1: GSS on 75% of the Kochia dataset ($k = 6$).

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[395.5, 408.2, 463.3, 518.4, 616.0, 662.6, 675.3, 700.8]	91.50 ± 0.88	92.20 ± 0.82	92.08 ± 0.82	92.12 ± 0.81
11	[395.5, 408.2, 463.3, 565.1, 675.3, 700.8, 726.2, 993.3]	93.20 ± 0.55	93.72 ± 0.51	93.57 ± 0.63	93.64 ± 0.57
10	[391.2, 463.32, 569.3, 675.3, 730.4, 963.6, 993.3, 1006.0]	92.91 ± 0.60	93.37 ± 0.71	93.16 ± 0.68	93.25 ± 0.66
9	[391.2, 463.3, 569.3, 671.1, 700.8, 717.7, 730.4, 993.3]	93.36 ± 0.68	93.87 ± 0.61	93.71 ± 0.69	93.78 ± 0.65
8	[387.0, 404.0, 463.3, 518.4, 577.8, 654.1, 675.3, 700.8]	93.24 ± 0.87	93.85 ± 0.72	93.72 ± 0.95	93.77 ± 0.82
7	[387.0, 404.0, 463.3, 518.4, 569.3, 654.1, 675.3, 700.8]	93.28 ± 0.62	93.93 ± 0.52	93.77 ± 0.56	93.84 ± 0.53
6	**[387.0, 404.0, 463.3, 586.3, 679.6, 700.8, 730.4, 1001.8]**	93.47 ± 0.59	94.15 ± 0.44	93.85 ± 0.60	**93.98 ± 0.51**
5	[387.0, 463.3, 586.3, 645.6, 700.8, 722.0, 980.6, 1001.8]	92.04 ± 0.60	92.65 ± 0.45	92.38 ± 0.59	92.49 ± 0.49

Table A.2: GSS on 75% of the Kochia dataset ($k = 8$).

VIF	Selected bands (nm)	OA	Prec	Rec	F1
12	[395.5, 408.2 , 463.3, 518.4, 565.1, 616.0, 675.3, 700.8, 722.0, 993.3]	94.72 ± 0.40	95.14 ± 0.39	94.99 ± 0.41	95.06 ± 0.39
11	[395.5, 408.2, 463.3, 565.1, 662.6, 675.3, 700.8, 713.5, 726.2, 993.3]	94.60 ± 0.34	94.99 ± 0.29	94.87 ± 0.36	94.92 ± 0.31
10	[391.2, 463.3, 518.4, 569.3, 616.0, 658.4, 675.3, 717.7, 730.4, 993.3]	94.35 ± 0.45	94.88 ± 0.44	94.37 ± 0.56	94.61 ± 0.48
9	[391.2, 463.3, 518.4, 569.3, 616.0, 671.1, 700.8, 717.7, 730.4, 993.3]	94.63 ± 0.43	95.15 ± 0.38	94.89 ± 0.49	95.01 ± 0.43
8	**[387.0, 404.0, 463.3, 518.4, 577.8, 654.1, 675.3, 700.8, 722.0, 1006.0]**	95.24 ± 0.36	95.67 ± 0.36	95.45 ± 0.39	**95.56 ± 0.37**
7	[387.0, 404.0, 463.3, 518.4, 569.3, 654.1, 675.3, 700.8, 722.0, 1006.0]	94.98 ± 0.44	95.40 ± 0.43	95.22 ± 0.38	95.30 ± 0.39
6	[387.0, 404.0, 463.3, 586.3, 679.6, 700.8, 717.7, 730.4, 963.6, 1001.8]	94.77 ± 0.49	95.18 ± 0.48	95.07 ± 0.54	95.12 ± 0.51
5	[387.0, 463.3, 586.3, 645.6, 700.8, 722.0, 832.2, 946.7, 980.6, 1001.8]	93.05 ± 0.95	93.61 ± 0.97	93.35 ± 0.82	93.47 ± 0.88

Table A.3: GSS on 75% of the Kochia dataset ($k = 10$).

VIF	Selected bands (nm)	OA	Prec	Rec	F1
12	[395.5, 463.3, 565.1, 700.8, 722.0 , 993.3]	88.74 ± 0.78	89.50 ± 0.92	88.89 ± 0.67	89.14 ± 0.74
11	[395.5, 408.2, 463.3, 586.3, 662.6, 700.8]	87.21 ± 1.15	88.56 ± 1.22	88.20 ± 1.14	88.35 ± 1.14
10	**[391.2, 463.3, 569.3, 675.3, 730.4, 993.3]**	88.97 ± 0.61	88.98 ± 0.50	89.21 ± 0.74	**89.54 ± 0.60**
9	[391.2, 463.3, 569.3, 700.8, 730.4, 993.3]	89.39 ± 0.70	90.12 ± 0.74	89.54 ± 0.71	89.47 ± 0.70
8	[387.0 , 404.0 , 463.3, 577.8, 700.8, 722.0]	88.65 ± 0.71	89.51 ± 0.59	89.05 ± 0.95	89.25 ± 0.72
7	[387.0, 404.0, 463.3, 569.3, 675.3, 700.8]	88.04 ± 1.07	89.05 ± 0.95	89.05 ± 1.06	89.02 ± 0.95
6	[387.0 , 404.0 , 463.3, 586.3, 700.8, 717.7]	88.66 ± 0.72	89.61 ± 0.65	89.12 ± 0.75	89.34 ± 0.68
5	[387.0, 463.3, 586.3, 645.6, 700.8, 722.0]	86.56 ± 1.06	88.14 ± 1.09	87.19 ± 0.94	87.59 ± 0.97

Table A.4: GSS on 50% of the Kochia dataset ($k = 6$).

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[395.5, 408.2 , 463.3, 565.1, 675.3, 700.8, 722.0, 993.3]	90.44 ± 0.72	91.28 ± 0.62	91.03 ± 0.72	91.12 ± 0.66
11	[395.5, 408.2 , 463.3, 565.1, 675.3, 700.8, 726.2, 993.3]	90.44 ± 1.01	91.25 ± 0.90	90.96 ± 1.04	91.06 ± 0.96
10	[391.2, 463.3, 569.3, 675.3, 730.4, 963.6, 993.3, 1006.0]	90.42 ± 0.96	91.21 ± 0.87	90.72 ± 1.00	90.93 ± 0.92
9	[391.2, 463.3, 569.3, 671.1, 700.8, 730.4, 963.6, 993.3]	90.62 ± 0.83	91.40 ± 0.74	91.06 ± 0.74	91.19 ± 0.74
8	**[387.0, 404.0, 463.3, 518.4, 577.8, 654.1, 675.3, 700.8]**	91.36 ± 0.51	92.18 ± 0.48	91.97 ± 0.61	**92.05 ± 0.52**
7	[387.0, 404.0, 463.3, 518.4, 569.3, 654.1, 675.3, 700.8]	91.28 ± 1.04	91.96 ± 0.89	91.96 ± 1.02	91.93 ± 0.94
6	[387.0, 404.0, 463.3, 586.3, 679.6, 700.8, 730.4, 1001.8]	90.92 ± 1.03	91.90 ± 0.85	91.42 ± 1.07	91.62 ± 0.96
5	[387.0, 463.3, 586.3, 645.6, 700.8, 722.0, 980.6, 1001.8]	89.16 ± 1.14	90.06 ± 0.96	89.58 ± 1.15	89.75 ± 1.04

Table A.5: GSS on 50% of the Kochia dataset ($k = 8$).

