PROGRESSIVELY EXPANDED NEURAL NETWORK FOR AUTOMATIC MATERIAL IDENTIFICATION IN HYPERSPECTRAL IMAGERY

Dissertation

Submitted to

The School of Engineering of the

UNIVERSITY OF DAYTON

In Partial Fulfillment of the Requirements for

The Degree of

Doctor of Philosophy in Engineering

By

Sidike Paheding

UNIVERSITY OF DAYTON

Dayton, Ohio

December, 2016
PROGRESSIVELY EXPANDED NEURAL NETWORK FOR AUTOMATIC
MATERIAL IDENTIFICATION IN HYPERSPECTRAL IMAGERY

Name: Paheding, Sidike

APPROVED BY:

Vijayan K. Asari, Ph.D.
Advisor Committee Chairman
Professor, Department of Electrical Engineering

Raul Ordonez, Ph.D.
Committee Member
Professor, Department of Electrical Engineering

Eric J. Balster, Ph.D.
Committee Member
Associate Professor, Department of Electrical Engineering

Muhammad N. Islam, Ph.D.
Committee Member
Professor, Department of Mathematics

Robert J. Wilkens, Ph.D., P.E.
Associate Dean for Research and Innovation, Professor
School of Engineering

Eddy Rojas, Ph.D., M.A., P. E.
Dean, School of Engineering
© Copyright by
Sidike Paheding
All rights reserved
2016
ABSTRACT

PROGRESSIVELY EXPANDED NEURAL NETWORK FOR AUTOMATIC MATERIAL IDENTIFICATION IN HYPERSPECTRAL IMAGERY

Name: Paheding, Sidike
University of Dayton
Advisor: Dr. Vijayan K. Asari

The science of hyperspectral remote sensing focuses on the exploitation of the spectral signatures of various materials to enhance capabilities including object detection, recognition, and material characterization. Hyperspectral imagery (HSI) has been extensively used for object detection and identification applications since it provides plenty of spectral information to uniquely identify materials by their reflectance spectra.

HSI-based object detection algorithms can be generally classified into stochastic and deterministic approaches. Deterministic approaches are comparatively simple to apply since it is usually based on direct spectral similarity such as spectral angles or spectral correlation. In contrast, stochastic algorithms require statistical modeling and estimation for target class and non-target class. Over the decades, many single class object detection methods have been proposed in the literature, however, deterministic multiclass object detection in HSI has not been explored. In this work, we propose a deterministic multiclass object detection scheme, named class-associative spectral fringe-adjusted joint transform correlation.
Human brain is capable of simultaneously processing high volumes of multi-modal data received every second of the day. In contrast, a machine sees input data simply as random binary numbers. Although machines are computationally efficient, they are inferior when comes to data abstraction and interpretation. Thus, mimicking the learning strength of human brain has been current trend in artificial intelligence. In this work, we present a biological inspired neural network, named progressively expanded neural network (PEN Net), based on nonlinear transformation of input neurons to a feature space for better pattern differentiation. In PEN Net, discrete fixed excitations are disassembled and scattered in the feature space as a nonlinear line. Each disassembled element on the line corresponds to a pattern with similar features. Unlike the conventional neural network where hidden neurons need to be iteratively adjusted to achieve better accuracy, our proposed PEN Net does not require hidden neurons tuning which achieves better computational efficiency, and it has also shown superior performance in HSI classification tasks compared to the state-of-the-arts.

Spectral-spatial features based HSI classification framework has shown stronger strength compared to spectral-only based methods. In our lastly proposed technique, PEN Net is incorporated with multiscale spatial features (i.e., multiscale complete local binary pattern) to perform a spectral-spatial classification of HSI. Several experiments demonstrate excellent performance of our proposed technique compared to the more recent developed approaches.
DEDICATED

To

My respectiveful parents — Rexat Sidik and Patima Ali

(I hope that this achievement will fulfill the dream you had for me when I was a kid)

My beloved wife — Shamsiya Anwar

(I hope that this accomplishment will be our ever-standing glory)

My lovely daughter — Faherya Paherdin

(I hope that this gift will inspire you along the way of your success)
ACKNOWLEDGMENTS

First and foremost, I would like to express my deepest gratitude and warmest affection to Professor Vijayan K. Asari who has been a fountain of knowledge and inspiration.

I would like to thank my Ph.D. advisory committee members, Dr. Raul Ordonez, Dr. Eric J. Balster and Dr. Muhammad N. Islam for their valuable suggestions. My grateful appreciation is extended to UD Vision Lab colleagues, all the faculty, staff and graduate students in this department, from whom I learned so much.

I also would like to thank my uncle Dr. Kurex Sidik for his support, advice and inspiration.

Finally, I would like to express my deep love and appreciation to my parents, my wife, my sister and my daughter for their love, concern and encouragement throughout my study.
# TABLE OF CONTENTS

ABSTRACT ................................................. iii
DEDICATION .............................................. v
ACKNOWLEDGMENTS ................................. vi
LIST OF FIGURES ................................. ix
LIST OF TABLES .................................... xii
NOMENCLATURE ..................................... xiv

I. INTRODUCTION ......................................... 1
   1.1 Hyperspectral Imagery (HSI) ................. 1
   1.2 HSI Processing .................................. 3
   1.3 Scientific Contributions ....................... 4
   1.4 Dissertation Outline ............................ 5

II. RELATED WORK ................................. 6
   2.1 Object Detection in HSI ....................... 6
   2.2 Object Classification in HSI ............... 10
   2.3 Spectral-Spatial Classification of Hyperspectral Data .................. 16

III. CLASS-ASSOCIATIVE SPECTRAL FRINGE-ADJUSTED JOINT TRANSFORM CORRELATION FOR MULTICLASS OBJECT DETECTION IN HSI ................. 18
   3.1 Introduction and Motivation .................. 18
   3.2 Theoretical Analysis ......................... 21
   3.3 Evaluation ................................ 28
      3.3.1 Dataset Description and Test Setups .......... 29
      3.3.2 Experimentation on Single-Class Object Detection ...... 31
      3.3.3 Experimentation on Multiclass Object Detection ....... 37
# LIST OF FIGURES

1.1  HSI illustration. Note that each single band is a grayscale image, showing colors in front face in HSI for contextual illustration. .............................. 2

1.2  Illustration of challenges in HSI Processing: Spectral variability and subpixel target presence. ................................................................. 3

1.3  A typical algorithm step for HSI processing. ................................. 4

3.1  Block diagram of the proposed pattern recognition scheme for the case of two reference spectra. JPS: joint power spectrum; CJPS: combination of JPS; CSGFAF: class-associative spectral generalized fringe-adjusted filter. ........................................ 21

3.2  Reference and input spectral signatures. (a) Reference 1 and 2 spectral signatures. (b) Reference 1 and reference 2 with a false input. ...................... 27

3.3  Correlation outputs with spectral signatures. (a) Correlation output of CSFJTC for an input as reference 1. (b) Correlation output of CSFJTC for an input as reference 2. (c) Correlation output of CSFJTC for a false input. For parameters in CSFJTC, $a_1 = a_2 = 0.5$ and $m = 2$. ..................................................... 28

3.4  Indian Pines data set. (a) False-color composite. (b) Ground truth. ........... 30

3.5  Salinas Scene data set. (a) False-color composite. (b) Ground truth. ........... 30

3.6  MROC for the Indian Pines dataset. ................................................... 33

3.7  MROC for the Salinas dataset. ............................................................ 34

3.8  Detection in single-class case: (a) Input scene, and (b) locations of the inserted targets. ................................................................. 36

3.9  3D correlation output for ten desired targets in the scene. ...................... 37
3.10 Detection in multiclass case: (a) Locations of the inserted targets, and (b) Corre-
lation output for multiple targets in (a), where T1 represents locations of all target
class 1, and T2 indicates locations of all target class 2. ......................... 40

3.11 A sample application of CSFJTC for specific material detection. ............. 41

4.1 Conventional NN model versus proposed NN model for a connection of two neurons. 47

4.2 Conventional NN model versus proposed PEN Net for a case of single hidden layer
model with input pattern with 3 excitations. ........................................ 49

4.3 Training process of multi-hidden layer architecture of PEN Net. ............. 53

4.4 Testing process of multi-hidden layer architecture of PEN Net. ............. 53

4.5 Proposed spectral feature based classification framework using PEN Net. ...... 55

4.6 Illustration of LBP calculation for $P = 8, R = 1$. ............................ 56

4.7 An example of LBP computation. ..................................................... 57

4.8 Sample of LBP coded image on HSI bands. ..................................... 58

4.9 Effect of image rotation on circular neighborhoods. .............................. 59

4.10 An example for CLBP components. (a) A $3 \times 3$ sample block; (b) the local differ-
ences; (c) the sign and (d) magnitude components. ................................ 61

4.11 An example of $CLBP_S$ and $CLBP_M$ coded images. (a) Input image (single band),
(b) $CLBP_S$ coded image, and (c) $CLBP_M$ coded image. ....................... 61

4.12 An example of 3-scale CLBP operator with $P = 8, R_1 = 1, R_2 = 2$, and $R_3 = 3$. 62

4.13 An overview of spatial feature based classification framework. ............................. 63

4.14 Graphically illustration of the proposed spatial feature based classification framework. 64

4.15 Illustration of spatial feature extraction using MS-CLBP. ........................ 65

4.16 Proposed spectral-spatial classification framework. ............................. 66

5.1 The University of Pavia dataset. (a) False-color composite. (b) Ground truth. ... 69
5.2 Classification maps of different methods on the Indian Pines dataset. . . . . . . . 78

5.3 Classification maps of different methods on the University of Pavia dataset. . . . 79

5.4 Classification maps of different methods on the Salinas dataset. . . . . . . . . . . 80

5.5 OA of PEN Net (SHL model), ELM, KNN, LDA versus the number of training
samples in (a) Indian Pines dataset, (b) University of Pavia dataset, and (c) Salinas
dataset. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 82

5.6 Effect of multiscale features in MS-CLBP. . . . . . . . . . . . . . . . . . . . . . . 83

5.7 Computation cost of multiscale features. . . . . . . . . . . . . . . . . . . . . . . . 83
LIST OF TABLES

3.1 Quantitative analysis of the CSFJTC outputs corresponding to Figs. 3.2 and 3.3. . . 29

3.2 AUROC comparison for the Indian Pines dataset. The best accuracy for each class and the best overall performance is highlighted in boldface. ............................. 35

3.3 AUROC comparison for the Salinas dataset. ............................................. 36

3.4 AUROC comparison for multiclass object detection in HSI. ......................... 38

5.1 Specifications of hyperspectral datasets used in the experiments. .................... 69

5.2 The number of training and testing samples distribution in the Indian Pines dataset. 70

5.3 The number of training and testing samples distribution in the University of Pavia dataset. ................................................................. 71

5.4 The number of training and testing samples distribution in the Salinas dataset. .... 71

5.5 Accuracy of function type with respect to the number of terms in the Indian Pines dataset. ................................................................. 73

5.6 Accuracy of function type with respect to the number of terms in the University of Pavia dataset. ................................................................. 74

5.7 Accuracy of function type with respect to the number of terms in the Salinas dataset. 74

5.8 Accuracy of every class (%), OA (%) for the Indian Pines dataset using 30% training samples. (The best accuracy is highlighted in boldface) ............... 75
5.9 Accuracy of every class (%), OA (%) for the University of Pavia dataset using 10% of training samples. ................................................................. 76

5.10 Accuracy of every class (%), OA (%) for the Salinas dataset using 10% training samples. ................................................................. 76

5.11 Performance Comparison of spectral-spatial features based HSI classification methods. ................................................................. 85
NOMENCLATURE

$\delta_{CEM}$ Constrained energy minimization (CEM) filter

$(P, R)$ The number of neighbors $(P)$ and radius $(R)$ in CLBP

$\alpha_k$ The $k$th term in the nonlinear expansion

$\hat{\beta}$ Output weight matrix computed by Moore-Penrose generalized inverse

$\hat{G}(u)$ FAF filtered JPS

$\tilde{H}(u)$ class-associative spectral generalized FAF (CSGFAF)

$\tilde{G}(u)$ CSGFAF filtered JPS

$b$ bias in the NN

$F(u)$ Fourier transform of the joint spectral signature

$H^\dagger$ Moore-Penrose generalized inverse of matrix $H$

$O(u)$ Joint power spectrum of $F(u)$

$r$ Reference/desired spectral signature

$w$ Weight matrix

$x$ Unknown/test spectral signature
Joint spectral signature

The $j$th coefficient in the JPS

Total number of class/ground category in the input HSI

Hidden layer activation function

Hidden layer output matrix

The number of spectral bands

Progressive expansion

The $i$th progressively expanded component

Local difference of the magnitude

Local difference of the sign

Contextual ELM

Complete local binary pattern

Convolutional neural network

Class-associative spectral fringe-adjusted joint transform correlation

Contextual SVM

Extreme learning machine

Fringe-adjusted filter

Guided filter based fine-tuning SAE

Hierarchical local receptive field based ELM
HSI  Hyperspectral imagery
JPS  Joint power spectrum
JTC  Joint transform correlation
KNN  K-nearest neighborhood
L-ELM Local receptive field based ELM
LDA  Linear discriminant analysis
MHL  Multi-hidden layer
MS-CLBP Multiscale CLBP
NN   Neural network
NRS  Nearest regularized subspace
OA   Overall accuracy
PCA  Principal component analysis
PEN Net Progressively expanded neural network
SADL Spatial-aware dictionary learning
SAE  Stacked autoencoder
SAM  Spectral angle mapper
SHL  Single hidden layer
SID  Spectral information divergence
SLFN Single hidden layer feed-forward neural network
SS Spectral-spatial

SS-MHL-PEN Spectral-spatial features based multi-hidden layer PEN Net

SS-SHL-PEN Spectral-spatial features based single hidden layer PEN Net

SVM Support vector machine
CHAPTER I

INTRODUCTION

1.1 Hyperspectral Imagery (HSI)

The science of hyperspectral remote sensing is based on taking a fraction of the electromagnetic spectrum and breaking it into very small bands for theoretical analysis and computations. Hyperspectral remote sensors collect image data simultaneously in dozens or hundreds of narrow and adjacent spectral bands. This builds a spectral cube as shown in Fig. 1.1, which is a three-dimensional array containing spatial information on the $x$-axis and $y$-axis and spectral information on the $z$-axis. The dimensional thickness of $z$ indicates the number of bands, in other words, each slice of hyperspectral cube corresponds to a certain wavelength. Thus, the combination of all wavelengths in a given spatial area builds complete spectral signatures for a specific material in the scene. The magnitude of the spectral signature is created by mapping a color to the intensity of the spectral response at different wavelengths. The different spectral signatures indicate unique materials being imaged, which also allows extraction of additional information that the human eye fails to capture with its receptors for red, green and blue.

Hyperspectral imagery (HSI) is produced by an instrument called an imaging spectrometer, which is used to acquire a spectrally-resolved image of an object. Specifically, it is made for the purpose of measuring the spectral content of light and electromagnetic energy. For HSI, the narrow
spectral channels of an imaging spectrometer form a continuous reflectance spectrum of the earth’s surface. The bandwidth of hyperspectral data typically ranges from 1 to 15 nanometers, while multispectral data typically contains 3 to 7 bands with bandwidths ranging from 50 to 120 nanometers, such as Landsat satellite sensors. Therefore, one of the primary advantages of hyperspectral images is that they provide higher spectral resolution than multispectral images. This merit assists in automatic target detection and classification in noisy backgrounds since objects vary uniquely from the natural background in absorbing and reflecting radiation at different wavelengths. In most cases, the targets can be differentiated and identified based on their spectral signature, this provides many real-life applications in life science, surveillance, agriculture, etc. However, considerable spectral variability as shown in Fig. 1.2, which refers to the phenomenon that spectra observed from samples of the same material will never be identical, and subpixel targets make HSI processing challenging. These undesired phenomenon may be due to atmospheric variations, illumination changes, sensor noise, etc. [1].

Figure 1.1: HSI illustration. Note that each single band is a grayscale image, showing colors in front face in HSI for contextual illustration.
1.2 HSI Processing

A common way to process HSI can be categorized into three distinct stages: dimensionality reduction, spatial/spectral content transformation, object detection or classification. Figure 1.3 shows an overview steps of this process. In the first stage, a dimensionality reduction technique is applied to remove data redundancy and segregate the noise in the data. Particularly, images are transformed into new bands to fulfill different purposes, such as decorrelation of spectral bands, reduction of noisy bands, and enhancement of certain features. This is a common process because HSI uses fine spectral resolution to increase the spectral information content about the target and it thus contains vast amounts of data, requiring computation-intensive processing and analysis. In the second stage of the process, spatial and spectral information from the stage one can be further processed to extract effective features that are the most capable of preserving class separability. In the last stage,
the extracted features are fed to an object detector or classifier for a variety of application purposes. This dissertation mainly focuses on object detection and classification algorithms in HSI.