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[395.5, 408.2 , 463.3, 518.4, 565.1, 616.0, 675.3, 700.8, 722.0, 993.3]	92.62 ± 0.79	93.31 ± 0.65	92.86 ± 0.79	93.06 ± 0.70
11	[395.5, 408.2, 463.3, 565.1, 662.6, 675.3, 700.8, 713.5, 726.2, 993.3]	92.84 ± 0.46	93.51 ± 0.51	92.98 ± 0.57	93.22 ± 0.52
10	[391.2, 463.3, 518.4, 569.3, 616.0, 658.4, 675.3, 717.7, 730.4, 993.3]	91.86 ± 0.75	92.71 ± 0.62	91.93 ± 0.81	92.27 ± 0.68
9	[391.2, 463.3, 518.4, 569.3, 671.1, 700.8, 717.7, 730.4, 963.6, 993.3]	92.32 ± 0.59	92.92 ± 0.55	92.54 ± 0.60	92.71 ± 0.55
8	[387.0, 404.0, 463.3, 518.4, 577.8, 654.1, 675.3, 700.8, 722.0, 1006.0]	93.23 ± 0.88	93.88 ± 0.68	93.39 ± 0.95	93.61 ± 0.80
7	**[387.0, 404.0, 463.3, 518.4, 569.3, 654.1, 675.3, 700.8, 722.0, 1006.0]**	93.19 ± 0.87	93.82 ± 0.79	93.47 ± 0.76	**93.62 ± 0.75**
6	[387.0, 404.0, 463.3, 586.3, 679.6, 700.8, 717.7, 730.4, 963.6, 1001.8]	92.60 ± 0.84	93.55 ± 0.65	92.81 ± 0.94	93.13 ± 0.78
5	[387.0, 463.3, 586.3, 645.6, 700.8, 722.0, 832.2, 946.7, 980.6, 1001.8]	90.73 ± 1.17	91.53 ± 1.17	90.98 ± 1.11	91.22 ± 1.08

Table A.6: GSS on 50% of the Kochia dataset ($k = 10$).

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[395.5, 408.2, 463.3, 565.1, 700.8, 722.0]	82.04 ± 1.57	83.31 ± 1.79	82.28 ± 1.41	82.66 ± 1.50
11	[395.5, 408.2, 463.3, 565.1, 700.8, 713.5]	82.04 ± 1.57	83.31 ± 1.79	82.28 ± 1.41	82.66 ± 1.50
10	**[391.2, 463.3, 569.3, 675.3, 730.4, 993.3]**	84.03 ± 0.85	85.38 ± 0.56	84.05 ± 0.94	**84.56 ± 0.74**
9	[391.2, 463.3, 569.3, 700.8, 730.4, 993.3]	83.09 ± 1.02	84.48 ± 1.06	82.89 ± 1.16	83.47 ± 1.13
8	[387.0 , 404.0 , 463.3, 577.8, 700.8, 722.0]	83.32 ± 1.64	84.43 ± 1.82	83.56 ± 1.55	83.83 ± 1.58
7	[387.0, 404.0, 463.3, 569.3, 675.3, 700.8]	83.85 ± 1.26	85.31 ± 1.48	84.01 ± 1.23	84.48 ± 1.20
6	[387.0 , 404.0 , 463.3, 586.3, 700.8, 717.7]	82.89 ± 1.23	84.15 ± 1.56	83.15 ± 1.22	83.45 ± 1.27
5	[387.0, 463.3, 586.3, 645.6, 700.8, 722.0]	80.84 ± 0.73	82.24 ± 0.62	81.12 ± 1.09	81.36 ± 0.79

Table A.7: GSS on 25% of the Kochia dataset ($k = 6$).

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[395.5, 408.2 , 463.3, 565.1, 675.3, 700.8, 722.0, 993.3]	85.68 ± 1.54	87.02 ± 1.30	86.04 ± 1.57	86.43 ± 1.47
11	[395.5, 408.2 , 463.3, 565.1, 675.3, 700.8, 726.2, 993.3]	85.98 ± 1.17	87.13 ± 1.20	86.36 ± 1.25	86.65 ± 1.24
10	[391.2, 463.3, 569.3, 675.3, 730.4, 963.6, 993.3, 1006.0]	85.98 ± 1.04	87.05 ± 1.21	85.94 ± 1.07	86.37 ± 1.10
9	[391.2, 463.3, 569.3, 671.1, 700.8, 717.7, 730.4, 993.3]	86.53 ± 0.95	87.71 ± 1.01	86.64 ± 1.11	87.05 ± 1.04
8	**[387.0, 404.0, 463.3, 518.4, 577.8, 654.1, 675.3, 700.8]**	86.32 ± 1.14	87.53 ± 1.15	86.76 ± 1.26	**87.05 ± 1.20**
7	[387.0, 404.0, 463.3, 518.4, 569.3, 654.1, 675.3, 700.8]	86.54 ± 1.74	87.59 ± 1.74	86.53 ± 1.86	86.96 ± 1.80
6	[387.0, 404.0, 463.3, 586.3, 679.6, 700.8, 730.4, 1001.8]	86.17 ± 1.44	87.26 ± 1.06	86.65 ± 1.60	86.85 ± 1.36
5	[387.0, 463.3, 586.3, 645.6, 700.8, 722.0, 980.6, 1001.8]	84.57 ± 1.30	85.79 ± 1.45	84.61 ± 1.35	85.03 ± 1.38

Table A.8: GSS on 25% of the Kochia dataset ($k = 8$).