![Figure 1.3: A typical algorithm step for HSI processing.](image)

### 1.3 Scientific Contributions

The novel scientific contributions in this dissertation are summarized as follows:

- Developed a new object detection algorithm that is capable of detecting multiple class of objects simultaneously from HSI with rapid speed and state-of-the-art accuracy. The developed method named **Class-associative Spectral Fringe-adjusted Joint Transform Correlation (CSFJTC)**.

- Developed a new biologically inspired machine learning algorithm, named **Progressively Expanded Neural Network (PEN Net)**, embeds nonlinear line transform that maps data to higher dimensional space for better pattern discrimination in HSI. The proposed neural network is developed both for single hidden layer model and multi-hidden layer model.
• Developed a new spectral-spatial features based HSI classification technique which contains novelty in effectively incorporating local textural features with multiple resolution in spatial domain to improve classification accuracy.

1.4 Dissertation Outline

The dissertation is structured as follows. Chapter II describes recent advances in HSI processing where several state-of-the-art techniques for object detection and classification are introduced. In Chapter III, we present our multiclass object detector (i.e., CSFJTC) theoretically and experimentally. Chapter IV first discusses the design of the proposed neural network model (i.e., PEN Net) in detail, and then introduces spectral-spatial features based HSI classification scheme. In Chapter V, we provide experimental results for HSI classification, as well as quantitative comparison with state-of-the-art methods. Finally, Chapter VI summarizes this dissertation and suggests future direction of this research.
There are two main applications of HSI that rely on the ability to distinguish material (targets) based on the spectral signatures and spatial information: target detection and classification.

2.1 Object Detection in HSI

In target detection applications, the objective is to search the pixels of an HSI for the occurrence of a specific target. Object detection algorithms can be generally classified into stochastic and deterministic approaches. Deterministic approaches are comparatively simple to apply since it is usually based on direct spectral similarity such as spectral angles [2] or spectral correlation [3]. In contrast, stochastic algorithms require statistical modeling and estimation for target class and non-target class [1], or look for pixels that are spectrally distinct from their neighbors without a priori information about the target [4]. In following, a brief review of several object detectors for HSI will be given.

Spectral Angle Mapper

The spectral angle mapper (SAM) [2] is an automated method for directly comparing image spectra with a known spectrum (usually determined in a lab or in the field with a spectrometer) or an endmember. SAM is a physically-based spectral finder that uses an many dimensional angle to
match pixels with reference spectra. The algorithm determines the spectral similarity between two spectra by calculating the angle between the spectra and treating them as vectors in a space with dimensionality equal to the number of bands \([2, 5]\). This technique is insensitive to illumination since the SAM algorithm uses only the vector direction instead of the vector length. Endmember spectra used by SAM can come from spectral libraries, or an averaged spectrum of a set of target pixels. This technique can be mathematically modeled as

\[
SAM(r, x) = \cos^{-1}\left(\frac{r^T x}{\sqrt{r^T r} \sqrt{x^T x}}\right)
\]  

(2.1)

where \(r, x\) are the spectral signatures of the reference pixel vector and a test pixel vector, respectively.

**Spectral Fringe-adjusted Joint Transform Correlation**

The basic concept of spectral fringe-adjusted joint transform correlation (SFJTC) is derived from the joint transform correlation (JTC) \([6]\), where a target signal is correlated with unknown input signals through the Fourier transform process. In case of hyperspectral pattern recognition for a single pixel object, the target signal can be viewed as a discrete 1-D vector to represent spectral signature. Incorporating the fringe-adjusted filter (FAF) \([7]\) in spectral JTC leads to the SFJTC. To seek the desired targets, all of the pixel spectra in hyperspectral data are individually correlated through the SFJTC process. To mathematically formulate this concept, consider a hyperspectral image with \(N\) pixels \(\{x_i\}_{i=1}^{N}\) in \(\mathbb{R}^l\), where \(l\) is the number of spectral bands and \(x_i = [x_{i1}, x_{i2}, \cdots, x_{il}]\) represents the \(i\)th \((i = 1, 2, \cdots, N)\) pixel spectrum and \(N\) is the total number of pixels. Further assume a row vector \(r = [r_1, r_2, \cdots, r_l]\) as the reference (pure target) spectral signature and let \(r\) and \(x_i\) be separated by a distance \(2d\) along the \(x\)-axis, then the joint spectral signature \(j_i\) can be expressed as

\[
J_i = r(x + d) + x_i(x - d).
\]  

(2.2)
From Eq. (2.2), it can be observed that each pixel vector in the hyperspectral data cube is individually jointed to the reference spectrum. Applying the Fourier transform to Eq. (2.2) yields

$$F_i(u) = |R(u)|\exp[j\phi_r(u) + jud] + |X_i(u)|\exp[j\phi_{x_i}(u) - jud]$$

(2.3)

where $F_i(u)$ is the Fourier transform of $j_i$, $|R(u)|$ and $|X_i(u)|$ are the amplitude; $\phi_r$ and $\phi_{x_i}$ are the phases of the Fourier transform of $r$ and $x_i$, respectively; and $u$ is a frequency-domain variable in the $x$-axis direction. The corresponding JPS can be calculated by

$$O_i(u) = |F_i(u)|^2 = F_i(u)F_i^*(u)$$

$$= |R(u)|^2 + |X_i(u)|^2 + |R(u)||X_i(u)|^*$$

$$\times \exp[j\phi_r(u) - j\phi_{x_i}(u) + jud] + |R(u)|^*|X_i(u)| \times \exp[j\phi_{x_i}(u) - j\phi_r(u) - jud]$$

(2.4)

where $^*$ denotes complex conjugate, $|R(u)|^2$ and $|X_i(u)|^2$ are the autocorrelation components of $r(x)$ and $x_i(x)$ respectively; and the last two terms are the cross-correlation components between the reference and input signatures. The inverse Fourier transform of the JPS in Eq. (2.4) yields

$$o_i(x) = r(x) \otimes r(x) + x_i(x) \otimes x_i(x) + r(x) \otimes x_i(x) * \delta(x - 2d) + x_i(x) \otimes r(x) * \delta(x + 2d)$$

(2.5)

where $\otimes$ and $*$ denote the correlation and convolution operations, respectively. The first two terms in Eq. (2.5) are autocorrelation terms, or zero-order diffractions at the origin of the output plane. These terms usually introduce false alarms to the system; thus, to avoid this issue, the power spectra of the input signature and the reference signature are subtracted from the JPS. This resulted in the modified JPS is expressed as

$$|P_i(u)|^2 = O_i(u) - |R(u)|^2 - |X_i(u)|^2$$

$$= |R(u)||X_i(u)|^* \times \exp[j\phi_r(u) - j\phi_{x_i}(u) + jud]$$

$$+ |R(u)|^*|X_i(u)| \times \exp[j\phi_{x_i}(u) - j\phi_r(u) - jud].$$

(2.6)

Performing the inverse Fourier transform of the modified JPS produces two cross-correlation components as follow

$$p_i(x) = r(x) \otimes x_i(x) * \delta(x - 2d) + x_i(x) \otimes r(x) * \delta(x + 2d).$$

(2.7)
The performance of the correlation output can be further improved by multiplying the modified JPS with the FAF before the final inverse Fourier transform. The FAF is characterized by the transfer function, which is defined as [7]

\[ H(u) = \frac{B(u)}{A(u) + |R(u)|^2} \]  

(2.8)

where \( A(u) \) and \( B(u) \) are either constants or functions of \( u \). When \( B(u) = 1 \) and \( |R(u)|^2 \gg A(u) \), the FAF becomes a perfect inverse filter. The FAF filtered JPS is given by

\[ \hat{G}_i(u) = H(u) \times |P_i(u)|^2. \]  

(2.9)

Finally, an inverse Fourier transform of the \( \hat{G}_i(u) \) yields the correlation output as

\[ \hat{g}_i(x) = F^{-1}\{H(u) \times |P_i(u)|^2\}. \]  

(2.10)

Assuming that the reference spectrum is the same as the input spectrum, Eq. (2.9) can be rewritten as

\[ \hat{G}_i(u) \approx |R(u)|^{-2} \times |R(u)|^2[\exp(j2ud) + \exp(-j2ud)] \]

\[ \approx \exp(j2ud) + \exp(-j2ud) \]

\[ \approx 2 \cos(2ud) \]  

(2.11)

Consequently, inverse Fourier transform of Eq. (2.11) becomes

\[ \hat{g}_i(x) \approx \delta(x + 2d) + \delta(x - 2d). \]  

(2.12)

From Eq. (2.12), it is evident that if the target signal exists in the input scene, the SFJTC produces the desired pair of delta function-like cross-correlation.

**Spectral Information Divergence**

Spectral information divergence (SID) uses a divergence measure to match pixels to reference spectra. The smaller divergence, the more likely the pixels are similar. Pixels with a measurement
greater than the specified maximum divergence threshold are not classified. Endmember spectra used by SID can come from ASCII files or spectral libraries. The SID can be expressed as \[ SID(r, s) = D(r||x) + D(x||r) \] (2.13)

where \( r \) and \( x \) are the two spectral signature vectors with \( l \) number of spectral bands in a HSI. \( D(r||x) \) is called the relative entropy of \( x \) with respect to \( r \), and \( D(x||r) \) is the relative entropy of \( r \) with respect to \( x \), which defines the average discrepancy in the self-information of \( x \) relative to the self-information of \( r \). Note that \( D(r||x) \neq D(x||r) \).

**Constrained Energy Minimization**

The goal of constrained energy minimization (CEM) is to minimize the output energy of unknown signal sources while emphasizing targets of interest by designing a finite impulse response filter specified by a weighted vector such that the filter allows the desired endmember signature \( r \) pass through using a specific constraint. The CEM, denoted as \( \delta_{CEM}(x) \), filter response for the pixel vector, given by

\[
\delta_{CEM}(x) = \frac{r^T \hat{C}^{-1} x}{r^T \hat{C}^{-1} r}
\] (2.14)

where \( \hat{C} \) is the estimated correlation matrix. CEM is more practical and general due to it only requires targets of interest without statistical modeling of background signals.

**2.2 Object Classification in HSI**

The objective of classification is to assign each pixel in HSI into a class that it belongs to, which also termed as thematic mapping. To perform the classification task, spectral libraries or training data and ground truth information are required. Two major challenges of HSI classification can be summarized into two aspects:
• The sparseness of the target class implies that insufficient availability of the training data.

• Considerable intra-class variability and interclass similarity.

To address these difficulties, many pixel-wise classification methods have been developed. Maximum likelihood or Bayesian estimation methods [8], decision trees [9, 10], neural networks (NN) [11, 12, 13], genetic algorithms [14], and kernel-based techniques [15] have been investigated for this purpose. Among these methods, support vector machine (SVM) [16] presents superior performance on HSI classification [17]. SVM aims to find an optimal hyperplane to divide the different class.

Recently, a biological inspired NN named extreme learning machine (ELM) [18] has attracted more and more attention of the community in the field of machine learning due to its higher regularization performance at a much faster speed over SVM and the other classifiers. The basic principle of ELM can be described as: when the input weight and bias are randomly allocated, the output weights are computed by the generalized inverse of the hidden layer outputs matrix. ELM on the HSI classification have been presented in [19, 20, 21, 22] and showed very promising results. In parallel, deep learning methods, which learn the representative and discriminative features in a hierarchical manner from the data, have also been explored aggressively by many fields of researchers. In following subsections, we will provide an overview of several popular NN based classification algorithms, i.e., back-propagation, ELM, and deep learning.

**Back-propagation**

Back-propagation [23] is a feed-forward multilayer perceptron (MLP) learning algorithm, which is constructed by multiple layers, namely an input layer, one or more hidden layers, and an output layer. For the case of one hidden layer, the input layer receives a signal and passes it through weighted connections to the hidden layers, the results of hidden layers are passed through another
weighted connections to the output layer $y_j (j = 1, 2, \ldots, c)$, where $c$ is the total number of output nodes.

The criterion of this network is to update the weights to produce desired outputs. There are two main passes for computing and updating the weights in the training stage: forward pass and backward pass. In the forward pass, the network computes the weighted sum for each node of hidden layer and output layer, whereas in the backward pass, it calculates the gradient of an error function with respect to all weights in the network. The objective of back-propagation algorithm is to minimize the error function in weight space using gradient descent method. The error function is computed by

$$E_j = \sum_j (t_j - y_j)^2$$

(2.15)

where $t_j$ is the target output. Since the back-propagation algorithm requires the desired output in priori, it can be categorized into a supervised learning algorithm.

**Extreme Learning Machine (ELM)**

ELM is a new type of single hidden layer feed-forward neural network (SLFN). ELM overcomes the problems of traditional feed-forward neural networks such as the presence of local minima, imprecise learning rate and slow rate of convergence. ELM has attracted more and more attention of the community in the field of machine learning due to its high regularization performance with fast speed processing speed [19, 21, 24].

ELM typically applies random computational nodes in the hidden layer and increases learning speed by means of randomly generated weights and biases for hidden nodes rather than iteratively adjusting network parameters, which is commonly adopted by gradient-based methods, such as back-propagation. The output function of ELM with $L$ hidden nodes for generalized SLFNs is
expressed as [18]

\[ y_L(x) = \sum_{i=1}^{L} \beta_i g_i(x) = \sum_{i=1}^{L} \beta_i G(w_i, b_i, x), \quad x \in \mathbb{R}^\ell, \quad \beta_i \in \mathbb{R}^c \]  

(2.16)

where \( w_i \in \mathbb{R}^\ell \) is the weight vector connecting the input nodes to the \( i \)th hidden node, \( b_i \) is the \( i \)th bias of the hidden node, \( g_i \) denotes the output function, i.e., activation function \( G(w_i, b_i, x) \) of the \( i \)th hidden node, and \( \beta_i = [\beta_1, \beta_2, \cdots, \beta_L]^T \) is the weight vector linking the \( i \)th hidden node to the output nodes. For \( N \) arbitrary distinct samples \((x_j, t_j)\), where \( x_j \in \mathbb{R}^\ell \) and \( t_j \in \mathbb{R}^c \) the SLFNs with \( L \) hidden nodes can approximate these \( N \) samples with zero error, meaning \( \sum_{j=1}^{N} \|y_j - t_j\| = 0 \).

Hence, there exists \((w_i, b_i)\) and \( \beta_i \), such that

\[ \sum_{i=1}^{L} \beta_i G(w_i, b_i, x) = t_j, \quad j = 1, 2, \cdots, N \]  

(2.17)

Equation (2.17) can be compactly written as

\[ H\beta = T \]  

(2.18)

where

\[
H = \begin{bmatrix}
h(x_1) \\
\vdots \\
h(x_N)
\end{bmatrix} = \begin{bmatrix}
G(w_1, b_1, x_1) & \cdots & G(w_L, b_L, x_1) \\
\vdots & \ddots & \vdots \\
G(w_1, b_1, x_N) & \cdots & G(w_L, b_L, x_N)
\end{bmatrix}
\]  

(2.19)

and

\[
\beta = [\beta_1, \beta_2, \cdots, \beta_c] \in \mathbb{R}^{L \times c}, \quad T = [t^T_1, t^T_2, \cdots, t^T_N]^T \in \mathbb{R}^{N \times c},
\]  

(2.20)

where \( T \) is the target matrix for training data, \( H \) is the hidden layer output matrix of the SLFN, and the \( i \)th column of \( H \) is the \( i \)th hidden node output with respect to inputs \( x_1, x_2, \cdots, x_N \), while the \( j \)th row, i.e., \( h(x_j) \), is the hidden layer feature mapping corresponding to the \( j \)th input \( x_j \). As the hidden node parameters \((w_i, b_i)\) can be randomly generated and remain unchanged, the only unknown parameters in ELM are the output weight vectors \( \beta_i \) between the hidden layer and the output layer, which can be simply resolved by ordinary least-square error analysis. Since ELM aims to minimize the training error \( \|H\beta - T\| \) and the norm of weights \( \|\beta\| \), the smallest norm
least-squares solution of the above linear system is obtained by computing output weight matrix $\beta$ by Moore-Penrose generalized inverse [25, 26, 27, 28] as

$$\hat{\beta} = H^\dagger T$$

(2.21)

where $H^\dagger$ is the Moore-Penrose generalized inverse of matrix $H$. Hence, the prediction value matrix $Y$ is expressed by

$$Y = H\hat{\beta} = HH^\dagger T$$

(2.22)

Then the error matrix can be described as

$$\varepsilon = \|Y - T\|^2 = \|HH^\dagger T - T\|^2$$

(2.23)

In order to increase the stability and generalization ability of the traditional ELM, the weight matrix $\hat{\beta}$ can be represented as

$$\hat{\beta} = (HH^T + \lambda I)^{-1}H^T T$$

(2.24)

where $\lambda$ is a constant and indicates the regularization parameter to balance the influence of the error term and the model complexity. ELM has several advantages, such as ease of use, faster learning speed, higher generalization performance, and being suitable for many nonlinear activation functions as well as kernel functions. It has also been shown that ELM yields much better generalization performance with much faster learning speed and less human interventions than other conventional methods, but there exists inconsistent behavior in its output accuracy. There are, however, many other issues that have not been solved yet, such as ensuring performance stabilization and fixing the optimum number of hidden neurons.

**Deep Learning**

Deep leaning is a deep NN (DNN) which contains more than one stage of non-linear feature transformation. Typical DNN architecture includes deep belief networks (DBN) [29], stacked auto-encoders (SAEs) [30] and convolutional neural networks (CNN) [31, 32]. Due to the composition
of many layer, DNNs are more capable for representing the highly varying nonlinear function compared to shallow learning approaches [33]. Moreover, DNNs are more efficient for learning because of the combination of feature extraction and classification layers. Most of the deep learning techniques do not require feature extraction and take raw images as inputs followed by image normalization. The low and middle levels of DNNs abstract the feature from the input image whereas the high level performs classification operation on the extracted features. The final layer of DNN uses a feed-forward neural network approach. As a result, it is structured as a uniform framework integrated with all necessary modules within a single network. Therefore, this network model often lead to better accuracy comparing with training of each module independently.

CNN is one of the most popular DNN techniques, which is inspired by biological visual cortex and tailored for computer vision tasks. In addition to the common advantages of DNNs, CNN has some extra properties: it is designed to imitate human visual processing, and it has highly optimized structures to learn the extraction and abstraction of two dimensional (2D) features. In particular, the max-pooling layer of CNN is very effective in absorbing shape variations. Moreover, composed of sparse connection with tied weights, CNN requires significantly fewer parameters than a fully connected network of similar size. Most of all, CNN is trainable with the gradient-based learning algorithm, and suffers less from the diminishing gradient problem. Given that the gradient-based algorithm trains the whole network to minimize an error criterion directly, CNN can produce highly optimized weights. Recently, deep CNN was applied for hyperspectral image classification and yielded very promising results [34].

DBN and SAE have been also applied to hyperspectral image classification tasks [35, 36]. In [35], the HSI classification framework is a fusion of PCA, SAE, and logistic regression. Specifically, as a deep learning architecture, SAE is utilized to extract useful high-level features, whereas in [36],
DBN-based deep learning architectures with spectral, spatial, and spectral-spatial features were proposed for HSI classification.

### 2.3 Spectral-Spatial Classification of Hyperspectral Data

Advances in spatial resolution enhancement of HSI provide new capability for further characterizing pixel signatures in a wide range of remote sensing applications, while attracting researchers’ interest in exploiting spatial information. Unlike the conventional HSI classification methods which only consider spectral signature of every pixel, the spatial-feature based approaches represent each pixel by extracting spatial context information of that pixel in every spectral band.

Over the last decades, a great deal of HSI classification schemes that use spatial features have been proposed in the literature. For instance, a family of composite kernels which integrates spatial and spectral information is introduced in [37]. Morphological profile (MP), utilizes morphological operation to generate spatial structural features, has been investigated for HSI classification [38]. Due to its successful performance, the improved versions of MP, such as extended MP (EMP) [39] and extended multi-attribute MP (EAMP) [40], were developed. For noise-robust HSI classification, Chen et al. [41] employed a multi-hypothesis prediction approach to incorporate spatial features to reconstruct HSI. Li et al. [42] proposed to include spatial information in HSI classification using a multilevel logistic Markov-Gibbs random field prior. Kang et al. [43] effectively utilized edge-preserving filtering as a probability optimization process to improve the classification output. Another edge computation based approach was presented in [44], where spatial and rotational autocorrelations of local image gradients are obtained by gradient local auto-correlations (GLAC) [45].

Texture information is another useful factor that can aid in HSI classification. Markov random fields (MRFs) can be used to extract texture features since they measure spatial relationship between the central pixels and its neighboring pixels, which have been successful applied in HSI
classification [46, 47]. Gabor feature as another texture descriptor have been used for HSI classification frameworks [48, 49]. In [48], two-dimensional Gabor features were generated in a principal component analysis projected subspace, while in [49] the three-dimensional Gabor filter bank was applied to hyperspectral images to capture specific orientation, scale, and wavelength-dependent properties of the data. Recently, LBP has shown very promising performance in HSI classification [50]. In this technique, the LBP code image is generated for each band in the input HSI. To describe the spatial characteristics of the pixel, the LBP histogram for each pixel of interest is computed with its corresponding neighborhood region. However, this method did not consider the texture features from the magnitude component of the image local differences, as well as the local features from multi-resolution of the image. Thus, it would be a great interest to utilize or develop advanced textural information for HSI classification.
CHAPTER III

CLASS-ASSOCIATIVE SPECTRAL FRINGE-ADJUSTED JOINT TRANSFORM CORRELATION FOR MULTICLASS OBJECT DETECTION IN HSI

We propose a deterministic object detection algorithm capable of detecting multiclass objects in HSI without any training or preprocessing. The proposed method, which is named class-associative spectral fringe-adjusted joint transform correlation (CSFJTC), is based on JTC between object and non-object spectral signatures to search for a similar match, which only requires one query (training-free) from the object’s spectral signature. Our method utilizes class-associative filtering, modified Fourier plane image subtraction, and fringe-adjusted JTC (FJTC) [7] techniques in spectral correlation domain to perform the object detection task. The output of CSFJTC yields a pair of sharp correlation peaks for a matched target and negligible or no correlation peaks for a mismatch. Experimental results, in terms of receiver operating characteristic (ROC) curves and area-under-ROC (AUROC), on two popular real-world hyperspectral datasets demonstrate the superiority of CSFJTC over the state-of-the-art hyperspectral object detection approaches.

3.1 Introduction and Motivation

The basic element of the proposed CSFJTC technique is JTC. JTC is originally developed for optical pattern recognition, where the reference image and the input scene are introduced in the input plane to create a joint image by use of a spatial light modulator (SLM). Then, an optical lens L1
performs the Fourier transform on the joint image. The intensity of the complex light distribution produced in the back focal plane of L1 called joint power spectrum (JPS), which is detected by a square-law device or a liquid-crystal light valve. In the final stage of JTC, the resultant JPS is inverse Fourier transformed by a lens L2 to yield the correlation output. If the target is present in the input scene, a pair of strong cross-correlation peaks will appear in the output plane. Since the role of L1 and L2 is to perform Fourier transform and the joint image can be achieved by the addition operation of the reference image and the input image, the optical implementation of JTC can be easily adopted in image processing.

JTC-based algorithms have shown promising results for pattern recognition in 2-D and 3-D image processing applications [7, 51, 52, 54, 55]. Among the various JTC techniques, FJTC appears to be particularly attractive because it avoids the issues otherwise associated with the alternates. To provide an efficient deterministic target detection algorithm in case of HSI, the spectral FJTC (SFJTC) has been proposed in [3]. The SFJTC determines a desired target by analyzing the correlation intensity between an unknown spectral signature and a known reference spectrum, and it is able to accommodate noise and certain variations of the spectral signatures compared to alternate deterministic detection algorithms in HSI. To improve the feasibility of the SFJTC, shifted phase-encoded SFJTC has been presented to alleviate the effects of false alarms and other artifacts [56], while the discrete wavelet-based SFJTC, as a supervised training algorithm, has been suggested to make the SFJTC more insensitive to spectral variability [57]. However, all of these techniques were designed to detect only similar patterns (single-class objects) in constant time using spectral signature correlation. Although JTC-based dissimilar pattern detection techniques from 2-D image have been introduced in the literatures [58, 59, 60], there was not attempt for multiclass target detection in HSI. Therefore, we propose a CSFJTC technique for detecting multiclass objects consisting of dissimilar patterns in HSI.
In our proposed algorithm, input spectral signatures from a given hyperspectral image data cube are correlated with multiple reference signatures via spectral signature combination and class-associative method [61]. To achieve a better correlation output, the concept of FJTC and the modified Fourier plane image subtraction (FPIS) [58] technique are incorporated in the multiple target detection processes. The output of CSFJTC provides a pair of sharp and high correlation peaks for a match and negligible or no correlation peaks for a mismatch. In other words, if there are desired multiclass patterns that are present in the scene, CSFJTC yields distinctive correlation peaks for multiclass objects simultaneously without losing inherent advantages of the SFJTC. Similar to some deterministic target detection approaches, it also does not need any a priori training step, whereas many machine learning techniques, such as support vector machines (SVMs) [16] and extreme learning machines (ELMs) [18], require the target and nontarget information before performing the target detection or classification process. Furthermore, CSFJTC employs the decision metric, such as peak-to-clutter mean (PCM) [3], to make its output hinged on the signature of the target but not the amplitude. This also enables CSFJTC robust to variations of spectral signature, since the reflectance information of a material is usually maintained in HSI, whereas the intensity may change due to background noise.

In a nutshell, the proposed method operates using a single pixel spectrum of a similar or dissimilar class of objects to find matches, does not require prior knowledge (learning) about objects or background, and does not require any preprocessing step of a target spectrum.

The main innovative contributions of this work can be summarized as follows.

- A CSFJTC technique is developed for object detection in HSI.
- A deterministic training-free multiclass object detection algorithm for hyperspectral data is proposed.
We also conduct a comparative study and investigate the relationship between SFJTC and CSFJTC techniques in hyperspectral target detection. For single class object detection task, it is shown that they are closely related and essentially equivalent, provided that the reference spectral signatures in SFJTC and CSFJTC are the same and input hyperspectral data have negligible noise or large signal-to-noise-ratio (SNR).

The rest of this chapter is organized as follows: Section 3.2 provides the formulation of the proposed method. In Section 3.3, test results are presented and discussed. Finally, Section 3.4 outlines concluding remarks of this technology.

3.2 Theoretical Analysis

A block diagram of the CSFJTC algorithm for correlating the input spectral signature with two reference spectra is shown in Fig. 3.1.

Figure 3.1: Block diagram of the proposed pattern recognition scheme for the case of two reference spectra. JPS: joint power spectrum; CJS: combination of JPS; CSGFAF: class-associative spectral generalized fringe-adjusted filter.
To simply illustrate the CSFJTC concept, let us consider that two known targets from two classes have unique spectral signatures, \( r_1(x) \) and \( r_2(x) \), respectively. Then, two joint signatures of \( r_1(x) \) and unknown input spectrum \( x_i(x) \) in the \( x \)-axis direction can be obtained as follows:

\[
\begin{align*}
J_{11i} &= r_1(x + d) + x_i(x - d). \\
J_{21i} &= r_1(x + d) - x_i(x - d).
\end{align*}
\]  

Similarly, using \( r_2(x) \) and \( x_i(x) \), another two joint spectral signatures are generated by

\[
\begin{align*}
J_{12i} &= r_2(x + d) + x_i(x - d). \\
J_{22i} &= r_2(x + d) - x_i(x - d).
\end{align*}
\]

Applying the Fourier Transform to Eqs. (3.1) and (3.2) with respect to reference spectrum \( r_1(x) \), we get

\[
\begin{align*}
F_{11i}(u) &= |R_1(u)| \exp[j\phi_{r_1}(u) + jud] + |X_i(u)| \exp[j\phi_{x_i}(u) - jud] \\
F_{21i}(u) &= |R_1(u)| \exp[j\phi_{r_1}(u) + jud] - |X_i(u)| \exp[j\phi_{x_i}(u) - jud]
\end{align*}
\]

where \( F_{11i}(u) \) and \( F_{21i}(u) \) are the Fourier transform of \( J_{11i} \) and \( J_{21i} \), respectively. Similarly, applying the Fourier transform to Eqs. (3.3) and (3.4), we obtain

\[
\begin{align*}
F_{12i}(u) &= |R_2(u)| \exp[j\phi_{r_2}(u) + jud] + |X_i(u)| \exp[j\phi_{x_i}(u) - jud] \\
F_{22i}(u) &= |R_2(u)| \exp[j\phi_{r_2}(u) + jud] - |X_i(u)| \exp[j\phi_{x_i}(u) - jud]
\end{align*}
\]

where \( F_{12i}(u) \) and \( F_{22i}(u) \) are the Fourier transform of \( J_{12i} \) and \( J_{22i} \), respectively. The corresponding JPS to Eqs. (3.5) and (3.6) can be computed separately as follows:

\[
O_{11i}(u) = |F_{11i}(u)|^2 = |R_1(u)|^2 + |X_i(u)|^2 + 2|R_1(u)||X_i(u)| \cos[\phi_{r_1}(u) - \phi_{x_i}(u) + jud]
\]

(3.9)
\[O_{21i}(u) = |F_{21i}(u)|^2 = |R_1(u)|^2 + |X_i(u)|^2 - 2|R_1(u)||X_i(u)| \cos[\phi_{r1}(u) - \phi_{x_i}(u) + jud]. \tag{3.10}\]

Likewise, the JPS for \(F_{12i}(u)\) and \(F_{22i}(u)\) can be expressed respectively as

\[O_{12i}(u) = |F_{12i}(u)|^2 = |R_2(u)|^2 + |X_i(u)|^2 + 2|R_2(u)||X_i(u)| \cos[\phi_{r2}(u) - \phi_{x_i}(u) + jud] \tag{3.11}\]

\[O_{22i}(u) = |F_{22i}(u)|^2 = |R_2(u)|^2 + |X_i(u)|^2 - 2|R_2(u)||X_i(u)| \cos[\phi_{r2}(u) - \phi_{x_i}(u) + jud] \tag{3.12}\]

On the right-hand sides of Eqs. (3.9) – (3.12), the first two terms correspond to the zero-order terms, whereas the third term is the desired cross-correlation between the reference spectrum and the input spectrum. The zero-order diffractions are due to autocorrelation of the reference spectrum and the input spectrum, which are responsible for producing false alarms in the JPS. To suppress the zero-order term and reduce the effects of input-scene noise, the combination of the JPS is computed as

\[|P_{1i}(u)|^2 = O_{11i}(u) - O_{21i}(u) = 4|R_1(u)||X_i(u)| \cos[\phi_{r1}(u) - \phi_{x_i}(u) + jud]. \tag{3.13}\]

\[|P_{2i}(u)|^2 = O_{12i}(u) - O_{22i}(u) = 4|R_2(u)||X_i(u)| \cos[\phi_{r2}(u) - \phi_{x_i}(u) + jud]. \tag{3.14}\]

From Eqs. (3.13) and (3.14), it is evident that the undesired zero-order terms are eliminated simply by subtraction operation using the JPS of two joint spectral signatures for each reference class, which can be considered as an alternative approach compared to FPIS; we may term this approach as a modified FPIS (MFPI). In contrast, SFJTC employs the FPIS technique to eliminate the zero-order terms, where the power spectra of the input signature and the reference signature are subtracted from the JPS. However, the FPIS approach requires the power spectra of the input signature and the reference signature, which introduces complexity for both real-world optical
implementation. Therefore, we generate two joint spectral signatures for each reference class to remove the undesired autocorrelation terms and avoid implementation complexity.

To detect multiple-class objects simultaneously, the JPS in Eqs. (3.13) and (3.14) are further combined to achieve equal correlation peaks for both targets, and their energy contents are controlled by coefficients $a_1$ and $a_2$, such that $a_1 + a_2 = 1$, as follows:

$$|P_i(u)|^2 = a_1|P_{1i}(u)|^2 + a_2|P_{2i}(u)|^2$$

$$= 4a_1|R_1(u)||X_i(u)|\cos[\phi_{r_1}(u) - \phi_{x_i}(u) + j\alpha]$$

$$+ 4a_2|R_2(u)||X_i(u)|\cos[\phi_{r_2}(u) - \phi_{x_i}(u) + j\alpha]$$

(3.15)

If there are $c$ classes in the input hyperspectral image, Eq. (3.15) can be rewritten as

$$|P_i(u)|^2 = \sum_{j}^c |P_{ji}(u)|^2$$

$$= 4a_1|R_1(u)||X_i(u)|\cos[\phi_{r_1}(u) - \phi_{x_i}(u) + j\alpha]$$

$$+ 4a_2|R_2(u)||X_i(u)|\cos[\phi_{r_2}(u) - \phi_{x_i}(u) + j\alpha]$$

$$+ \cdots + 4a_c|R_c(u)||X_i(u)|\cos[\phi_{r_c}(u) - \phi_{x_i}(u) + j\alpha]$$

(3.16)

where $a_1, a_2, \cdots, a_c$ are nonzero coefficients constrained to $\sum_{j}^c a_j = 1$. Note that $a_1$ and $a_2$ in Eq. (3.15) correspond to the case of two classes. The values of $a_j (j = 1, 2, \cdots, c)$ may be varied depending on the energy content of the JPS. It is observed that the detection performance can be improved by fine tuning these parameters.