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[395.5, 408.2 , 463.3, 518.4, 565.1, 616.0, 675.3, 700.8, 722.0, 993.3]	87.01 ± 1.29	88.18 ± 1.48	86.98 ± 0.97	87.48 ± 1.16
11	[395.5, 408.2, 463.3, 565.1, 662.6, 675.3, 700.8, 713.5, 726.2, 993.3]	87.23 ± 1.20	88.23 ± 1.11	87.34 ± 1.11	87.70 ± 1.06
10	[391.2, 463.3, 518.4, 569.3, 616.0, 658.4, 675.3, 717.7, 730.4, 993.3]	85.81 ± 1.43	87.14 ± 1.40	85.68 ± 1.19	86.25 ± 1.23
9	[391.2, 463.3, 518.4, 569.3, 671.1, 700.8, 717.7, 730.4, 963.6, 993.3]	86.72 ± 1.05	87.64 ± 0.92	86.70 ± 1.05	87.06 ± 0.94
8	**[387.0, 404.0, 463.3, 518.4, 577.8, 654.1, 675.3, 700.8, 722.0, 1006.0]**	87.49 ± 1.32	88.61 ± 1.51	85.57 ± 1.15	**87.96 ± 1.21**
7	[387.0, 404.0, 463.3, 518.4, 569.3, 654.1, 675.3, 700.8, 722.0, 1006.0]	87.24 ± 2.07	88.34 ± 1.88	87.22 ± 2.04	87.69 ± 1.94
6	[387.0, 404.0, 463.3, 586.3, 679.6, 700.8, 717.7, 730.4, 963.6, 1001.8]	86.84 ± 1.33	88.12 ± 1.30	86.82 ± 1.27	87.33 ± 1.25
5	[387.0, 463.3, 586.3, 645.6, 700.8, 722.0, 832.2, 946.7, 980.6, 1001.8]	84.38 ± 1.43	85.65 ± 1.41	84.43 ± 1.65	84.93 ± 1.53

Table A.9: GSS on 25% of the Kochia dataset ($k = 10$).

A.2 Indian Pines Dataset

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[498.3, 646.7, 706.2, 754.4, 1023.7]	96.63 ± 0.66	97.07 ± 0.91	96.96 ± 0.98	96.96 ± 0.90
11	[557.5, 587.2, 706.2, 754.4, 1023.7]	95.64 ± 0.55	96.74 ± 0.46	96.30 ± 1.17	96.45 ± 0.83
10	**[498.3, 626.9, 706.2, 754.4, 1023.7]**	96.72 ± 0.55	97.35 ± 0.56	96.98 ± 1.01	**97.09 ± 0.84**
9	[557.5, 626.9, 706.2, 821.8, 1023.7]	96.72 ± 0.62	97.08 ± 0.75	96.57 ± 1.24	96.72 ± 1.10
8	[607.0, 646.7, 706.2, 821.8, 1023.7]	96.40 ± 1.12	96.85 ± 1.08	96.75 ± 1.38	96.75 ± 1.23
7	[567.4, 587.2, 706.2, 821.8, 1023.7]	96.44 ± 0.69	96.81 ± 1.01	96.68 ± 0.95	96.70 ± 0.97
6	[577.3, 706.2, 821.8, 918.0, 1023.7]	94.80 ± 0.91	95.64 ± 1.27	95.49 ± 1.30	95.43 ± 1.36
5	[577.3, 715.8, 812.2, 918.0, 1023.7]	94.81 ± 1.29	96.02 ± 0.96	95.71 ± 1.06	95.82 ± 1.00

Table A.10: GSS on 75% of the Indian Pines dataset.

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[498.3, 646.7, 706.2, 754.4, 1023.7]	94.31 ± 0.57	95.48 ± 0.56	94.99 ± 0.80	95.14 ± 0.41
11	[557.5, 587.2, 706.2, 754.4, 1023.7]	93.36 ± 0.65	95.13 ± 0.79	94.08 ± 1.18	94.49 ± 0.69
10	[498.3, 626.9, 706.2, 754.4, 1023.7]	93.68 ± 0.86	95.03 ± 0.70	94.43 ± 0.97	94.61 ± 0.84
9	**[557.5, 626.9, 706.2, 821.8, 1023.7]**	94.41 ± 0.43	95.29 ± 0.46	94.37 ± 1.26	**94.69 ± 0.87**
8	[607.0, 646.7, 706.2, 821.8, 1023.7]	94.07 ± 0.98	95.24 ± 0.97	94.76 ± 1.48	94.49 ± 1.12
7	[567.4, 587.2, 706.2, 821.8, 1023.7]	93.87 ± 0.79	94.95 ± 0.66	94.47 ± 1.35	94.56 ± 0.94
6	[577.3, 706.2, 821.8, 918.0, 1023.7]	91.82 ± 0.85	93.82 ± 0.70	92.46 ± 1.70	92.91 ± 0.97
5	[577.3, 715.8, 812.2, 918.0, 1023.7]	91.29 ± 1.03	93.61 ± 1.06	92.46 ± 1.32	92.81 ± 0.82

Table A.11: GSS on 50% of the Indian Pines dataset.

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[498.3, 646.7, 706.2, 754.4, 1023.7]	86.95 ± 1.64	89.39 ± 1.57	83.75 ± 2.48	85.26 ± 2.14
11	[557.5, 587.2, 706.2, 754.4, 1023.7]	86.36 ± 1.19	89.89 ± 1.46	84.69 ± 3.31	86.12 ± 2.77
10	**[498.3, 626.9, 706.2, 754.4, 1023.7]**	87.23 ± 1.91	90.20 ± 2.56	84.48 ± 3.92	**86.17 ± 3.62**
9	[557.5, 626.9, 706.2, 821.8, 1023.7]	87.91 ± 1.78	87.12 ± 5.04	82.88 ± 5.60	83.79 ± 5.51
8	[607.0, 646.7, 706.2, 821.8, 1023.7]	87.40 ± 1.58	90.23 ± 2.30	83.35 ± 4.82	84.95 ± 4.12
7	[567.4, 587.2, 706.2, 821.8, 1023.7]	88.03 ± 1.27	89.69 ± 2.06	84.86 ± 4.24	85.92 ± 3.72
6	[577.3, 706.2, 821.8, 918.0, 1023.7]	83.75 ± 2.36	86.75 ± 2.04	80.31 ± 2.94	82.09 ± 2.57
5	[577.3, 715.8, 812.2, 918.0, 1023.7]	84.22 ± 1.43	88.65 ± 1.13	83.15 ± 2.53	84.87 ± 2.08

Table A.12: GSS on 25% of the Indian Pines dataset.

A.3 Salinas Dataset

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[532.14, 731.83, 950.54, 1368.93, 2044.06]	98.36 ± 0.19	99.25 ± 0.09	99.30 ± 0.11	99.27 ± 0.10
11	[541.65, 731.83, 950.54, 1368.93, 2044.06]	98.38 ± 0.13	99.25 ± 0.06	99.30 ± 0.08	99.27 ± 0.07
10	[541.65, 731.83, 950.54, 1254.82, 2044.06]	98.25 ± 0.15	99.18 ± 0.11	99.24 ± 0.08	99.21 ± 0.09
9	[541.65, 731.83, 950.54, 1254.82, 2044.06]	98.25 ± 0.15	99.18 ± 0.11	99.24 ± 0.08	99.21 ± 0.09
8	[731.83, 950.54, 1159.73, 1254.82, 2044.06]	98.81 ± 0.10	99.46 ± 0.05	99.41 ± 0.07	99.43 ± 0.06
7	[731.83, 950.54, 1159.73, 1254.82, 2034.55]	98.73 ± 0.13	99.39 ± 0.10	99.37 ± 0.08	99.38 ± 0.09
6	**[570.18, 950.54, 1159.73, 1245.31, 2034.55]**	98.91 ± 0.12	99.45 ± 0.07	99.49 ± 0.06	**99.47 ± 0.06**
5	[560.67, 950.54, 1245.31, 1368.93, 2034.55]	98.75 ± 0.13	99.43 ± 0.07	99.43 ± 0.07	99.43 ± 0.07

Table A.13: GSS on 75% of the Salinas dataset.