In FJTC, the FAF is effectively utilized to suppress the noise spectrum at the Fourier plane. Thus, to produce sharper and larger correlation peak intensity (CPI), the FAF filter is employed and reformulated to a class-associative spectral generalized FAF (CSGFAF), which is defined as

$$\tilde{H}(u) = \frac{1}{\epsilon + |R_1(u)|^m + |R_2(u)|^m}$$

(3.17)

where $\epsilon$ is a constant that is used to avoid the presence of zero poles that may force the gain of the CSGFAF to approach infinity, thereby creating a serious design problem in realizing this filter. The
parameter $m$ is a constant that can be either 0, 1, or 2. Depending on the value of $m$, the CSGFAF in Eq. (3.17) corresponds to the classical matched filter ($m = 0$), the phase-only filter ($m = 1$), or the FAF ($m = 2$). Thus, all important types of matched-filter-based correlators can be implemented using the proposed CSGFAF, while avoiding the limitations of matched-filter-based correlators. The study in [60] has shown that, for an input signal with noise, the phase-only filter ($m = 1$) yields better correlation outputs than the FAF ($m = 2$). Accordingly, the JPS in Eq. (3.15) can be further enhanced by multiplying the CSGFAF and yields

$$\tilde{G}_i(u) = \left| P_i(u) \right|^2 \times \tilde{H}(u) = \frac{a_1 \left| P_{i1}(u) \right|^2 + a_2 \left| P_{i2}(u) \right|^2}{\epsilon + |R_1(u)|^m + |R_2(u)|^m}. \quad (3.18)$$

It is worth mentioning that, for single-class object detection, only one reference spectral signature will be used in CSFJTC. In this case, Eq. (3.18) will be expressed as

$$\tilde{G}_i(u) = |P_i(u)|^2 \times \tilde{H}(u) = \frac{4a_1 |R_1(u)| \left| X_i(u) \right| \cos[\phi_{r_1}(u) - \phi_{x_i}(u) + j\delta u]}{\epsilon + |R_1(u)|^m}. \quad (3.19)$$

If input and reference spectral signatures are the same and $|R_1(u)|^m \gg \epsilon$ for $m = 2$, then Eq. (3.19) becomes

$$\tilde{G}_i(u) \approx 4a_1 \cos(2\delta u). \quad (3.20)$$

Finally, applying an inverse Fourier transform to $\tilde{G}_i(u)$ produces the final correlation output.

For the case of two reference spectral signatures from different classes, i.e., $r_1 \neq r_2$, if the input pixel spectral signature is the same as $r_1$, the parameter $m = 2$, and for $|R_1(u)|^2, |R_2(u)|^2 \gg \epsilon$, $\tilde{G}_i(u)$ in Eq. (3.18) becomes

$$\tilde{G}_i(u) \approx |P_i(u)|^2 \times \tilde{H}(u) = \frac{4a_1 \cos(2\delta u) + a_2 \kappa_1 \cos[\phi_{r_2}(u) - \phi_{x_i}(u) + 2\delta u]}{1 + \kappa_1^2}. \quad (3.21)$$

where $\kappa_1 = \frac{|R_2(u)|}{|R_1(u)|}$. For $\kappa_1 \ll 1$, $\tilde{G}_i(u)$ in Eq. (3.21) can be approximated as

$$\tilde{G}_i(u) \approx 4a_1 \cos(2\delta u). \quad (3.22)$$
Similarly, if the input spectrum is the same as \( r_2 \), \( \tilde{G}_i(u) \) in Eq. (3.18) becomes

\[
\tilde{G}_i(u) \approx |P_i(u)|^2 \times \mathcal{H}(u) = 4 \frac{a_2 \cos(2ud) + a_1 \kappa_2 \cos[\phi_r(u) - \phi_{x_i}(u) + 2ud]}{1 + \kappa_2^2}. \tag{3.23}
\]

where \( \kappa_2 = \frac{|R_1(u)|}{|R_2(u)|} \). for \( \kappa_2 \ll 1 \), \( \tilde{G}_i(u) \) in Eq. (3.23) can be approximated as

\[
\tilde{G}_i(u) \approx 4a_2 \cos(2ud). \tag{3.24}
\]

For \( \kappa_1 = 1 \), i.e., \( r_1 = r_2 \), using the same parameters as in Eq. (3.21), \( \tilde{G}_i(u) \) can be estimated as

\[
\tilde{G}_i(u) \approx 2[a_1 \cos(2ud) + a_2 \cos(2ud)] \approx 2 \cos(2ud). \tag{3.25}
\]

Accordingly, the same result can be obtained for Eq. (3.23) when \( \kappa_2 = 1 \). From Eqs. (3.22), (3.24) and (3.25), it is observed that an inverse Fourier transform of \( \tilde{G}_i(u) \) will generate a pair of delta-function-like cross-correlation peaks indicating the presence of a target. In addition, it can be observed that SFJTC and CSFJTC yield similar outputs when input targets are the same as the reference spectral signature. However, here, we assume that SNR is large enough or noise can be negligible.

A study in [3] has shown that the highest value of the correlation output does not yield a reliable decision for target discrimination. This is because the CPI from false signals may have very close value with that from true signals in the correlation plane. Therefore, to avoid false target detection, we utilize PCM instead of CPI to identify the targets. PCM corresponding to the \( i \)th pixel spectrum is defined as

\[
\text{PCM} = \frac{\text{CPI}}{L-1} \sum_{\bar{k}=1}^{\bar{L}} \sum_{\bar{k} \neq \text{CPI}} \tilde{g}_i(x) \tag{3.26}
\]

where \( \tilde{g}_i(x) \) is the inverse Fourier transform of \( \tilde{G}_i(u) \) in Eq. (3.25). \( L \) represents the half-length of the correlation output vector, and \( 1 \leq \bar{k} \leq \bar{L} \). If the target presents in the input image, the final correlation output in the half correlation plane will produce the desired delta-function-like correlation peak at the location of the target in the scene.
The performance of the proposed technique can also be quantified in terms of the true positive rate (TPR) and false positive rate (FPR), which are defined as

\[
TPR = \frac{\sum t_p}{\sum t_p + \sum f_n}
\]  

\[
FPR = \frac{\sum f_p}{\sum f_p + \sum t_n}
\]

where \(t_p\) is the target pixel detected correctly (true positive), \(f_p\) is the false alarm target pixel (false positive), \(t_n\) is the background pixel detected correctly (true negative), and \(f_n\) is the false background pixel (false negative).

Figure 3.2: Reference and input spectral signatures. (a) Reference 1 and 2 spectral signatures. (b) Reference 1 and reference 2 with a false input.

To illustrate a correlation output of CSFJTC based on the analysis earlier, we correlate two known reference spectral signatures with three input spectral signatures, as shown in Fig. 3.2(a) and (b), respectively. Fig. 3.2(a) shows two known reference signals, whereas Fig. 3.2(b) shows that the unknown input spectral signal is dissimilar with either the reference signal 1 or the reference signal 2, which may be considered as a nontarget object or false input. The corresponding correlation
outputs of the CSFJTC are shown in Fig. 3.3, where the y-axis indicates the correlation intensity, i.e., CPI. In this figure, it is evident that CSFJTC yields a pair of distinct peaks for a matched signal and negligible ones for a mismatch. Table 3.1 shows quantitative results that are consistent with Fig. 3.3, in terms of CPI and PCM metrics. From this table, it can be seen that CSFJTC produces much higher CPI and PCM, when an input is a true target spectral signature rather than a false signal. It can also be observed that PCM provides better discriminability than CPI. Note that the values of CPI and PCM for the two true inputs are not equal. This is because the energy contents of the power spectra of the two reference signals are different. However, we can obtain the desired correlation output by simply adjusting parameters $a_1$ and $a_2$.

Figure 3.3: Correlation outputs with spectral signatures. (a) Correlation output of CSFJTC for an input as reference 1. (b) Correlation output of CSFJTC for an input as reference 2. (c) Correlation output of CSFJTC for a false input. For parameters in CSFJTC, $a_1 = a_2 = 0.5$ and $m = 2$.

3.3 Evaluation

Here, we validate the proposed CSFJTC technique on two commonly used hyperspectral datasets from two aspects. We first evaluate the performance of CSFJTC in single-class object detection in HSI, as well as comparisons to other eight competitive hyperspectral target detection algorithms, which do not require training. They are adaptive matched filter (AMF) [62], spectral JTC (SJTC)
Table 3.1: Quantitative analysis of the CSFJTC outputs corresponding to Figs. 3.2 and 3.3.

<table>
<thead>
<tr>
<th>Spectral signatures</th>
<th>Evaluation metric</th>
<th>CSFJTC results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference 1 as an input</td>
<td>CPI</td>
<td>0.59</td>
</tr>
<tr>
<td></td>
<td>PCM</td>
<td>32.48</td>
</tr>
<tr>
<td>Reference 2 as an input</td>
<td>CPI</td>
<td>0.54</td>
</tr>
<tr>
<td></td>
<td>PCM</td>
<td>23.63</td>
</tr>
<tr>
<td>False target input</td>
<td>CPI</td>
<td>0.14</td>
</tr>
<tr>
<td></td>
<td>PCM</td>
<td>3.93</td>
</tr>
</tbody>
</table>

[3], SFJTC, SAM [2], CEM, adaptive cosine/coherent estimator (ACE) [63], spectral information divergence (SID) [64], and Euclidian minimum distance (EMD) [65]. Then, we conduct experiments on multiclass object detection tasks with one query in each class of objects as a reference spectral signature for CSFJTC and compare the results with different FPIS and parameter $m$.

### 3.3.1 Dataset Description and Test Setups

Two widely used standard hyperspectral datasets, as described in the following, are used for testing and evaluation.

**Indian Pines**: The Indian Pines hyperspectral data set was acquired by the Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) over northwestern Indiana. The image has 220 spectral bands of size $145 \times 145$ pixels within a spectral range from 0.4 to 2.5 $\mu m$. There are 16 mutually exclusive classes considered in this test site. Prior to experiments, 20 water absorption bands are removed, resulting in 200 spectral bands remained. Fig. 3.4(a) shows the false-color composite image, and the corresponding ground truth is depicted in Fig. 3.4(b).

**Salinas**: This scene was gathered by the 224-band AVIRIS over Salinas Valley, CA. The data set has a size of $512 \times 217$ pixels, with a spatial resolution of 3.7 $m$ per pixel. Twenty water absorption bands ($108–112, 154–167, and 224$) were discarded before experiments. The scene is covered by
vegetables, bare soils, and vineyard fields, which consists of sixteen classes. Figs. 3.5(a) and (b) shows the corresponding false-color composite image and ground truth data, respectively.

Figure 3.4: Indian Pines data set. (a) False-color composite. (b) Ground truth.

Figure 3.5: Salinas Scene data set. (a) False-color composite. (b) Ground truth.
3.3.2 Experimentation on Single-Class Object Detection

In Section 3.2, we have theoretically shown that the proposed CSFJTC algorithm can perform single-class object detection tasks by choosing one reference spectral signature for each class at a time. Similar to SFJTC, CSFJTC is also insensitive to the intensity changes of the reference signature, since it records the ratio of the highest peak to the clutter mean for the pixel under analysis. We also concluded that, under certain circumstances, SFJTC and CSFJTC will yield approximately the same results. To examine these remarks, we apply CSFJTC and SFJTC, as well as SJTC, AMF, SAM, CEM, ACE, SID, and EMD, on the three aforementioned hyperspectral datasets. The detection results are then compared both visually and quantitatively using ROC curves and the corresponding AUROC. The ROC curve describes the TPR as a function of the FPR.

Experimental Setup: In all experiments, we first normalize the input image between 0 and 1 to control the magnitude of the computation within the range. Next, we apply the maximum noise fraction (MNF) [66] transform, which is also referred to as noise-adjusted principal component (PC) [67] transform, on the original image as a feature extraction. Compared to PC analysis (PCA) [68], MNF arranges PCs in decreasing order of SNR rather than variance. By retaining the sufficient lower bands in the MNF-transformed image, we are able to reduce the data dimension and obtain large-SNR band images. As mentioned earlier, larger SNR makes the outputs of CSFJTC and SFJTC the same or similar, and the MNF transform provides such a stage for this condition. In this experiment, we empirically choose the first 50 bands of the MNF-transformed image. To convincingly compare and estimate the capabilities of the proposed method with other approaches, we run the experiments ten times with randomly selected reference spectrum for each trial, and ROC and AUROC are averaged over these ten repeated trials.

Parameter Setting: For the parameters of CGFAF in CSFJTC, we select standard values $\epsilon = 10^{-3}$ and $m = 2$. To demonstrate the similarity of SFJTC and CSFJTC for single-class object
detection, we use one reference spectral signature in CSFJTC for each class object detection, and the JPS coefficient $a_j$ (where $j = 1$ in single-class object detection) is set to 1. As for the FAF in SFJTC, we also set $A(u) = 10^{-3}$ and $B(u) = 1$ to have a fair comparison with CSFJTC.

Results and Comparisons: Here, since we deal with single-class object detection, ROC curves can be generated for each class to visually assess different detection methods. However, due to large amount of classes in the datasets, it would be inefficient to present all ROC curves here. Thus, we suggest producing a mean ROC (MROC) curve for each data set, by averaging the ROC of each class. Figs. 3.6 and 3.7 show the corresponding MROC curves obtained by different methods that tested on the Indian Pines, Salinas, and University of Pavia datasets, respectively. Qualitatively, the closer to the upper left corner of the plot in the ROC curves, the better the performance. Consequently, a bigger value of AUROC indicates a better outcome.

From test results on the Indian Pines dataset, as shown in Fig. 3.6 and Table 3.2, it is clear that CSFJTC and SFJTC have superior performance than the other approaches. It is worth noting that CSFJTC and SFJTC have identical results; hence, their ROC curves in Fig. 3.6 are overlapped, which makes the plot for SFJTC not visible in this figure and also the other similar figures. Table 3.2 reports the corresponding AUROC of various detection algorithms for each class in the Indian Pines data set. The bold numbers in the section denote the greatest performance among all detectors. From this section, it can be observed that both SFJTC and CSFJTC provide the highest AUROC in 9 out of 16 classes and thus yield the best results when considering the average AUROC. Comparing AMF with CEM, it is not surprising to discover that they have very similar outputs. This is due to the fact that the AMF is essentially a mean-centered version of CEM, except that AMF uses the covariance matrix instead of the correlation matrix used in CEM.
Figure 3.7 depicts MROC curves for the Salinas dataset, and Table 3.3 exhibits the corresponding AUROC for the 16 types of land-cover classes. Similar to the results obtained from the experiments on the Indian Pines image, CSFJTC and SFJTC outperform the other methods, in terms of AUROC. Moreover, from ROC curves provided in Fig. 3.7, we observe that, for any given FPR, CSFJTC and SFJTC produce higher TPR than the other techniques, indicating their effectiveness.

Performances of the aforementioned detectors can be further analyzed, in terms of their mathematical formulation, such as deterministic- or statistic-based measure. Deterministic target detection techniques, such as SAM, calculate the spectral similarity value between an input spectral signature and target spectral signatures using only spectral vectors, which is effective only if the spectral signature vectors to be compared are pure signatures of the materials. Accordingly, if a target signature
Figure 3.7: MROC for the Salinas dataset.

vector is either mixed by other substances, such as background signatures (nontarget signatures), or embedded in a single signature as a subpixel target, the deterministic approach may fail in target determination and produce false alarms. These obstacles can be partially resolved by statistical approaches, such as ACE, which is capable of detecting subpixel targets. This echoes with results in Tables 3.2 and 3.3, where statistical approaches yield better detection results than deterministic approaches due to mixed spectral signatures that may occur in the input datasets. However, many statistical models, such as AMF and ACE, assume the background as a multivariate normal distribution, which requires that covariance matrices should be calculated over a normally distributed data region, which is not always true in real-world HSI data. Although CEM as a subpixel detector is not based on assumption of background statistics, it is found to be very sensitive to noise [69].
Table 3.2: AUROC comparison for the Indian Pines dataset. The best accuracy for each class and the best overall performance is highlighted in boldface.