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[532.14, 731.83, 950.54, 1368.93, 2044.06]	97.79 ± 0.12	98.95 ± 0.10	98.97 ± 0.11	98.96 ± 0.09
11	[541.65, 731.83, 950.54, 1368.93, 2044.06]	97.82 ± 0.15	98.93 ± 0.11	98.98 ± 0.09	98.95 ± 0.08
10	[541.65, 731.83, 950.54, 1359.42, 2044.06]	97.79 ± 0.13	98.97 ± 0.06	99.00 ± 0.05	98.99 ± 0.05
9	[541.65, 731.83, 950.54, 1359.42, 2044.06]	97.79 ± 0.13	98.97 ± 0.06	99.00 ± 0.05	98.99 ± 0.05
8	[731.83, 950.54, 1159.73, 1254.82, 2044.06]	98.20 ± 0.13	99.12 ± 0.09	99.09 ± 0.06	99.10 ± 0.06
7	[731.83, 950.54, 1159.73, 1254.82, 2034.55]	98.09 ± 0.14	99.06 ± 0.10	99.01 ± 0.08	99.03 ± 0.08
6	**[570.18, 950.54, 1159.73, 1245.31, 2034.55]**	98.35 ± 0.23	99.14 ± 0.12	99.16 ± 0.09	**99.15 ± 0.10**
5	[560.67, 950.54, 1245.31, 1368.93, 2034.55]	98.18 ± 0.12	99.14 ± 0.07	99.12 ± 0.08	99.13 ± 0.07

Table A.14: GSS on 50% of the Salinas dataset.

VIF	**Selected bands** (nm)	**OA**	**Prec**	**Rec**	**F1**
12	[532.14, 731.83, 950.54, 1368.93, 2044.06]	96.74 ± 0.28	98.31 ± 0.22	98.37 ± 0.22	98.34 ± 0.21
11	[541.65, 731.83, 950.54, 1368.93, 2044.06]	96.73 ± 0.24	98.32 ± 0.12	98.39 ± 0.14	98.35 ± 0.13
10	[541.65, 731.83, 950.54, 1359.42, 2044.06]	96.64 ± 0.27	98.38 ± 0.20	98.43 ± 0.21	98.40 ± 0.19
9	[541.65, 731.83, 950.54, 1359.42, 2044.06]	96.64 ± 0.27	98.38 ± 0.20	98.43 ± 0.21	98.40 ± 0.19
8	[731.83, 950.54, 1159.73, 1254.82, 2044.06]	96.92 ± 0.31	98.48 ± 0.19	98.36 ± 0.23	98.41 ± 0.19
7	**[731.83, 950.54, 1159.73, 1254.82, 2034.55]**	97.09 ± 0.27	98.59 ± 0.18	98.46 ± 0.21	**98.51 ± 0.18**
6	[570.18, 950.54, 1245.31, 1368.93, 2034.55]	97.04 ± 0.36	98.49 ± 0.21	98.50 ± 0.17	98.49 ± 0.18
5	[560.67, 950.54, 1245.31, 1368.93, 2034.55]	96.89 ± 0.49	98.43 ± 0.25	98.39 ± 0.26	98.40 ± 0.25

Table A.15: GSS on 25% of the Salinas dataset.

APPENDIX B

BAND SELECTION METHODS ON REDUCED DATASETS

B.1 Kochia Dataset

Method	OA	Prec	Rec	F1
FNGBS	82.86 ± 1.44	83.71 ± 1.28	82.84 ± 1.27	83.23 ± 1.26
	84.77 ± 0.83	85.50 ± 0.73	84.43 ± 0.89	84.91 ± 0.78
PLS-DA	83.09 ± 1.42	83.45 ± 1.34	83.04 ± 1.59	83.20 ± 1.46
	86.93 ± 0.84	87.49 ± 0.82	86.86 ± 1.05	87.14 ± 0.91
OCF	88.48 ± 0.79	89.08 ± 0.72	88.71 ± 0.87	88.87 ± 0.76
	90.41 ± 0.47	91.12 ± 0.54	90.54 ± 0.53	90.81 ± 0.49
SR-SSIM	84.58 ± 0.89	86.01 ± 0.82	85.42 ± 0.92	85.69 ± 0.84
	86.48 ± 1.10	87.36 ± 0.88	87.27 ± 1.23	87.30 ± 1.02
HAGRID	89.84 ± 0.46	90.39 ± 0.42	90.29 ± 0.52	90.33 ± 0.43
	90.40 ± 0.50	91.03 ± 0.43	90.78 ± 0.59	90.89 ± 0.48
IBRA-PCA	81.73 ± 1.55	82.11 ± 1.42	81.61 ± 1.70	81.82 ± 1.53
	89.47 ± 0.89	89.96 ± 0.93	89.34 ± 1.12	89.62 ± 1.01
IBRA-PLS-DA	89.89 ± 0.92	90.66 ± 0.82	90.42 ± 0.94	90.51 ± 0.85
	91.11 ± 0.60	91.77 ± 0.51	91.39 ± 0.73	91.57 ± 0.60
IBRA-GSS	**91.08 ± 0.41**	**91.70 ± 0.40**	**91.44 ± 0.44**	**91.59 ± 0.41**
	91.68 ± 0.59	**92.30 ± 0.65**	**91.93 ± 0.60**	**92.10 ± 0.59**

Table B.1: Feature selection method comparison on 75% of the Kochia dataset ($k = 6$).

Method	OA	Prec	Rec	F1
FNGBS	91.58 ± 0.57	92.38 ± 0.58	91.88 ± 0.65	92.11 ± 0.48
	91.61 ± 0.66	92.27 ± 0.60	91.73 ± 0.77	91.98 ± 0.66
PLS-DA	89.46 ± 0.93	90.56 ± 0.84	89.95 ± 0.78	90.24 ± 0.77
	90.27 ± 0.62	91.00 ± 0.60	90.50 ± 0.67	90.73 ± 0.62
OCF	91.48 ± 0.67	92.28 ± 0.67	91.95 ± 0.65	92.10 ± 0.64
	91.33 ± 0.48	92.02 ± 0.50	91.72 ± 0.56	91.86 ± 0.49
SR-SSIM	88.94 ± 0.95	90.07 ± 0.90	89.56 ± 1.02	89.79 ± 0.93
	90.48 ± 0.76	91.45 ± 0.69	90.92 ± 0.86	91.17 ± 0.76
HAGRID	90.43 ± 1.76	91.34 ± 1.64	90.73 ± 1.69	91.01 ± 1.65
	91.17 ± 0.68	91.61 ± 0.76	91.40 ± 0.76	91.49 ± 0.69
IBRA-PCA	82.23 ± 2.43	82.64 ± 2.46	82.24 ± 2.54	82.42 ± 2.49
	93.80 ± 0.45	94.17 ± 0.50	93.94 ± 0.48	94.04 ± 0.46
IBRA-PLS-DA	**94.48 ± 0.43**	**94.86 ± 0.46**	**94.73 ± 0.49**	**94.78 ± 0.43**
	94.47 ± 0.59	**94.86 ± 0.46**	**94.73 ± 0.49**	**94.78 ± 0.43**
IBRA-GSS	93.47 ± 0.59	94.15 ± 0.44	93.85 ± 0.60	93.98 ± 0.51
	93.20 ± 0.61	93.81 ± 0.39	93.48 ± 0.79	93.63 ± 0.60

Table B.2: Feature selection method comparison on 75% of the Kochia dataset ($k = 8$).

Method	OA	Prec	Rec	F1
FNGBS	92.26 ± 0.45	92.81 ± 0.37	92.34 ± 0.63	92.56 ± 0.48
	92.84 ± 0.79	93.26 ± 0.75	92.90 ± 0.73	93.07 ± 0.74
PLS-DA	92.66 ± 1.06	93.24 ± 0.94	92.90 ± 0.89	93.06 ± 0.91
	93.48 ± 0.64	93.94 ± 0.59	93.66 ± 0.69	93.79 ± 0.63
OCF	93.22 ± 0.64	93.71 ± 0.64	93.23 ± 0.74	93.44 ± 0.67
	92.81 ± 0.28	93.37 ± 0.41	92.89 ± 0.32	93.11 ± 0.30
SR-SSIM	90.65 ± 0.89	91.65 ± 0.82	91.36 ± 0.84	91.49 ± 0.81
	92.14 ± 0.91	93.00 ± 0.73	92.58 ± 1.06	92.77 ± 0.89
HAGRID	92.73 ± 0.54	93.28 ± 0.46	92.96 ± 0.56	93.11 ± 0.49
	94.08 ± 0.46	94.46 ± 0.51	94.24 ± 0.41	94.24 ± 0.45
IBRA-PCA	94.60 ± 0.45	95.02 ± 0.40	94.93 ± 0.45	94.97 ± 0.41
	96.01 ± 0.38	**96.24 ± 0.34**	**96.17 ± 0.33**	**96.20 ± 0.33**
IBRA-PLS-DA	**95.55 ± 0.42**	**95.85 ± 0.40**	**95.79 ± 0.44**	**95.81 ± 0.39**
	95.98 ± 0.66	96.27 ± 0.64	96.07 ± 0.72	96.16 ± 0.66
IBRA-GSS	95.24 ± 0.36	95.67 ± 0.36	95.45 ± 0.39	95.56 ± 0.37
	94.30 ± 0.61	94.78 ± 0.58	94.55 ± 0.60	94.65 ± 0.58

Table B.3: Feature selection method comparison on 75% of the Kochia dataset ($k = 10$).

Method	OA	Prec	Rec	F1
FNGBS	80.44 ± 1.36	81.28 ± 1.77	80.05 ± 1.28	80.58 ± 1.46
	81.92 ± 0.90	82.76 ± 1.25	81.23 ± 0.67	81.88 ± 0.85
PLS-DA	80.63 ± 1.51	81.66 ± 1.98	80.06 ± 1.42	80.73 ± 1.60
	83.91 ± 1.58	84.65 ± 1.67	83.70 ± 1.11	84.12 ± 1.63
OCF	85.60 ± 1.00	86.62 ± 0.88	85.77 ± 1.11	86.14 ± 0.95
	88.05 ± 0.87	88.90 ± 0.76	87.98 ± 1.13	88.38 ± 0.93
SR-SSIM	81.74 ± 1.66	83.31 ± 1.34	82.90 ± 1.71	83.07 ± 1.51
	84.41 ± 1.22	85.42 ± 1.19	85.34 ± 1.27	85.35 ± 1.21
HAGRID	87.53 ± 0.44	88.47 ± 0.57	87.77 ± 0.65	88.08 ± 0.49
	88.73 ± 0.78	89.722 ± 0.69	88.98 ± 0.99	89.30 ± 0.79
IBRA-PCA	79.58 ± 1.42	79.86 ± 1.32	79.27 ± 1.88	79.49 ± 1.57
	87.06 ± 1.05	87.70 ± 1.05	86.57 ± 1.23	87.06 ± 1.12
IBRA-PLS-DA	87.21 ± 1.01	88.21 ± 1.04	87.57 ± 1.10	87.81 ± 1.03
	88.20 ± 0.97	89.24 ± 0.91	88.43 ± 1.08	88.76 ± 0.98
IBRA-GSS	**89.39 ± 0.70**	**90.12 ± 0.74**	**89.54 ± 0.71**	**89.77 ± 0.70**
	89.69 ± 0.89	**90.55 ± 0.66**	**89.79 ± 1.03**	**90.13 ± 0.84**

Table B.4: Feature selection method comparison on 50% of the Kochia dataset ($k = 6$).

Method	OA	Prec	Rec	F1
FNGBS	88.68 ± 1.13	89.76 ± 1.08	88.85 ± 1.18	89.25 ± 1.11
	88.94 ± 0.87	89.77 ± 0.79	89.03 ± 0.89	89.35 ± 0.81
PLS-DA	86.90 ± 1.39	88.41 ± 1.08	87.23 ± 1.62	87.24 ± 1.36
	86.91 ± 1.34	87.58 ± 1.22	86.97 ± 1.45	87.23 ± 1.32
OCF	89.63 ± 1.33	90.68 ± 1.07	90.05 ± 1.27	90.32 ± 1.17
	89.15 ± 1.08	90.16 ± 1.06	89.54 ± 1.14	89.81 ± 1.09
SR-SSIM	86.58 ± 1.58	87.88 ± 1.22	87.37 ± 1.73	87.58 ± 1.49
	87.82 ± 1.24	88.92 ± 1.27	88.44 ± 1.33	88.63 ± 1.28
HAGRID	87.11 ± 1.42	88.45 ± 1.24	87.38 ± 1.51	87.84 ± 0.71
	88.41 ± 1.31	89.10 ± 1.29	89.65 ± 1.16	88.84 ± 1.21
IBRA-PCA	79.07 ± 2.19	79.61 ± 2.14	78.93 ± 2.70	79.17 ± 2.42
	92.13 ± 1.02	92.51 ± 0.88	92.21 ± 1.09	92.34 ± 0.98
IBRA-PLS-DA	**92.20 ± 0.86**	**92.66 ± 0.97**	**92.59 ± 0.77**	**92.61 ± 0.84**
	92.55 ± 0.87	**93.20 ± 0.73**	**92.75 ± 0.94**	**92.95 ± 0.84**
IBRA-GSS	91.36 ± 0.51	92.18 ± 0.48	91.97 ± 0.61	92.05 ± 0.52
	90.10 ± 0.74	90.91 ± 0.62	90.88 ± 0.84	90.87 ± 0.69

Table B.5: Feature selection method comparison on 50% of the Kochia dataset ($k = 8$).