<table>
<thead>
<tr>
<th>Class</th>
<th>Samples</th>
<th>SAM</th>
<th>ACE</th>
<th>CEM</th>
<th>AMF</th>
<th>SID</th>
<th>EMD</th>
<th>SFJC</th>
<th>SFJC</th>
<th>CSFJC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alfalfa</td>
<td>46</td>
<td>0.401</td>
<td>0.868</td>
<td>0.923</td>
<td>0.929</td>
<td>0.891</td>
<td>0.326</td>
<td>0.578</td>
<td>0.936</td>
<td>0.936</td>
</tr>
<tr>
<td>Corn-notill</td>
<td>1428</td>
<td>0.431</td>
<td>0.539</td>
<td>0.598</td>
<td>0.602</td>
<td>0.829</td>
<td>0.601</td>
<td>0.889</td>
<td>0.839</td>
<td>0.839</td>
</tr>
<tr>
<td>Corn-mintill</td>
<td>830</td>
<td>0.567</td>
<td>0.625</td>
<td>0.685</td>
<td>0.685</td>
<td>0.858</td>
<td>0.879</td>
<td>0.768</td>
<td>0.866</td>
<td>0.866</td>
</tr>
<tr>
<td>Corn</td>
<td>237</td>
<td>0.651</td>
<td>0.536</td>
<td>0.614</td>
<td>0.616</td>
<td>0.764</td>
<td>0.926</td>
<td>0.666</td>
<td>0.833</td>
<td>0.833</td>
</tr>
<tr>
<td>Grass-pasture</td>
<td>483</td>
<td>0.626</td>
<td>0.703</td>
<td>0.710</td>
<td>0.713</td>
<td>0.665</td>
<td>0.791</td>
<td>0.265</td>
<td>0.716</td>
<td>0.716</td>
</tr>
<tr>
<td>Grass-trees</td>
<td>730</td>
<td>0.591</td>
<td>0.720</td>
<td>0.758</td>
<td>0.767</td>
<td>0.756</td>
<td>0.614</td>
<td>0.392</td>
<td>0.790</td>
<td>0.790</td>
</tr>
<tr>
<td>Grass-pasture-mowed</td>
<td>28</td>
<td>0.416</td>
<td>0.931</td>
<td>0.969</td>
<td>0.967</td>
<td>0.932</td>
<td>0.248</td>
<td>0.598</td>
<td>0.971</td>
<td>0.971</td>
</tr>
<tr>
<td>Hay-windrowed</td>
<td>478</td>
<td>0.602</td>
<td>0.767</td>
<td>0.877</td>
<td>0.875</td>
<td>0.921</td>
<td>0.099</td>
<td>0.608</td>
<td>0.958</td>
<td>0.958</td>
</tr>
<tr>
<td>Oats</td>
<td>20</td>
<td>0.597</td>
<td>0.812</td>
<td>0.888</td>
<td>0.887</td>
<td>0.942</td>
<td>0.849</td>
<td>0.438</td>
<td>0.955</td>
<td>0.955</td>
</tr>
<tr>
<td>Soybean-notill</td>
<td>972</td>
<td>0.362</td>
<td>0.562</td>
<td>0.624</td>
<td>0.622</td>
<td>0.828</td>
<td>0.473</td>
<td>0.847</td>
<td>0.835</td>
<td>0.835</td>
</tr>
<tr>
<td>Soybean-mintill</td>
<td>2455</td>
<td>0.491</td>
<td>0.565</td>
<td>0.636</td>
<td>0.635</td>
<td>0.833</td>
<td>0.592</td>
<td>0.887</td>
<td>0.863</td>
<td>0.863</td>
</tr>
<tr>
<td>Soybean-clean</td>
<td>593</td>
<td>0.445</td>
<td>0.540</td>
<td>0.638</td>
<td>0.638</td>
<td>0.775</td>
<td>0.722</td>
<td>0.739</td>
<td>0.781</td>
<td>0.781</td>
</tr>
<tr>
<td>Wheat</td>
<td>205</td>
<td>0.490</td>
<td>0.893</td>
<td>0.908</td>
<td>0.931</td>
<td>0.918</td>
<td>0.772</td>
<td>0.347</td>
<td>0.942</td>
<td>0.942</td>
</tr>
<tr>
<td>Woods</td>
<td>1265</td>
<td>0.487</td>
<td>0.613</td>
<td>0.662</td>
<td>0.665</td>
<td>0.868</td>
<td>0.251</td>
<td>0.077</td>
<td>0.910</td>
<td>0.910</td>
</tr>
<tr>
<td>Buildings-Grass-Trees-Dives</td>
<td>386</td>
<td>0.600</td>
<td>0.560</td>
<td>0.624</td>
<td>0.622</td>
<td>0.703</td>
<td>0.518</td>
<td>0.338</td>
<td>0.742</td>
<td>0.742</td>
</tr>
<tr>
<td>Stone-Steel-Towers</td>
<td>93</td>
<td>0.515</td>
<td>0.846</td>
<td>0.929</td>
<td>0.938</td>
<td>0.842</td>
<td>0.680</td>
<td>0.737</td>
<td>0.866</td>
<td>0.866</td>
</tr>
<tr>
<td><strong>Average AUROC</strong></td>
<td></td>
<td>0.517</td>
<td>0.693</td>
<td>0.753</td>
<td>0.756</td>
<td>0.833</td>
<td>0.584</td>
<td>0.573</td>
<td>0.863</td>
<td>0.863</td>
</tr>
</tbody>
</table>

Furthermore, it is worth noting that, although the proposed CSFJTC is a deterministic target detection algorithm, it even outperforms the aforementioned statistic-based methods. This is because CSFJTC is not a simple step process to measure the similarity of spectral signatures, but it involves multiple stages to aid target determination, such as JPS subtraction and fringe-adjusted filtering, which play significant roles in accommodating corrupted spectral signatures.

**Experiments on difference HSI sensor for single-class objec detection**

Last but not the least, we further investigate our CSFJTC technique on a different HSI sensor to check its robustness. The HSI sensor we used here is the Resonon Pika II hyperspectral camera, which provides 240 spectral channels that ranges from 400-900nm with 2.1nm spectral resolution.
Table 3.3: AUROC comparison for the Salinas dataset.

<table>
<thead>
<tr>
<th>Class</th>
<th>Samples</th>
<th>Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Broccoli green weeds 1</td>
<td>2009</td>
<td>SAM 0.501 ACE 0.985 CEM 0.994 AMF 0.994 SID 0.993 EMD 0.820 SJTC 0.998 SFJT 0.995 CSFJT 0.993</td>
</tr>
<tr>
<td>Broccoli green weeds 2</td>
<td>3726</td>
<td></td>
</tr>
<tr>
<td>Fallow</td>
<td>1976</td>
<td>0.741 0.867 0.786 0.785 0.829 0.174 0.651 0.870 0.991 0.991</td>
</tr>
<tr>
<td>Fallow rough plow</td>
<td>1394</td>
<td></td>
</tr>
<tr>
<td>Fallow smooth</td>
<td>2678</td>
<td>0.303 0.745 0.834 0.842 0.951 0.483 0.660 0.989 0.989</td>
</tr>
<tr>
<td>Stubble</td>
<td>3959</td>
<td>0.599 0.948 0.980 0.981 0.996 0.470 0.630 0.997 0.997</td>
</tr>
<tr>
<td>Celery</td>
<td>3579</td>
<td>0.366 0.971 0.981 0.986 0.970 0.229 0.969 0.983 0.983</td>
</tr>
<tr>
<td>Grapes_untrained</td>
<td>11271</td>
<td>0.762 0.605 0.726 0.723 0.901 0.742 0.932 0.938 0.938</td>
</tr>
<tr>
<td>Soil_vineyard develop</td>
<td>6203</td>
<td>0.311 0.602 0.717 0.717 0.890 0.746 0.823 0.953 0.953</td>
</tr>
<tr>
<td>Corn_senesced green weeds</td>
<td>3278</td>
<td>0.571 0.679 0.770 0.777 0.740 0.728 0.508 0.761 0.761</td>
</tr>
<tr>
<td>Lettuce_romaine_4wk</td>
<td>1068</td>
<td>0.764 0.966 0.981 0.981 0.929 0.772 0.541 0.938 0.938</td>
</tr>
<tr>
<td>Lettuce_romaine_5wk</td>
<td>1927</td>
<td>0.533 0.576 0.671 0.670 0.874 0.755 0.578 0.870 0.870</td>
</tr>
<tr>
<td>Lettuce_romaine_6wk</td>
<td>916</td>
<td>0.698 0.915 0.951 0.954 0.989 0.730 0.993 0.995 0.995</td>
</tr>
<tr>
<td>Lettuce_romaine_7wk</td>
<td>1070</td>
<td>0.676 0.942 0.969 0.969 0.890 0.735 0.963 0.946 0.946</td>
</tr>
<tr>
<td>VInyards_untrained</td>
<td>7268</td>
<td>0.851 0.579 0.689 0.690 0.917 0.850 0.904 0.938 0.938</td>
</tr>
<tr>
<td>VInyards_vertical_trellis</td>
<td>1807</td>
<td>0.577 0.915 0.947 0.950 0.895 0.897 0.884 0.906 0.906</td>
</tr>
<tr>
<td>Average AUROC</td>
<td></td>
<td>0.582 0.803 0.865 0.868 0.918 0.623 0.781 0.941 0.941</td>
</tr>
</tbody>
</table>

Figure 3.8: Detection in single-class case: (a) Input scene, and (b) locations of the inserted targets.

To evaluate the performance for multiple single-class target detection, we randomly selected ten pixels from the input hyperspectral dataset and replaced their signatures with ten target signatures.
obtained from another HSI. Figure 3.8(a) shows true color bands of the input hyperspectral data cube and Figure 3.8(b) indicates location of the inserted targets in the scene. The corresponding correlation output is shown in Fig. 3.9. From this result, it is evident that the proposed technique successfully detected all of the targets and rejected non-target objects.

3.3.3 Experimentation on Multiclass Object Detection

There are three major differences between single-class and multiclass object detection in the proposed CSFJTC method:

- Only one reference spectral signature is used in single-class objection detection, while \( c (c \geq 2) \) dissimilar reference spectral signatures (one reference for each class) are used in multiclass
object detection, which allows for detecting multiple objects from different classes simultaneously.

- One JPS parameter $a_1$ is used in single-class objection detection, whereas $c (c \geq 2)$ such parameters, as shown in Eq. (3.16), are used in multiclass object detection, where each parameter $a_j (j = 1, 2, \cdots, c)$ corresponds to each reference-spectral-signature-related JPS.

- Formulation of the CSGFAF for single-class and multiclass object detection is slightly different. For the former, only two terms (i.e., $\epsilon$ and $|R_1(u)|^m$) appear in the denominator of CSGFAF, while $c + 1$ terms (i.e., $\epsilon, |R_j(u)|^m, j = 1, 2, \cdots, c$) are produced in the latter.

**Preprocessing:** Similar to single-class object detection experiments, the MNF is also first applied to the raw images before subsequent processes. For the purpose of multiclass object detection, we randomly select one MNF-transformed target spectral signature from each class to represent reference spectral signatures, which is a matrix with a size: (number of target classes) (dimension of MNF-transformed pixel vector). For example, if we use 50 MNF bands for the Indian Pines dataset, the reference spectral signatures will be a matrix with size of $16 \times 50$. During the experiments, we varies the MNF bands from 50 to 200 for both CSFJT ($m = 1$) and CSFJT ($m = 2$) and the best accuracy is reported.

<table>
<thead>
<tr>
<th>Table 3.4: AUROC comparison for multiclass object detection in HSI.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Indian Pines Dataset</strong></td>
</tr>
<tr>
<td>-------------------------</td>
</tr>
<tr>
<td>CSFJT ($m=1$)</td>
</tr>
<tr>
<td>0.697</td>
</tr>
</tbody>
</table>
Parameter Setting: For the JPS coefficients $a_j (j = 1, 2, \cdots, c)$, we select $a_1 = a_2 = \cdots = a_c = 1/c$ to have equal weights for each class of JPS, where $c$ is 16, for both Indian Pines and Salinas datasets. It is worth mentioning that varying the values of these coefficients may produce slightly different results based on the input data set. The way to obtain the optimal coefficient will be investigated in our future research direction. As for the parameters $m$ in CSGFAF, we choose $m = 1$ and $m = 2$ to examine their performance.

Effect of Parameter $m$: The influence of parameter $m$ on the detection results for CSFJTC is summarized in Table 3.4. From these results, it can be seen that CSFJTC ($m = 1$) provides the better accuracy for both datasets than CSFJTC ($m = 2$). This is due to the fact that there is always an additive amount of noise from various sources (sensor, atmosphere, etc.) in the input hyperspectral image and higher MNF bands contain more noise information, and this will cause the JPS to contain noise terms in forms of autocorrelation of noise itself, cross-correlation between the noise and the reference, and cross-correlation between the noise and the input pixel spectral signature. If we choose $m = 2$ in CSFJTC for the case of $c$ classes, the aforementioned noise terms will be divided by a factor of $\sum_{j=1}^{c} |R_j(u)|^2$, while the factor will be $\sum_{j=1}^{c} |R_j(u)|$ for $m = 1$. When values of $|R_j(u)| \ll 1$, the effect of noise will be less dominant if $m = 1$. Furthermore, it is observed that varying MNF bands or raw number of spectral bands yields different results. Thus, a wise choice of parameter $m$ is important to achieve greater detection performance.

Experiments on difference HSI sensor for multiclass object detection

Finally, similar to single-class detection, we further investigate our CSFJTC technique on a different HSI sensor to check its robustness. In this experiment, we randomly chose six pixels from the input scene in Fig. 3.8 (a) and replaced their signatures with six targets including two classes of targets with three patterns for each. Figure 3.10 (a) is the binary truth image showing the locations of the inserted targets. The corresponding 3D detection output of the proposed algorithm is shown
in Fig. 3.10 (b). From Fig. 3.10 (b), we can conclude that the proposed algorithm is able to detect multiple dissimilar targets simultaneously with high discriminability. In addition, although the desired peaks are clearly visible and distinct as seen in Fig. 3.10, the difference between the PCM heights of the desired targets may vary, this is due to the dissimilarity in the energy content of the reference signatures, which can be alleviated by adjusting the values of $a_j$, here $j = 1, 2$ since there are two classes. In this experiment, both of the values of $a_1$ and $a_2$ are set to 0.5. A sample application of CSFJTC for specific material detection is provided in Fig. 3.11.

3.4 Summary

In this dissertation, a new deterministic pattern recognition technique in HSI has been proposed. The proposed algorithm is designed to detect multiclass objects consisting of similar and dissimilar target spectral signatures. Since our method is deterministic, no prior training or complex stochastic
analysis is required. This would show the usefulness of the proposed CSFJTC algorithm, particularly when no sufficient training data are available for performing successful object detection tasks. The obtained results suggest robustness and accuracy of CSFJTC, as compared to the other state-of-the-art training-free object detectors in HSI. An observation on the proposed method is that the optimization of filter parameters will have influence on the object discrimination performance. Test results from multiclass object detection showed that CSFJTC with $m = 1$ yields better outputs than that using $m = 2$. In general, considering all comparison results, we can assess that CSFJTC can be a promising candidate for object detection in hyperspectral remote sensing applications. From the output image in Fig. 3.11 can be seen that most of desired regions (colored in red) are successfully detected, which reflects the effectiveness of the proposed CSFJTC.
CHAPTER IV

PROGRESSIVELY EXPANDED NEURAL NETWORK

This chapter introduces a novel feed-forward neural network (NN), named progressively expanded neuron network (PEN Net), which is derived from a biological perspective. The novelty of this technique lies on a neuron structure modeling, where it embeds a nonlinear lines attraction progressively in the input layer and hidden layers. This is quite different from the other types of NNs, such as deep NNs where feature mapping or representation occurs in the hidden layers with complex hierarchical structures. On the other hand, PEN Net is a very efficient learning algorithm that can be easily extended from single hidden layer to multiple hidden layers or very deep NN while preserving its efficiency.

4.1 Introduction and Motivation

A neuron is an excitable cell in the nervous system. The brain is made up of approximately 86 billion neurons [70], and they communicate with each other via either chemical or electrical synapses with fast processing and transmission speed. The brain is a so intelligent system that it can systemically organize tons of neurons to perform difference tasks.

NNs, loosely modeled by the densely hyper-connected neurons of the brain, are computing paradigms which mimic the human learning mechanism by changing the strength of synaptic connections. The output of a NN relies on the cooperation of the individual neuron within the network.
The main objective of NN is to develop a better learning mechanism that allows for the network to perform a variety of tasks with fast speed. Many network architectures have been proposed by researchers on the basis of modeling the neurons and synaptic weights connection. Different types of NNs have their own advantages and disadvantages. Several limitations of current state-of-the-art NNs can be summarized as follows:

- Requires many labeled training data for obtaining satisfactory classification accuracy.
- A large amount of iterations is need to achieve better convergence, such as back-propagation and its variants.
- Computationally inefficient in training such as some of deep neural networks.
- The classification accuracy is very sensitive to parameters setting.
- Requires hand-crafted engineering features in priori to introduce robust nonlinear data transform in the network.
- Requires complex hierarchical structures to perform nonlinear feature mapping for classifying complicated data.

Considering above listed aspects, we intend to develop a new type of NN model that alleviate these limitations as much as possible. Specifically, we will emphasize on two main developments:

- Introduce nonlinear connection between neurons in NN for learning complex data structure, this allows for no or less requirements of hand-crafted engineering features to achieve desired performance.
- Develop an efficient NN structure that provides fast processing speed.
The remainder of this chapter is organized as follows. Section 4.2 presents the proposed NN architecture along with mathematically modeling. Section 4.3 introduces the proposed spectral feature based HSI classification framework. Spectral-spatial features fusion based classification scheme is provided in Section 4.4. Finally, Section 4.5 summarizes this chapter.

4.2 Proposed Neural Network Model

Throughout the study of state-of-the-art NN architectures, it is observed that biologically inspired NNs usually provide superior performance compared to the other NNs or classifiers. Motivated by this, we herein propose a new biologically inspired NN model, which derives from the functioning and structure of the neuron connections in the brain.

4.2.1 Neuroscience Observations

In biological neurons, the connections between neurons takes place across synapses and dendrites receive signals from other neurons at synapses. Studies have shown that dendrites could be decomposed into many independent sub-units of integration [71, 72, 73]. The inspiration of our proposed NN is derived from the following two findings:

- **Nonlinear dendritic summation strengthens computational capability of a single neuron** [73].
  It has proven that a single nonlinear dendritic sub-unit along with the somatic non-linearity can compute linearly non-separable functions.