Method	OA	Prec	Rec	F1
FNGBS	89.49 ± 0.68	90.26 ± 0.80	89.63 ± 0.71	89.89 ± 0.71
	90.27 ± 0.64	90.99 ± 0.75	90.18 ± 0.70	90.54 ± 0.68
PLS-DA	89.79 ± 1.14	90.69 ± 1.37	89.91 ± 0.99	90.24 ± 1.13
	91.44 ± 0.91	92.21 ± 1.01	91.42 ± 1.01	91.76 ± 0.97
OCF	90.76 ± 0.89	91.79 ± 0.80	90.58 ± 0.91	91.11 ± 0.82
	90.35 ± 0.70	91.20 ± 0.72	90.29 ± 0.68	90.69 ± 0.67
SR-SSIM	88.83 ± 1.77	90.08 ± 1.59	89.63 ± 1.76	89.82 ± 1.69
	89.57 ± 1.02	90.51 ± 1.09	90.39 ± 0.79	90.41 ± 0.94
HAGRID	91.06 ± 0.75	91.62 ± 0.98	91.15 ± 0.56	91.36 ± 0.71
	91.79 ± 0.49	92.33 ± 0.54	91.77 ± 0.70	92.01 ± 0.58
IBRA-PCA	92.74 ± 0.71	93.28 ± 0.53	93.07 ± 0.77	93.16 ± 0.64
	94.41 ± 0.53	**94.73 ± 0.34**	**94.51 ± 0.59**	**94.61 ± 0.51**
IBRA-PLS-DA	**93.52 ± 1.06**	**94.00 ± 1.01**	**93.64 ± 1.14**	**93.79 ± 1.06**
	94.19 ± 0.59	94.72 ± 0.59	94.21 ± 0.73	94.45 ± 0.64
IBRA-GSS	93.19 ± 0.87	93.82 ± 0.79	93.47 ± 0.76	93.62 ± 0.75
	92.61 ± 0.74	93.40 ± 0.56	92.68 ± 0.78	93.00 ± 0.64

Table B.6: Feature selection method comparison on 50% of the Kochia dataset ($k = 10$).

Method	OA	Prec	Rec	F1
FNGBS	77.39 ± 1.28	78.99 ± 1.40	76.47 ± 1.51	77.37 ± 1.37
	77.51 ± 1.43	78.87 ± 1.54	76.46 ± 1.59	77.42 ± 1.42
PLS-DA	77.01 ± 1.61	78.34 ± 2.23	76.38 ± 1.52	77.09 ± 1.67
	83.91 ± 1.58	84.65 ± 1.67	83.70 ± 1.11	84.12 ± 1.63
OCF	81.61 ± 1.17	82.74 ± 1.12	81.49 ± 1.46	82.02 ± 1.21
	83.11 ± 1.22	84.04 ± 1.10	82.88 ± 1.48	83.34 ± 1.21
SR-SSIM	78.23 ± 1.52	79.91 ± 1.46	79.28 ± 1.73	79.52 ± 1.52
	80.13 ± 1.68	81.39 ± 1.95	80.88 ± 1.38	81.04 ± 1.64
HAGRID	82.67 ± 1.20	83.98 ± 1.18	82.62 ± 1.38	83.14 ± 1.25
	83.77 ± 0.89	85.02 ± 1.21	83.65 ± 0.92	84.20 ± 0.85
IBRA-PCA	76.48 ± 1.74	76.83 ± 2.08	76.10 ± 1.99	76.31 ± 1.93
	82.71 ± 1.28	83.41 ± 1.25	82.27 ± 1.51	82.69 ± 1.29
IBRA-PLS-DA	84.52 ± 1.20	85.43 ± 1.29	84.61 ± 1.42	84.94 ± 1.28
	83.58 ± 1.54	84.51 ± 1.98	83.96 ± 1.49	84.15 ± 1.71
IBRA-GSS	**84.03 ± 0.85**	**85.38 ± 0.56**	**84.05 ± 0.94**	**84.56 ± 0.74**
	84.96 ± 1.19	**85.24 ± 1.36**	**85.01 ± 1.29**	**85.46 ± 1.19**

Table B.7: Feature selection method comparison on 25% of the Kochia dataset ($k = 6$).

Method	OA	Prec	Rec	F1
FNGBS	84.72 ± 0.55	93.63 ± 0.55	93.10 ± 0.46	93.34 ± 0.50
	93.12 ± 0.32	93.69 ± 0.28	93.13 ± 0.46	93.39 ± 0.34
PLS-DA	92.31 ± 0.83	93.00 ± 0.81	92.73 ± 0.89	92.86 ± 0.83
	90.53 ± 1.65	91.70 ± 1.37	91.01 ± 1.72	91.32 ± 1.57
OCF	93.05 ± 0.90	93.86 ± 0.80	93.43 ± 0.84	93.62 ± 0.83
	92.89 ± 0.71	93.45 ± 0.64	93.21 ± 0.67	93.32 ± 0.65
SR-SSIM	83.14 ± 1.29	84.59 ± 1.35	83.85 ± 1.22	84.13 ± 1.21
	84.79 ± 1.00	86.29 ± 1.49	84.91 ± 0.80	85.51 ± 1.08
HAGRID	91.59 ± 1.25	92.52 ± 1.05	91.88 ± 1.24	92.17 ± 1.16
	92.82 ± 0.79	93.41 ± 0.69	92.92 ± 0.91	93.15 ± 0.80
IBRA-PCA	83.75 ± 2.40	84.33 ± 2.48	83.77 ± 2.43	84.02 ± 2.45
	95.04 ± 0.70	**95.35 ± 0.60**	**95.21 ± 0.81**	**95.27 ± 0.70**
IBRA-PLS-DA	**95.46 ± 0.53**	**95.91 ± 0.46**	**95.63 ± 0.49**	**95.76 ± 0.47**
	95.42 ± 0.60	95.85 ± 0.52	95.62 ± 0.57	95.72 ± 0.53
IBRA-GSS	94.79 ± 0.65	95.28 ± 0.55	95.20 ± 0.67	95.23 ± 0.61
	94.08 ± 0.60	94.67 ± 0.57	94.54 ± 0.54	94.60 ± 0.54

Table B.8: Feature selection method comparison on 25% of the Kochia dataset ($k = 8$).

Method	OA	Prec	Rec	F1
FNGBS	84.17 ± 1.14	85.57 ± 1.30	83.86 ± 1.07	84.57 ± 1.11
	85.37 ± 1.43	86.64 ± 1.31	85.07 ± 1.58	85.70 ± 1.47
PLS-DA	85.71 ± 1.06	87.26 ± 1.11	85.45 ± 1.03	86.17 ± 1.05
	86.32 ± 1.25	87.64 ± 0.90	86.09 ± 1.35	86.75 ± 1.15
OCF	84.37 ± 1.26	85.71 ± 1.30	84.35 ± 1.09	84.91 ± 1.14
	84.00 ± 1.73	85.44 ± 1.71	83.97 ± 1.63	84.55 ± 1.62
SR-SSIM	84.25 ± 1.21	86.08 ± 1.44	85.03 ± 0.91	85.42 ± 1.04
	84.77 ± 1.61	86.25 ± 1.65	85.39 ± 1.31	85.72 ± 1.42
HAGRID	85.33 ± 0.98	86.86 ± 1.35	84.98 ± 0.91	85.73 ± 0.99
	86.14 ± 1.17	87.45 ± 1.18	85.97 ± 1.06	85.57 ± 1.10
IBRA-PCA	87.92 ± 1.02	88.94 ± 0.94	88.07 ± 1.11	88.45 ± 1.00
	90.14 ± 1.09	**90.79 ± 1.14**	**90.20 ± 1.18**	**90.46 ± 1.14**
IBRA-PLS-DA	**89.90 ± 1.10**	**90.72 ± 1.31**	**90.17 ± 0.95**	**90.39 ± 1.08**
	89.68 ± 1.20	90.65 ± 1.23	89.86 ± 1.03	90.21 ± 1.10
IBRA-GSS	87.49 ± 1.32	88.61 ± 1.51	87.57 ± 1.15	87.96 ± 1.21
	86.99 ± 1.41	88.55 ± 1.30	86.88 ± 1.49	87.55 ± 1.32

Table B.9: Feature selection method comparison on 25% of the Kochia dataset ($k = 10$).

B.2 Indian Pines Dataset

Method	OA	Prec	Rec	F1
PLS-DA	94.46 ± 3.38	95.36 ± 3.25	94.11 ± 2.50	95.57 ± 3.09
	94.42 ± 4.42	95.48 ± 3.23	93.51 ± 5.89	94.18 ± 5.25
OCF	94.36 ± 1.20	95.47 ± 1.20	94.90 ± 1.14	95.07 ± 1.21
	94.24 ± 3.48	95.74 ± 2.52	95.13 ± 2.17	97.31 ± 2.47
SR-SSIM	95.13 ± 1.13	96.05 ± 0.96	95.52 ± 1.36	95.69 ± 1.11
	94.56 ± 1.57	95.72 ± 1.25	95.50 ± 1.77	95.46 ± 1.52
HAGRID	94.47 ± 0.94	94.99 ± 1.27	94.25 ± 1.42	94.51 ± 1.31
	94.93 ± 1.42	95.45 ± 1.52	95.09 ± 1.88	95.15 ± 1.72
FNGBS	95.93 ± 1.00	96.51 ± 1.06	96.64 ± 0.87	96.51 ± 0.98
	95.49 ± 1.62	95.34 ± 1.88	96.16 ± 1.90	95.90 ± 2.12
IBRA-PCA	91.51 ± 0.84	93.99 ± 1.00	91.50 ± 1.67	92.29 ± 1.21
	94.74 ± 1.64	95.55 ± 1.33	97.75 ± 1.08	95.54 ± 1.03
IBRA-PLS-DA	92.89 ± 5.01	94.67 ± 3.37	93.99 ± 4.21	93.87 ± 4.74
	94.24 ± 2.71	95.63 ± 1.77	94.80 ± 2.72	94.83 ± 2.94
IBRA-GSS	**96.72 ± 0.55**	**97.35 ± 0.56**	**96.98 ± 1.01**	**97.09 ± 0.84**
	96.63 ± 1.07	**96.85 ± 0.96**	**96.99 ± 0.86**	**96.86 ± 0.84**

Table B.10: Feature selection method comparison on 75% of the IP dataset.

Method	OA	Prec	Rec	F1
PLS-DA	92.01 ± 0.70	92.77 ± 1.04	90.04 ± 1.80	91.00 ± 1.13
	92.67 ± 0.92	93.84 ± 0.93	91.97 ± 1.63	92.70 ± 1.20
OCF	90.44 ± 1.22	92.31 ± 0.93	91.06 ± 1.46	91.44 ± 1.12
	91.58 ± 1.25	93.72 ± 0.92	93.13 ± 1.17	93.28 ± 1.00
SR-SSIM	90.37 ± 4.09	90.90 ± 5.19	89.97 ± 2.51	89.58 ± 4.29
	89.65 ± 3.56	91.27 ± 3.46	90.91 ± 2.53	90.67 ± 3.22
HAGRID	90.97 ± 0.90	92.51 ± 0.88	90.62 ± 1.63	91.23 ± 1.21
	91.89 ± 1.06	93.39 ± 0.64	92.58 ± 1.11	92.80 ± 0.73
FNGBS	92.24 ± 0.66	94.50 ± 1.15	93.52 ± 1.20	93.81 ± 1.12
	93.35 ± 0.81	94.93 ± 0.77	93.80 ± 1.51	94.16 ± 1.10
IBRA-PCA	85.92 ± 1.16	89.68 ± 1.75	85.30 ± 2.16	86.36 ± 1.12
	90.95 ± 1.00	92.89 ± 1.34	91.77 ± 1.60	92.07 ± 1.44
IBRA-PLS-DA	86.98 ± 4.76	90.22 ± 3.54	87.85 ± 4.67	88.12 ± 5.49
	89.81 ± 5.96	92.40 ± 4.45	91.10 ± 4.82	90.80 ± 6.06
IBRA-GSS	**94.41 ± 0.43**	**95.29 ± 0.46**	**94.37 ± 1.26**	**94.69 ± 0.87**
	93.73 ± 1.03	**95.22 ± 0.90**	**94.17 ± 1.41**	**94.56 ± 1.04**

Table B.11: Feature selection method comparison on 50% of the IP dataset.

Method	OA	Prec	Rec	F1
PLS-DA	84.98 ± 1.32	87.51 ± 2.37	78.85 ± 4.27	80.97 ± 3.92
	85.83 ± 1.49	87.36 ± 3.11	79.88 ± 3.44	81.72 ± 3.13
OCF	82.23 ± 2.47	83.28 ± 2.83	76.71 ± 3.67	78.24 ± 3.34
	83.51 ± 1.26	85.84 ± 3.53	80.40 ± 3.87	81.70 ± 3.67
SR-SSIM	82.44 ± 9.21	83.65 ± 8.71	79.40 ± 8.46	79.63 ± 9.25
	81.87 ± 9.67	83.58 ± 8.92	78.79 ± 9.56	79.53 ± 10.59
HAGRID	82.58 ± 1.71	82.43 ± 2.68	77.63 ± 3.32	78.68 ± 3.21
	83.73 ± 2.20	85.62 ± 2.90	80.89 ± 2.91	81.73 ± 1.00
FNGBS	87.03 ± 1.18	87.45 ± 2.29	83.74 ± 3.56	84.47 ± 3.00
	86.92 ± 0.91	87.72 ± 1.75	84.37 ± 2.38	84.72 ± 1.80
IBRA-PCA	78.67 ± 1.42	82.77 ± 3.68	75.65 ± 3.80	77.05 ± 3.56
	84.86 ± 1.35	97.04 ± 2.25	82.99 ± 2.76	83.95 ± 2.67
IBRA-PLS-DA	82.02 ± 9.11	84.76 ± 8.84	82.80 ± 5.54	82.24 ± 8.47
	83.97 ± 8.33	86.55 ± 4.98	82.14 ± 8.05	82.56 ± 7.86
IBRA-GSS	**87.23 ± 1.91**	**90.20 ± 2.56**	**84.48 ± 3.92**	**86.17 ± 3.62**
	88.41 ± 1.25	**90.49 ± 2.18**	**85.87 ± 3.05**	**87.10 ± 2.76**

Table B.12: Feature selection method comparison on 25% of the IP dataset.