- **Synapses are originally formed randomly between axon and dendrites in activity-independent manner, and they actively communicate within the same postsynaptic compartment.** [74].

According to these findings, our corresponding hypotheses are summarized as follows:
• **Hypothesis 1:** By introducing nonlinearity in input data (i.e., synaptic inputs or excitations) would also allow for nonlinear sub-unit integration, and thus could enhance computability of a neuron.

• **Hypothesis 2:** Synaptic weights can be randomly generated which are independent of the environments. This hypothesis is also in accord with ELM theory [18] where it has shown that a successful NN can be made by random generation of weights without iteratively tuning process.

Nonlinearity in the network can be introduced by nonlinear line attractor (NLA) NN [75]. NLA network is a recurrent NN which encapsulates the point attractors scattered in the state space as a nonlinear line of attraction instead of point attractors, where fixed points in the line correspond to similar patterns [76]. One of the major drawbacks of this NN is that hidden layers are not incorporated in the network. In other words, the relationship of input node and output node is only considered. This weakens the capability of deeper learning of more complex data. To compensate this limitations, the nonlinear line attractor architecture can be reformed into a feed-forward NN with a polynomial weighting scheme, named a hierarchical autoassociative polynomial network (HAP Net) [77]. HAP net can be easily extended to a deeper network and it has shown competitive results in hand-written digit recognition compared to the other deep NNs such as CNN. However, HAP Net is computationally intensive which includes a number of iterative processes to minimize the error for training the pattern, and it only considers one type of polynomial nonlinear attraction. In contrast, we argue that a family of patterns could be better attracted on a different nonlinear line. Furthermore, the higher-order terms in HAP Net may have less contributions to the network for a normalized input data due to the higher order terms may introduce outliers.

To alleviate these drawbacks, we propose an efficient and effective NN architecture that has several merits: 1) performs fast processing speed; 2) introduces progressive node summation in the
network to avoid insignificant contributions of high-order terms; 3) suggesting a nonlinear synaptic input generation by use of Taylor series.

4.2.2 Proposed Architecture

We propose a new type of feed-forward NN architecture. In this architecture, an excitation or node is decomposed into multiple nodes with different orders and coefficients, then each terms is progressively added to form progressive nonlinear line. This allow each synaptic input progressively scattered on the nonlinear line of attraction. Figure 4.1 shows a comparison between conventional NN model and the proposed model. Here we take two neurons connection case for better illustration. From this figure, it can be seen that the conventional network connection is expressed as

\[ y = g(wx) \]  

(4.1)

where \( x \) is an input node, \( w \) is a weight, \( y \) is the corresponding output node, and \( g \) is the activation function. In contrast, our suggested model generates the output node by

\[
\begin{align*}
y & = g(\sum_{i=1}^{3} w_is_i) \\
& = g(w_1s_1 + w_2s_2 + w_3s_3) \\
& = g(w_1(\alpha_1x^{p_1}) + w_2(\alpha_1x^{p_1} + \alpha_2x^{p_2}) + w_3(\alpha_1x^{p_1} + \alpha_2x^{p_2} + \alpha_3x^{p_3}))
\end{align*}
\]

(4.2)

where \( w_i, p_i \) and \( \alpha_i \) are respectively the weight, power and coefficient of the \( i \)th node, here \( i = 1, 2, 3 \). \( s_i \) is the \( i \)th progressively expanded component.

Nonlinear Line Expansion

Since each nonlinear line is represented as infinite sum of terms that are calculated at a single point, nonlinear line attractor can be associated with a nonlinear function. To formulate the terms in the expansion that generates nonlinear line, we can utilize Taylor series. This allows for a function can be approximated by using a finite number of terms of its Taylor series.
A Taylor series is a series expansion of a function about a fixed point. An expansion of a real function \( f(x) \) at a point \( x = \phi \) is given by

\[
f(x)_{x=\phi} = f(\phi) + \frac{f'(\phi)}{1!}(x - \phi) + \frac{f''(\phi)}{2!}(x - \phi)^2 + \frac{f'''(\phi)}{3!}(x - \phi)^3 + \cdots + \frac{f^n(\phi)}{n!}(x - \phi)^n + \cdots.
\]

(4.3)

A more compact form can be written as

\[
f(x)_{x=\phi} = \sum_{n=0}^{\infty} \frac{f^n(\phi)}{n!}(x - \phi)^n.
\]

(4.4)

where \( n! \) represents the factorial of \( n \) and \( f^n(\phi) \) denotes the \( n \)th derivative of function \( f \) computed at the point \( \phi \). When \( \phi = 0 \), the series is also called a Maclaurin series. Several popular Maclaurin series expansions are given below:
Exponential function:

\[ e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots \]  \hspace{1cm} (4.5)

Geometric series function:

\[ \frac{1}{1-x} = \sum_{n=0}^{\infty} x^n = 1 + x + x^2 + x^3 + \cdots \text{ for } |x| < 1 \]  \hspace{1cm} (4.6)

Trigonometric functions:

\[ \sin(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots \]  \hspace{1cm} (4.7)

Logarithmic function:

\[ \ln(1 + x) = \sum_{n=1}^{\infty} \frac{(-1)^{n+1} x^n}{n} = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \cdots \text{ for } -1 < x \leq 1 \]  \hspace{1cm} (4.8)

Hyperbolic functions:

\[ \sinh(x) = \sum_{n=1}^{\infty} \frac{x^{2n+1}}{(2n+1)!} = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \cdots \]  \hspace{1cm} (4.9)

Given many different types of series expansions, the choice of nonlinear function is an open question. It may vary depending on specific applications.

**Single Hidden Layer Model**

For single hidden layer (SHL) model in PEN Net, input nodes are initially nonlinearly expanded and weighted summed through weight matrix \( w \), then a nonlinear activation function is applied in the hidden layer. Next, the output of activation function, denoted \( h \), is again expanded. Finally the expanded \( h \) is weighted through output weight matrix \( \beta \). In contrast, the conventional NN tends to produce outputs through directly weighted sum along with activation function, note that \( w^{(1)} \) and \( w^{(2)} \) are weight matrices that connect input layer with hidden layer and output layer, respectively. Figure 4.2 shows the proposed architecture versus the conventional one.
Figure 4.2: Conventional NN model versus proposed PEN Net for a case of single hidden layer model with input pattern with 3 excitations.

Given $N$ training samples $\{x_i, y_i\}_{i=1}^N$. Each input variables $x_i \in \mathbb{R}^\ell$ and the target output vector $y_i \in \mathbb{R}^c$, where $\ell$ is the dimension of each input vector, and $c$ denotes the number of object
class in the dataset. Each neuron in \( x_i \), denoted as \( x_{iz} \) for \( z = 1, 2, \cdots, \ell \), is expanded nonlinear summations according to Maclaurin series of a specific nonlinear function as follows:

\[
f(x_{iz}) = \alpha_1 x_{iz}^{p_1} + \alpha_2 x_{iz}^{p_2} + \alpha_3 x_{iz}^{p_3} + \cdots + \alpha_k x_{iz}^{p_k} + \cdots.
\] (4.10)

where \( \alpha_k \) and \( p_k \) are the coefficient and power of the \( k \)th term in the expansion. After expansion, each term in Eq. (4.10) are progressively assigned to a node as shown in Figs. 4.1 and 4.2. The progressive summation avoids decayed contribution of further terms which have higher orders. The progressively summed terms, denoted as \( s_{zk} \). If \( s^{(1)} \) represent the input layer expansion for input pattern \( x_i \), the output function \( h(s^{(1)}) \) with \( L \) hidden nodes is computed by

\[
h(s^{(1)}) = g(ws^{(1)} + b)
\] (4.11)

where \( w \in \mathbb{R}^{L \times K \cdot \ell} \) are the random weights, \( K \) is the total number terms used in the expansion, and \( b = [b_1, b_2, \cdots, b_{K \cdot \ell}] \) are the random biases between the input and hidden nodes. \( g(\cdot) \) is a hidden layer activation function, several popular activation functions are given as follows [78]:

- **Sigmoid function**
  \[
g(w, b, x) = \frac{1}{1 + \exp(-(wx + b))}.
\] (4.12)

- **Gaussian function**
  \[
g(w, b, x) = \exp(-b \| x - w \|^2)
\] (4.13)

- **Fourier function**
  \[
g(w, b, x) = \sin(wx + b)
\] (4.14)

- **Softplus function**
  \[
g(w, b, x) = \ln(1 + \exp(wx + b))
\] (4.15)
Next, the output of hidden node $h$ is expanded into multiple progressively added units as shown in Fig. 4.2. Subsequently, the network output $\hat{y}$ is computed by

$$\hat{y} = s^{(2)} \beta$$

(4.16)

where $\beta$ is weight matrix connecting to the output nodes. Since the objective of network is to minimize the training error, i.e., $\|y - \hat{y}\|_{\text{min}}$, where $y$ is the target output. Thus, the minimum error can be achieved by

$$s^{(2)} \beta = y$$

(4.17)

Accordingly, for $N$ input samples, a more compact form can be written as

$$S^{(2)} B = Y$$

(4.18)

where $Y \in \mathbb{R}^{N \times c}$ is the desired output matrix in training samples and $B \in \mathbb{R}^{(K \cdot \ell) \times c}$. $S^{(2)} \in \mathbb{R}^{N \times (K \cdot \ell)}$ refers to the expanded hidden layer output matrix. $B$ can be estimated by a least squares solution as [25]

$$\hat{B} = (S^{(2)})^\dagger Y$$

(4.19)

where $(S^{(2)})^\dagger$ represents the Moore-Penrose generalized inverse [26, 27] of matrix $S^{(2)}$. Hence, the predicted output matrix, denoted as $\hat{Y}$, for all input samples is expressed by

$$\hat{Y} = S^{(2)} \hat{B} = S^{(2)} (S^{(2)})^\dagger Y$$

(4.20)

From above analysis, it can be observed that learning is made without iteratively tuning weights which significantly reduces computation time, whereas gradient decent methods such as backpropagation requires a large amount of iteration for better convergence, which is very computationally expensive. Note that the regularized least squares estimation can be incorporated in Eq. (4.20) for better generalization purpose.
Multi-Hidden Layer Model

In multi-hidden layer (MHL) case, one of the challenging issues is that the desired output after each hidden layer is unknown until the hidden layer that is adjacent to the final output layer, which makes it difficulty to compute output weight matrix during hidden layers. One of the solutions suggests that output weight $\beta$ can be obtained by projecting the input data onto feature space $\mathbf{h}$ [79] during hidden layers to perform unsupervised learning. In other words, the output weight is accountable for learning the transformation from the feature space to input data. Inspired by this concept, for MHL-PEN Net, the output weights for the $\nu$th (e.g., the total of $V - 1$ hidden layers) hidden layer can be calculated by

$$\hat{\beta}_\nu = \left( (\mathbf{h}^{(\nu)})^T \right)^\dagger (\mathbf{s}^{(\nu)})^T \quad (4.21)$$

and $\mathbf{h}^{(\nu)}$ is computed by

$$\mathbf{h}^{(\nu)} = g(\mathbf{w}_\nu \mathbf{s}^{(\nu)} + \mathbf{b}) \text{ for } \nu = 1, 2, V - 1 \quad (4.22)$$

Then the output of each hidden layer, denoted as $\mathbf{o}^{(\nu)}$, is obtained by

$$\mathbf{o}^{(\nu)} = g(\hat{\beta}_\nu \mathbf{s}^{(\nu)}) \quad (4.23)$$

and

$$\mathbf{s}^{(\nu+1)} = PE(\mathbf{o}^{(\nu)}) \quad (4.24)$$

where $PE(\cdot)$ represents a progressive expansion process using Taylor series of a nonlinear function. Note that the other portion of computation in MHL-PEN Net is performed in the same fashion as SHL case. Figure 4.3 illustrates a MHL structure of the proposed PEN Net in the training process.

In testing stage of the proposed PEN Net as shown in Fig. 4.4, each layer output is obtained by multiplication of the expanded input with trained weights in corresponding layer, followed by
activation function, expressed by

$$h^{(\nu)} = g(\hat{\beta}_v, s^{(\nu)})$$

(4.25)

The other computation processes are similar to SHL model. It is worth mentioning that we can also introduce denosing autoencoders [80, 81] concept into MHL-PEN Net to improve the classification.
accuracy. The reason for introducing the denosing criteria is to make a higher level representation more robust under corruption of the input. As described in [81], there are three different types of corruption processes: 1) add isotropic Gaussian noise to the input randomly; 2) making noise: make a fraction $\nu$ of the input to be 0; 3) salt-and-pepper noise: make a fraction $\nu$ of the input to be their minimum or maximum possible value.

### 4.3 Spectral Feature Based HSI Classification Framework

In this section, we introduce PEN Net for pixel-wise object classification using the pure spectral signature from each labeled pixel in the input data. Our proposed framework for spectral-only based HSI classification is illustrated in Fig. 4.5. As shown in Fig. 4.5, every single pixel in the raw data is collected as a one-dimensional (1-D) vector where its dimension depends on the number of bands in the input HSI.

In training stage, a subset of all labeled pixels are selected and then fed into PEN Net for training, whereas in the testing part, the rest of pixels (excluding pixels that used in the training) are used for pixel class prediction. Note that the entire procedure dose not include preprocessing or any other feature transform, only the pure spectral features are used for hyperspectral data classification.

The choice of SHL-PEN Net and MHL-PEN Net for hyperspectral data classification are user defined. For the data with complex characteristics, such as a big amount of intra-class variation and inter-class variations, one might choose MHL-PEN Net instead of SHL-PEN model, since SHL would not be powerful enough to perform complex nonlinear feature transformation and to differentiate various of pixels from different classes. However, as for less complex data with sufficient training samples, the SHL-PEN model would be a better choice, especially when processing time is a priority in the application.
4.4 Spectral-Spatial Feature Based HSI Classification Framework

Current state-of-the-art hyperspectral image classification methods suggest to incorporate spatial and spectral information which has been shown very promising results even using very small amount of target data for training. In this chapter, we demonstrate a spectral-spatial features based HSI analysis.

Spatial information has shown significant contribution for hyperspectral image classification. Local binary pattern (LBP) [82] has been applied for extracting spatial texture features in HSI classification [50], and it yields significantly better results compared to the other spatial feature based HSI classification techniques. In this method, the LBP code image is generated for each band in the input HSI. To describe the spatial characteristics of the pixel, the LBP histogram for each pixel of interest is computed with its corresponding neighborhood region. However, this method did not consider the texture features from the magnitude component of the image local differences, as well as the local features from multi-resolution of the image. Therefore, we propose a new spatial-feature based HSI classification framework which computes CLBP with multiple scales. Since CLBP contains
local structural components: the difference signs (i.e., original LBP) and the difference magnitudes, we can combine these two components to obtain rich textural information. Furthermore, multiscale analysis in CLBP could be used to further improve the classification accuracy. To the best of our knowledge, this is the first time to exploit multiscale CLBP (MS-CLBP) for HSI classification. In this section, we first give a brief review of LBP and CLBP with their variations for spatial feature extraction, and then introduce the proposed HSI classification framework.

**Local Binary Pattern (LBP)**

LBP is an efficient texture descriptor which obtained by reassigning the pixel labels by comparing the intensity value of the center pixel with that of its related neighbor pixels. The standard version of LBP is formed by thresholding a $3 \times 3$ neighborhood. Let $n_i (i = 0, 1, \ldots, 7)$ be the graylevel of each neighbor pixel and $n_c$ be the center pixel graylevel. If $n_i$ is greater than or equal to $n_c$, the binary result of the pixel is set to 1, otherwise to 0. Consequently, there will be an 8-bit binary pattern to represent the pixel. The decimal value of the binary pattern is the LBP feature.

An arbitrary circular neighborhood can be introduced to the original LBP [82]. It defines the neighborhood as a set of sampling points that evenly distributed on a circle centered at a pixel to be labeled. Let $LBP_{(P,R)}$ denote the LBP feature of a pixels circular neighborhood, where $P$ and $R$ are the number of sampling points and the radius of the circle, respectively. The LBP feature can be calculated as follows:

$$LBP_{P=8, R=1} = T(n_0 - n_c)2^0 + T(n_1 - n_c)2^1 + T(n_2 - n_c)2^2 + T(n_3 - n_c)2^3 + T(n_4 - n_c)2^4 + T(n_5 - n_c)2^5 + T(n_6 - n_c)2^6 + T(n_7 - n_c)2^7$$

Figure 4.6: Illustration of LBP calculation for $P = 8, R = 1$. 
represents total number of sampling points and \( R \) represents the radius of the circle, then \( LBP_{(P,R)} \) can be formulated as:

\[
LBP_{(P,R)} = \sum_{i=0}^{P-1} T(b)2^i
\]  
(4.26)

and

\[
Tr(b) = Tr(n_i - n_c) = \begin{cases} 
1, & b \geq 0 \\
0, & b < 0 
\end{cases}
\]  
(4.27)

Supposed the coordinate of \( n_c \) is \((0, 0)\), then the coordinates of \( n_i \) (neighbors in the circle) are can be calculated by \((-R\sin(2\pi i/P), R\cos(2\pi i/P))\). \( Tr(b) \) is the binary value of each neighbor pixel after being thresholded, and \( b \) represents the difference between the center pixels graylevel and the intensity value of a sampling point on the circle. Figure 4.6 demonstrates a LBP feature calculation for a neighborhood of 8 pixels, i.e, \((P, R) = (8, 1)\), and Fig. 4.7 shows a corresponding example. Fig. 4.8 shows sample outputs of LBP coded image using HSI bands 120 and 200. From the LBP output image can be seen that it depicts texture information of the input scene. The LBP descriptor has been found to be computationally cheaper and robust against illumination changes.