B.3 Salinas Dataset

Method	OA	Prec	Rec	F1
PLS-DA	97.33 ± 0.18	98.68 ± 0.15	98.80 ± 0.10	98.74 ± 0.12
	98.17 ± 0.15	99.07 ± 0.03	99.16 ± 0.08	99.11 ± 0.05
OCF	98.07 ± 0.10	99.03 ± 0.06	99.11 ± 0.07	99.07 ± 0.06
	98.11 ± 0.17	99.05 ± 0.12	99.13 ± 0.09	99.09 ± 0.10
SR-SSIM	97.43 ± 0.22	98.90 ± 0.13	98.89 ± 0.12	98.89 ± 0.12
	97.85 ± 0.18	99.06 ± 0.12	99.04 ± 0.14	99.05 ± 0.13
HAGRID	98.05 ± 0.19	99.03 ± 0.07	99.13 ± 0.09	99.08 ± 0.08
	98.03 ± 0.17	99.02 ± 0.09	99.10 ± 0.11	99.06 ± 0.00
FNGBS	97.99 ± 0.12	99.13 ± 0.07	99.15 ± 0.08	99.14 ± 0.07
	98.08 ± 0.17	99.17 ± 0.12	99.15 ± 0.10	99.16 ± 0.11
IBRA-PCA	94.87 ± 0.96	97.35 ± 0.84	97.11 ± 1.19	97.16 ± 1.12
	97.96 ± 0.23	99.14 ± 0.11	99.15 ± 0.10	99.14 ± 0.10
IBRA-PLS-DA	95.32 ± 0.14	97.86 ± 0.15	98.00 ± 0.10	97.92 ± 0.11
	97.64 ± 0.19	98.93 ± 0.12	98.99 ± 0.08	98.96 ± 0.10
IBRA-GSS	**98.91 ± 0.12**	**99.45 ± 0.07**	**99.49 ± 0.06**	**99.47 ± 0.06**
	98.94 ± 0.08	**99.47 ± 0.06**	**99.50 ± 0.07**	**99.48 ± 0.06**

Table B.13: Feature selection method comparison on 75% of the SA dataset.

Method	OA	Prec	Rec	F1
PLS-DA	96.71 ± 0.24	98.40 ± 0.16	98.50 ± 0.13	98.45 ± 0.14
	97.30 ± 0.27	98.64 ± 0.13	98.72 ± 0.15	98.67 ± 0.13
OCF	97.41 ± 0.11	98.64 ± 0.13	98.70 ± 0.10	98.67 ± 0.10
	97.40 ± 0.12	98.67 ± 0.11	98.75 ± 0.10	98.71 ± 0.09
SR-SSIM	96.71 ± 0.23	98.56 ± 0.10	98.53 ± 0.13	98.54 ± 0.11
	97.06 ± 0.32	98.70 ± 0.16	98.68 ± 0.16	98.69 ± 0.16
HAGRID	97.32 ± 0.17	98.65 ± 0.09	98.75 ± 0.09	98.70 ± 0.09
	97.27 ± 0.24	98.61 ± 0.11	98.71 ± 0.10	98.66 ± 0.10
FNGBS	97.12 ± 0.15	98.71 ± 0.08	98.72 ± 0.07	98.71 ± 0.06
	97.39 ± 0.18	98.87 ± 0.07	98.84 ± 0.09	98.85 ± 0.07
IBRA-PCA	93.79 ± 1.21	96.90 ± 0.85	96.48 ± 1.45	96.63 ± 1.23
	97.20 ± 0.26	98.81 ± 0.09	98.81 ± 0.10	98.81 ± 0.09
IBRA-PLS-DA	94.10 ± 0.38	97.18 ± 0.32	97.37 ± 0.19	97.26 ± 0.24
	96.87 ± 0.18	98.54 ± 0.10	98.59 ± 0.10	98.56 ± 0.09
IBRA-GSS	**98.35 ± 0.23**	**99.15 ± 0.12**	**99.16 ± 0.09**	**99.15 ± 0.10**
	98.27 ± 0.18	**99.12 ± 0.15**	**99.13 ± 0.08**	**99.12 ± 0.11**

Table B.14: Feature selection method comparison on 50% of the SA dataset.

Method	OA	Prec	Rec	F1
PLS-DA	94.93 ± 0.37	97.28 ± 0.33	97.57 ± 0.17	97.40 ± 0.25
	95.71 ± 0.74	97.65 ± 0.38	97.82 ± 0.37	97.72 ± 0.37
OCF	96.29 ± 0.30	97.90 ± 0.21	97.41 ± 0.27	97.91 ± 0.28
	96.36 ± 0.26	98.03 ± 0.19	98.00 ± 0.17	98.01 ± 0.17
SR-SSIM	95.39 ± 0.31	97.80 ± 0.23	97.63 ± 0.27	97.70 ± 0.25
	95.72 ± 0.42	97.93 ± 0.24	97.83 ± 0.26	97.87 ± 0.24
HAGRID	96.00 ± 0.28	97.70 ± 0.20	97.94 ± 0.26	97.81 ± 0.22
	95.92 ± 0.32	97.71 ± 0.21	97.90 ± 0.22	97.80 ± 0.20
FNGBS	96.10 ± 0.24	98.13 ± 0.18	98.19 ± 0.22	98.15 ± 0.19
	96.26 ± 0.34	98.20 ± 0.24	98.24 ± 0.23	98.21 ± 0.22
IBRA-PCA	92.54 ± 0.52	96.14 ± 0.37	95.89 ± 0.53	95.98 ± 0.46
	96.07 ± 0.37	98.23 ± 0.22	98.15 ± 0.25	98.18 ± 0.23
IBRA-PLS-DA	92.44 ± 0.47	96.13 ± 0.37	96.12 ± 0.32	96.10 ± 0.32
	95.42 ± 0.29	97.98 ± 0.19	97.90 ± 0.22	97.93 ± 0.19
IBRA-GSS	**97.09 ± 0.27**	**98.59 ± 0.18**	**98.46 ± 0.21**	**98.51 ± 0.18**
	97.06 ± 0.43	**98.49 ± 0.27**	**98.44 ± 0.32**	**98.46 ± 0.29**

Table B.15: Feature selection method comparison on 25% of the SA dataset.