---

<table>
<thead>
<tr>
<th>Input</th>
<th>Differences</th>
<th>Thresholding</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 35 25</td>
<td>-20 5 -5</td>
<td>0 1 0</td>
</tr>
<tr>
<td>25 30 50</td>
<td>-5 20</td>
<td>0 1</td>
</tr>
<tr>
<td>5 30 10</td>
<td>-25 0 -20</td>
<td>0 1</td>
</tr>
</tbody>
</table>

Pattern = 01000101

\[ LBP = 1+0+4+0+0+0+64+0 = 69 \]

Figure 4.7: An example of LBP computation.
Once the LBP pattern of each pixel in an image is obtained, a histogram can be computed to represent as a texture information. LBP captures various patterns in the neighborhood of each pixel. For very smooth regions, the differences between the center pixel graylevel and its neighbors will be zero in all directions, whereas in transition around edges, the LBP operator produces very high difference in the gradient direction.

**Rotation Invariant LBP (LBP-ROT)**

The LBP operator with $P$ neighbors generates $2^P$ different binary patterns. When an image is rotated, the gray values of neighboring pixels will circular shift as shown in Fig. 4.9. Since the starting points in circle for LBP coding is fixed in priori, the image rotating will result in different LBP values, except the patterns only contain 0s or 1s that remain constant at all rotation angles. A rotation invariant LBP can be obtained by

$$LBP_{P,R}^{ri} = \min_{OR}(LBP_{P,R}, i) \text{ for } i = 0, 1, \cdots, P - 1$$  \hspace{1cm} (4.28)
where \( ROR(b, i) \) denotes a circular bit-wise shift \( i \) times on the \( P \)-bit number \( b \). Accordingly, the LBP-ROT is achieved by rotating the neighbor bits \( P \) times until the minimum LBP value is found. For instance, the bit set 10001110, 01000111, and 10100011 produced from different rotations of local pattern will be changed to the minimum LBP code as 01000111.

![Figure 4.9: Effect of image rotation on circular neighborhoods.](image)

**Uniform LBP**

A LBP is called uniform, denoted as \( U(LBP_{P,R}) \), if the bit set contains at most two bitwise transitions from 0 to 1 or vice versa. For example, bit sets 00000001 is a uniform LBP because it has only 2 transition, whereas 1001000 is nonuniform since there is 4 spatial transitions. In the computation of the LBP histogram, uniform patterns are used so that the histogram has a separate bin for every uniform pattern and all non-uniform patterns are assigned to a single bin, which result in \( P \times (P-1) + 3 \) distinct histogram bins. Uniform pattern detects texture primitives (micro-textons) such as flat area, edges, spots, etc.
Uniform LBP-ROT

Combination of uniform pattern with LBP-ROT leads to uniform LBP-ROT, denoted as $LBP_{P,R}^{riu}$. The formulation can be expressed as

$$LBP_{P,R}^{riu} = \begin{cases} \sum_{i=0}^{P-1} T(n_i - n_c), & \text{if } U(LBP_{P,R}) \leq 2 \\ P + 1, & \text{otherwise} \end{cases}$$

(4.29)

Mapping from $LBP_{P,R}$ to $LBP_{P,R}^{riu}$ produces $P + 2$ output values since we assign a unique label to each of uniform pattern and group all nonuniform patterns under a separated label ($P + 1$).

CLBP

CLBP describes a local region by its center pixel ($CLBP_C$) and a local difference of the sign ($CLBP_S$) and magnitude ($CLBP_M$). $CLBP_S$ is the same as the classical LBP, while $CLBP_M$ operator is computed by \[83\]

$$CLBP_{M(P,R)} = \sum_{i=0}^{P-1} t(\bar{m}_i, e)2^i, \quad t(\bar{m}_i, e) = \begin{cases} 1, & \bar{m}_i \geq e \\ 0, & \bar{m}_i < e \end{cases}$$

(4.30)

where $\bar{m}_i = |n_i - n_c|$, and $e$ is a threshold value defined as the mean value of $\bar{m}_i$ from the whole image. To consistent with $CLBP_S$ and $CLBP_M$, the center pixel $CLBP_C$ is defined as

$$CLBP_{C(P,R)} = t(x, c_I)$$

(4.31)

where $c_I$ is set as the mean of all intensity values from the input image. Figure 4.10 shows an example for CLBP operator. Fig. 4.10(a) shows a $3 \times 3$ local region in the image with central pixel being 30, Fig. 4.10 (b) shows the differences between center pixel and its 8 neighbors, and the corresponding sign and magnitude are shown in Figs. 4.10 (c) and 4.10 (d), respectively.

Figure 4.11 illustrates an example of the $CLBP_{S(8,1)}$ and $CLBP_{M(8,1)}$ coded images on the 100th band of Pavia University hyperspectral dataset. It can be observed that $CLBP_S$ and $CLBP_M$ operators both can capture the spatial pattern, such as edges and corners, although $CLBP_S$ operator is able to provide more detailed texture information than the $CLBP_M$ operator. It has been
verified that the fusion of $\text{CLBP}_{S(P,R)}$, $\text{CLBP}_{M(P,R)}$, $\text{CLBP}_{C(P,R)}$ can significantly improve texture classification accuracy [83]. In this work, we only consider the $\text{CLBP}_{S}$ and $\text{CLBP}_{M}$ for computational efficiency.

Figure 4.10: An example for CLBP components. (a) A 3 x 3 sample block; (b) the local differences; (c) the sign and (d) magnitude components.

Figure 4.11: An example of $\text{CLBP}_{S}$ and $\text{CLBP}_{M}$ coded images. (a) Input image (single band), (b) $\text{CLBP}_{S}$ coded image, and (c) $\text{CLBP}_{M}$ coded image.
LBP and CLBP with Multiscale

Multiscale computation can be realized by combining multiple operators at varying parameters \((P, R)\). This is beneficial for image classification since the dominant feature may present at any spatial resolution and using single scale in LBP or CLBP may not guarantee to capture the most discriminate feature. Therefore, it would be great of interest to utilize multiscale LBP and CLBP for a spatial-feature based hyperspectral image classification. For simplicity, we fix the number of neighbors \(P\) and experiment different scale values \(R\) to find better results. However, \(P\) could be varying and combining with different values of \(R\). In order to generate a compact feature vector based on multiscale analysis, histograms of MS-CLBP and MS-LBP are first computed and then concatenated to form the final feature vector for each pixel in HSI. For example, the feature vector for each scale CLBP can be modeled as

\[
\mathbf{f}_{n_c,P,v} = [\text{CLBP}_{S(P,R_v)}, \text{CLBP}_{M(P,R_v)}], \quad v = 1, 2, \ldots, s
\]  

(4.32)

where \(s\) is the total number of scale in the CLBP, and \(\mathbf{f}_{n_c,P,v}\) represents the \(v\)th feature vector for a local region with \(n_c\) as the center pixel. Note that we fix the number of neighbors \(P\) in this representation. An example of a 3-scale CLBP operator is depicted in Fig. 4.12.

![Figure 4.12: An example of 3-scale CLBP operator with \(P = 8\), \(R_1 = 1\), \(R_2 = 2\), and \(R_3 = 3\).](image)
4.4.1 Spatial Feature Extraction From HSI

To make use of spatial information from HSI for improving classification accuracy, MS-CLBP feature extracted from a 2D neighborhood region of each pixel is considered, and then pixel-wise classification is performed by PEN Net. Figure 4.13 depicts a block diagram of the proposed spatial classification strategy, while Fig. 4.14 provides a more detailed graphically illustration.

![Figure 4.13: An overview of spatial feature based classification framework.](image)

In the first step, principal component analysis (PCA) [68] is applied to reduce the HSI data dimension. PCA extracts an image as a set of new orthogonal variables, called principal components (PCs), to display the pattern of similarity of the variables as points in the scene without losing too much information. In our implementation, PCA is conducted along the spectral dimension and we retain the first few PCs for subsequent computation, which dramatically reduces computation complexity in subsequent processes. On the other hand, since feature extraction is performed on each band in HSI, the total concatenated feature dimension may go very large, thus it may be wise to use PCA to extract distinctive band information instead of using all of them.
Next, MS-CLBP is applied to each band of PCA transformed data. Specifically, both multiscale versions of $CLBP_S$ and $MS - CLBP_M$ images are first obtained for each band, and then we compute the corresponding histogram for a predefined neighbor region around the labeled pixel as the center. If we set the number of PCs as $L_1$, number of scales as $L_2$, and the number of histogram bin as $L_3$, then a pixel can be presented as a 1-D vector with a dimension of $L_1 \times L_2 \times L_3$. Figure 4.15 shows a visual description of this procedure.
Finally, the obtained feature vectors are fed into a PEN Net, where the class label for each pixel will be predicted using a SHL-PEN Net or MHL-PEN Net.

### 4.4.2 Proposed Spectral-Spatial Features Based HSI Classification Framework

In the proposed framework, a spectral-spatial classification of HSI is achieved by combining spectral and spatial features to form a hybrid feature vector, and then using PEN Net to assign object category for each pixel from HSI. The corresponding flowchart is shown in Fig. 4.16. The classification steps can be summarized as follows:

1. The raw hyperspectral data is converted as a 1-D vector for each pixel. Pixels from various materials has different spectral signatures.
2. PCA is applied to raw HSI, and first few PCs are selected to represent the input HSI data.

3. Spatial features are extracted from each PCs using MS-CLBP, then feature vectors are concatenated as a 1-D vector to newly represent each pixel in the data.

4. Spectral and spatial features are stacked together to form an integrated representation of original HSI.

5. Finally, the integrated features are fed into a PEN Net to perform classification tasks.

It is worth to mention that using pure pixel spectrum helps to discriminate different class of materials in the scene, while incorporation of spatial information, intra-class variance can be alleviated for the pixels in a neighbor region which can improve the classification performance [36].

Figure 4.16: Proposed spectral-spatial classification framework.
4.5 Summary

In this chapter, we have introduced a new biological inspired NN named PEN Net. We first develop a SHL based PEN Net structure, and then extended SHL-PEN to MHL-PEN. The main merit of the PEN Net is that it transforms the input data into a nonlinear feature space by applying Taylor series of a nonlinear function without complex mathematical formulation. In addition, we also present a new spectral-spatial features based HSI classification framework using MS-CLBP and PEN Net. Introducing spatial information is to reduce the intra-class variances so that improve the classification performance. The integration spectral and spatial features is achieved by using a vector stacking. In other word, every spectral signature of a pixel is stitched to the end of the feature vector for MS-CLBP. Finally, the fused feature is fed into PEN Net to perform the pixel-wise classification.
CHAPTER V

EXPERIMENTS AND DISCUSSION ON HSI CLASSIFICATION

Three real-world hyperspectral datasets are used to verify the effectiveness and efficiency of the proposed PEN Net method. The performance is also compared with other existing state-of-the-art HSI classification algorithms.

5.1 Dataset Description

A summary of experimental HSI datasets is provided in Table 5.1. Apart from two HSI datasets (Indian Pines dataset and Salinas dataset) mentioned in Section 3.3, we also used the University of Pavia dataset 1 for classification performance evaluation. This test site was collected by the Reflective Optics System Imaging Spectrometer (ROSIS) over Pavia, northern Italy. The image has 115 bands of size 610 × 340 pixels with wavelength ranging from 0.43 to 0.86 m and is characterized by a spatial resolution of 1.3-meter per pixel. By removing the noisy bands, the remaining 103 spectral channels were used in the experiment. Nine land-cover classes were identified as shown in Fig. 5.1.

5.2 Experimental Setup

In each classification experiment, the number of training and testing samples are selected randomly. Specifically, we use 30% of labeled samples from Indian Pine dataset for training and the

Table 5.1: Specifications of hyperspectral datasets used in the experiments.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Sensor</th>
<th>Spectral domain (nm)</th>
<th>Spatial domain (m²)</th>
<th># classes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>range</td>
<td>resolution</td>
<td>#bands</td>
</tr>
<tr>
<td>Indian Pines</td>
<td>AVIRIS</td>
<td>400-2500</td>
<td>10</td>
<td>220</td>
</tr>
<tr>
<td>Univ. of Pavia</td>
<td>ROSIS</td>
<td>430-860</td>
<td>4</td>
<td>103</td>
</tr>
<tr>
<td>Salinas</td>
<td>AVIRIS</td>
<td>400-2500</td>
<td>10</td>
<td>204</td>
</tr>
</tbody>
</table>

Figure 5.1: The University of Pavia dataset. (a) False-color composite. (b) Ground truth.

rest of labeled pixels for testing. As for both of the University of Pavia dataset and Salinas dataset, the training set contains 10% samples and testing set consists the rest of 90% samples. It is clear that more percentage of training samples is used for Indian Pine dataset, this is because of some classes contain very few samples (e.g., 20 pixels in Oats), and it would be better for classifiers have sufficient training samples to learn various patterns from statistic point of view. Tables 5.2,
5.3, and 5.4 provide more specific information of the number of training and testing samples for above-mentioned three datasets separately.

Table 5.2: The number of training and testing samples distribution in the Indian Pines dataset.

<table>
<thead>
<tr>
<th>Class Label</th>
<th>Class Name</th>
<th># Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Train</td>
</tr>
<tr>
<td>1</td>
<td>Alfalfa</td>
<td>14</td>
</tr>
<tr>
<td>2</td>
<td>Corn-notill</td>
<td>429</td>
</tr>
<tr>
<td>3</td>
<td>Corn-mintill</td>
<td>249</td>
</tr>
<tr>
<td>4</td>
<td>Corn</td>
<td>72</td>
</tr>
<tr>
<td>5</td>
<td>Grass-pasture</td>
<td>145</td>
</tr>
<tr>
<td>6</td>
<td>Grass-trees</td>
<td>219</td>
</tr>
<tr>
<td>7</td>
<td>Grass-pasture-mowed</td>
<td>9</td>
</tr>
<tr>
<td>8</td>
<td>Hay-windrowed</td>
<td>144</td>
</tr>
<tr>
<td>9</td>
<td>Oats</td>
<td>6</td>
</tr>
<tr>
<td>10</td>
<td>Soybean-notill</td>
<td>292</td>
</tr>
<tr>
<td>11</td>
<td>Soybean-mintill</td>
<td>737</td>
</tr>
<tr>
<td>12</td>
<td>Soybean-clean</td>
<td>178</td>
</tr>
<tr>
<td>13</td>
<td>Wheat</td>
<td>62</td>
</tr>
<tr>
<td>14</td>
<td>Woods</td>
<td>380</td>
</tr>
<tr>
<td>15</td>
<td>Buildings-Grass-Trees-Drives</td>
<td>116</td>
</tr>
<tr>
<td>16</td>
<td>Stone-Steel-Towers</td>
<td>28</td>
</tr>
</tbody>
</table>

We compare our proposed method with SVM, KNN, ELM, nearest regularized subspace (NRS) [84] classifier, and linear discriminant analysis (LDA) [85] based classifier. For the SVM, we use a linear SVM with C-SVC type from LIBSVM [86] software package, and an optimal regularization parameter $C$ is chosen from exponentially growing sequences $\{C = 2^{-5}, 2^{-3}, \cdots, 2^{15}\}$. For KNN, the range of the nearest neighbors are $\{1, 3, 5, \cdots, 30\}$ as suggested in [87] and Euclidean distance is employed. For ELM and the proposed PEN Net, sigmoid function is used as the activation function, the number of neurons are chosen from 100 to 300 at a step of 10, and all of them are

[https://www.csie.ntu.edu.tw/ cjlin/libsvm/](https://www.csie.ntu.edu.tw/ cjlin/libsvm/)
Table 5.3: The number of training and testing samples distribution in the University of Pavia dataset.

<table>
<thead>
<tr>
<th>Class Label</th>
<th>Class Name</th>
<th># Samples</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Train</td>
<td>Test</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Asphalt</td>
<td>664</td>
<td>5967</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Meadows</td>
<td>1865</td>
<td>16784</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Gravel</td>
<td>210</td>
<td>1889</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Trees</td>
<td>307</td>
<td>2757</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Painted metal sheets</td>
<td>135</td>
<td>1210</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Bare Soil</td>
<td>503</td>
<td>4526</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Bitumen</td>
<td>133</td>
<td>1197</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Self-Blocking Bricks</td>
<td>369</td>
<td>3313</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Shadows</td>
<td>95</td>
<td>852</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.4: The number of training and testing samples distribution in the Salinas dataset.

<table>
<thead>
<tr>
<th>Class Label</th>
<th>Class Name</th>
<th># Samples</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Train</td>
<td>Test</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Brocoli green weeds 1</td>
<td>201</td>
<td>1808</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Brocoli green weeds 2</td>
<td>373</td>
<td>3353</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Fallow</td>
<td>198</td>
<td>1778</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Fallow_rough_plow</td>
<td>140</td>
<td>1254</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Fallow_smooth</td>
<td>268</td>
<td>2410</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Stubble</td>
<td>396</td>
<td>3563</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Celery</td>
<td>358</td>
<td>3221</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Grapes_untrained</td>
<td>1128</td>
<td>10143</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Soil_vinyard_develop</td>
<td>621</td>
<td>5582</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Corn_senesced_green_weeds</td>
<td>328</td>
<td>2950</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Lettuce_romaine_4wk</td>
<td>107</td>
<td>961</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Lettuce_romaine_5wk</td>
<td>193</td>
<td>1734</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Lettuce_romaine_6wk</td>
<td>92</td>
<td>824</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Lettuce_romaine_7wk</td>
<td>107</td>
<td>963</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Vinyard_untrained</td>
<td>727</td>
<td>6541</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Vinyard_vertical_trellis</td>
<td>181</td>
<td>1626</td>
<td></td>
</tr>
</tbody>
</table>
implemented without regularization terms. For MHL-PEN Net, we use two hidden layers in the experiments for computational efficiency. For NRS, the global regularization parameter is selected over the range of \( \{10^{-3}, 10^{-2}, 10^{-1}, \ldots, 10^3\} \). In our experiments, all related parameters tuning, if not specified, is conducted using grid search with 3-fold cross-validation on available training set, and only the optimal ones are used in the testing.

In evaluation, all experiments are repeated for 10 times with random selection of training and testing samples to avoid any bias, and the averaged results were calculated and evaluated in term of overall accuracy (OA), individual class accuracy and processing time (training + testing). For each HSI dataset, all competing methods use the same training samples and testing samples to produce a fair comparison. The computing platform is an Intel Core i7-3630MQ 2.4-GHz machine with 24 GB of RAM, and all implementations is performed on MATLAB-2016a.

5.3 Results and Comparisons

5.3.1 Analysis of PEN Net configuration

In our proposed PEN Net, there could be various configurations which may lead to different results. The difference in configuration is mainly derived from two aspects: types of nonlinear function and number of terms in the function. In the following, we will investigate these two aspects to better understand the characteristics of PEN Net.

Function Type Versus Number of Terms

As mentioned in Chapter IV, one of main process in PEN Net is to incorporate Taylor series of a predefined nonlinear function. It thus introduces the selection of the nonlinear function with the number of expanded terms in the nonlinear form. we herein investigate the relationship between difference types of nonlinear functions and the number of terms used in the expansion. Tables 5.5, 5.6, and 5.7 present the overall accuracy of SHL-PEN Net using different nonlinear functions with
various number of terms. From these results, it can be seen that using more than one term in the expansion always provides better accuracy than using only one term. This is because more terms introduce higher dimensional feature space, which improves the capability of pattern discrimination. However, the amount of terms should not be too large, which may cause over-fitting and thus deteriorates classification accuracy as well as processing speed. This statement can be verified from Table 5.6, as the number of terms increases from 3 to 5, the OA decreases. It can be also observed that different type expansions on a fixed term numbers yield similar accuracy, although the function $\ln(1 + x)$ provides superior performance compared to the other two competitor overall, since it achieves two times highest OA (the highest OA is marked in boldface) when using the aforementioned three datasets.

Table 5.5: Accuracy of function type with respect to the number of terms in the Indian Pines dataset.

<table>
<thead>
<tr>
<th># Terms</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functions</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\sin x$</td>
<td>79.31</td>
<td>82.55</td>
<td>82.95</td>
<td>82.78</td>
<td>82.50</td>
</tr>
<tr>
<td>$\ln (1 + x)$</td>
<td>79.31</td>
<td>82.73</td>
<td>83.38</td>
<td>82.91</td>
<td>82.31</td>
</tr>
<tr>
<td>$\arctan x$</td>
<td>79.31</td>
<td>82.58</td>
<td>82.90</td>
<td>82.42</td>
<td>81.89</td>
</tr>
</tbody>
</table>

5.3.2 Quantitative Evaluation on HSI Classification

Since spatial feature extraction using MS-CLBP is applied to each single band, more spectral bands causes higher dimension of spatial features and computational complexity. Thus, it is suggested to use a band selection technique in priori to reduce HSI dimension. In our implementation, we fix the number of PCs to be 5 for all datasets as it is also used in [36]. The optimal window
Table 5.6: Accuracy of function type with respect to the number of terms in the University of Pavia dataset.

<table>
<thead>
<tr>
<th># Terms</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functions</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\sin x$</td>
<td>90.16</td>
<td>92.51</td>
<td>92.66</td>
<td>92.34</td>
<td>92.27</td>
</tr>
<tr>
<td>$\ln (1 + x)$</td>
<td>90.16</td>
<td>92.44</td>
<td>92.69</td>
<td>92.26</td>
<td>91.69</td>
</tr>
<tr>
<td>$\arctan x$</td>
<td>90.16</td>
<td>92.55</td>
<td>92.65</td>
<td>92.34</td>
<td>91.14</td>
</tr>
</tbody>
</table>

Table 5.7: Accuracy of function type with respect to the number of terms in the Salinas dataset.

<table>
<thead>
<tr>
<th># Terms</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functions</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\sin x$</td>
<td>92.19</td>
<td>93.22</td>
<td>93.61</td>
<td>93.74</td>
<td><strong>93.76</strong></td>
</tr>
<tr>
<td>$\ln (1 + x)$</td>
<td>92.19</td>
<td>93.19</td>
<td>93.66</td>
<td>93.69</td>
<td>93.60</td>
</tr>
<tr>
<td>$\arctan x$</td>
<td>92.19</td>
<td>93.21</td>
<td>93.63</td>
<td>93.61</td>
<td>93.51</td>
</tr>
</tbody>
</table>

size for computing histograms based on MS-CLBP features is highly relying on the image spatial resolution and land cover structure. In general, one should choose window size that can cover the entire texture pattern while include one type of material [88]. Following the parameters setting suggested in [50], we choose window sizes for Indian Pines, University of Pavia and Salinas datasets as $17 \times 17$, $21 \times 21$, and $25 \times 25$, respectively. In CLBP, radius $R$ defines the area of selecting circular neighbors, whereas $P$ controls the dimensionality of CLBP histogram. Considering computational efficiency, we fix the $P$ value to be 8, and employ rotation-invariant uniform CLBP. After obtaining all spatial features, the raw spectral signatures are stacked with the corresponding spatial features.
for each labeled pixel in the dataset, then fed into the PEN Net for training and predicting class label. Tables 5.8, 5.9, and 5.10 shows the averaged classification accuracy over the ten trials of repeated experiments using the aforementioned three datasets.

Table 5.8: Accuracy of every class (%), OA (%) for the Indian Pines dataset using 30% training samples. (The best accuracy is highlighted in boldface)

<table>
<thead>
<tr>
<th>Class Labels</th>
<th>SVM</th>
<th>ELM</th>
<th>KNN</th>
<th>NRS</th>
<th>LDA</th>
<th>SHL-PEN</th>
<th>MHL-PEN</th>
<th>SS-SHL-PEN</th>
<th>SS-MHL-PEN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>56.56</td>
<td>7.81</td>
<td>38.44</td>
<td>54.38</td>
<td>72.50</td>
<td>52.81</td>
<td>51.56</td>
<td><strong>95.94</strong></td>
<td>95.63</td>
</tr>
<tr>
<td>2</td>
<td>79.21</td>
<td>78.99</td>
<td>58.90</td>
<td>54.84</td>
<td>72.55</td>
<td>82.28</td>
<td>81.15</td>
<td>97.04</td>
<td>96.99</td>
</tr>
<tr>
<td>3</td>
<td>56.18</td>
<td>56.63</td>
<td>54.32</td>
<td>43.48</td>
<td>62.20</td>
<td>62.60</td>
<td>63.86</td>
<td>97.14</td>
<td><strong>97.35</strong></td>
</tr>
<tr>
<td>4</td>
<td>32.67</td>
<td>35.39</td>
<td>38.61</td>
<td>74.06</td>
<td>62.12</td>
<td>51.82</td>
<td>57.88</td>
<td>97.45</td>
<td>96.36</td>
</tr>
<tr>
<td>5</td>
<td>88.43</td>
<td>88.93</td>
<td>84.94</td>
<td>87.99</td>
<td>87.51</td>
<td>89.64</td>
<td>90.21</td>
<td><strong>97.84</strong></td>
<td>97.37</td>
</tr>
<tr>
<td>6</td>
<td>97.05</td>
<td>97.93</td>
<td>96.71</td>
<td>93.13</td>
<td>92.04</td>
<td>98.20</td>
<td>98.04</td>
<td><strong>99.86</strong></td>
<td>99.78</td>
</tr>
<tr>
<td>7</td>
<td>55.79</td>
<td>59.26</td>
<td>85.26</td>
<td>77.37</td>
<td>87.37</td>
<td>67.89</td>
<td>62.63</td>
<td><strong>93.68</strong></td>
<td>91.58</td>
</tr>
<tr>
<td>8</td>
<td>98.95</td>
<td>99.73</td>
<td>98.26</td>
<td>97.10</td>
<td>97.90</td>
<td>99.52</td>
<td>99.16</td>
<td><strong>100.00</strong></td>
<td>99.91</td>
</tr>
<tr>
<td>9</td>
<td>21.43</td>
<td>0.00</td>
<td>24.29</td>
<td>16.45</td>
<td>64.29</td>
<td>28.57</td>
<td>26.43</td>
<td>80.00</td>
<td><strong>84.29</strong></td>
</tr>
<tr>
<td>10</td>
<td>64.81</td>
<td>62.57</td>
<td>74.94</td>
<td>74.41</td>
<td>60.37</td>
<td>73.38</td>
<td>76.46</td>
<td>96.10</td>
<td>96.07</td>
</tr>
<tr>
<td>11</td>
<td>83.07</td>
<td>84.19</td>
<td>78.03</td>
<td>77.18</td>
<td>77.39</td>
<td>84.89</td>
<td>86.23</td>
<td>98.68</td>
<td><strong>98.91</strong></td>
</tr>
<tr>
<td>12</td>
<td>69.83</td>
<td>73.98</td>
<td>47.32</td>
<td>55.69</td>
<td>78.36</td>
<td>77.66</td>
<td>77.93</td>
<td><strong>97.35</strong></td>
<td>97.16</td>
</tr>
<tr>
<td>13</td>
<td>97.20</td>
<td>97.69</td>
<td>92.73</td>
<td>91.89</td>
<td>96.71</td>
<td>97.41</td>
<td>97.41</td>
<td><strong>98.60</strong></td>
<td>98.95</td>
</tr>
<tr>
<td>14</td>
<td>96.99</td>
<td>96.85</td>
<td>94.29</td>
<td>92.72</td>
<td>93.32</td>
<td>96.55</td>
<td>96.70</td>
<td>99.95</td>
<td>99.85</td>
</tr>
<tr>
<td>15</td>
<td>61.67</td>
<td>56.63</td>
<td>33.59</td>
<td>44.81</td>
<td>63.07</td>
<td>64.63</td>
<td>62.41</td>
<td><strong>98.30</strong></td>
<td>98.15</td>
</tr>
<tr>
<td>16</td>
<td>82.92</td>
<td>77.54</td>
<td>87.69</td>
<td>87.69</td>
<td>86.77</td>
<td>62.92</td>
<td>64.46</td>
<td>91.38</td>
<td><strong>94.31</strong></td>
</tr>
<tr>
<td>OA</td>
<td>79.57</td>
<td>79.31</td>
<td>73.53</td>
<td>73.20</td>
<td>77.93</td>
<td>82.73</td>
<td>83.37</td>
<td>98.10</td>
<td><strong>98.11</strong></td>
</tr>
<tr>
<td>Time in Sec. (Train + Test)</td>
<td>184.16</td>
<td>0.23</td>
<td>1.26</td>
<td>211.53</td>
<td>0.65</td>
<td>0.60</td>
<td>1.07</td>
<td>11.06</td>
<td>11.36</td>
</tr>
</tbody>
</table>

For PEN Net, we use 2 terms in the expansion with Taylor series of the function $ln(1 + x)$, and only 1 scale from MS-CLBP is extracted for generating spatial features. One may use multiple scales to get better accuracy, however it will be more time-consuming. All the other parameters in competing methods are set as described in Section 5.2. As can be observed, our proposed spectral-spatial (SS) features based PEN Net classification schemes (i.e., SS-SHL-PEN and SS-MHL-PEN) achieved significantly better accuracy for all three datasets, even provide the higher accuracy for every class in most of the experiments. Furthermore, our SHL-PEN Net and MHL-PEN Net yield better OA than SVM, ELM, KNN, NRS and LDA. Comparing SHL-PEN Net and MHL-PEN Net,
Table 5.9: Accuracy of every class (\%), OA (\%) for the University of Pavia dataset using 10% of training samples.

<table>
<thead>
<tr>
<th>Class Labels</th>
<th>SVM</th>
<th>ELM</th>
<th>KNN</th>
<th>NRS</th>
<th>LDA</th>
<th>SHL-PEN</th>
<th>MHL-PEN</th>
<th>SS-SHL-PEN</th>
<th>SS-MHL-PEN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>91.33</td>
<td>93.35</td>
<td>86.76</td>
<td>89.97</td>
<td>88.56</td>
<td>92.40</td>
<td>91.58</td>
<td>99.19</td>
<td><strong>99.23</strong></td>
</tr>
<tr>
<td>2</td>
<td>97.33</td>
<td>99.11</td>
<td>96.27</td>
<td>97.98</td>
<td>93.27</td>
<td>98.90</td>
<td>98.82</td>
<td>99.97</td>
<td><strong>99.98</strong></td>
</tr>
<tr>
<td>3</td>
<td>68.49</td>
<td>60.57</td>
<td>66.31</td>
<td>63.95</td>
<td>65.11</td>
<td>68.35</td>
<td>71.79</td>
<td><strong>97.61</strong></td>
<td>97.26</td>
</tr>
<tr>
<td>4</td>
<td>84.71</td>
<td>91.58</td>
<td>82.38</td>
<td>78.44</td>
<td>85.74</td>
<td>93.65</td>
<td>92.87</td>
<td>98.80</td>
<td>98.66</td>
</tr>
<tr>
<td>6</td>
<td>59.66</td>
<td>74.81</td>
<td>60.20</td>
<td>58.61</td>
<td>63.08</td>
<td>83.07</td>
<td>83.60</td>
<td>99.54</td>
<td><strong>99.61</strong></td>
</tr>
<tr>
<td>7</td>
<td>27.35</td>
<td>33.25</td>
<td>82.58</td>
<td>63.17</td>
<td>42.18</td>
<td>64.12</td>
<td>76.32</td>
<td><strong>98.44</strong></td>
<td>98.30</td>
</tr>
<tr>
<td>8</td>
<td>68.51</td>
<td>90.61</td>
<td>84.29</td>
<td>81.25</td>
<td>75.46</td>
<td>91.39</td>
<td>90.97</td>
<td>98.86</td>
<td><strong>98.90</strong></td>
</tr>
<tr>
<td>9</td>
<td>99.15</td>
<td>99.31</td>
<td><strong>99.82</strong></td>
<td>92.41</td>
<td>99.50</td>
<td>98.99</td>
<td>93.84</td>
<td>99.64</td>
<td>98.70</td>
</tr>
</tbody>
</table>

Time in Sec. (Train + Test) | 182.71 | 0.60 | 4.53 | 4584.32 | 0.62 | 1.28 | 1.66 | 55.76 | 57.13 |

Table 5.10: Accuracy of every class (\%), OA (\%) for the Salinas dataset using 10% training samples.

<table>
<thead>
<tr>
<th>Class Labels</th>
<th>SVM</th>
<th>ELM</th>
<th>KNN</th>
<th>NRS</th>
<th>LDA</th>
<th>SHL-PEN</th>
<th>MHL-PEN</th>
<th>SS-SHL-PEN</th>
<th>SS-MHL-PEN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>99.81</td>
<td>99.54</td>
<td>97.85</td>
<td>99.42</td>
<td>99.73</td>
<td>99.71</td>
<td>99.73</td>
<td><strong>100.00</strong></td>
<td>100.00</td>
</tr>
<tr>
<td>3</td>
<td>99.24</td>
<td>94.78</td>
<td>98.95</td>
<td>99.75</td>
<td>97.77</td>
<td>99.55</td>
<td>99.84</td>
<td>99.99</td>
<td><strong>100.00</strong></td>
</tr>
<tr>
<td>4</td>
<td>99.23</td>
<td>98.78</td>
<td>99.37</td>
<td>99.29</td>
<td>97.26</td>
<td>98.30</td>
<td>98.56</td>
<td><strong>99.91</strong></td>
<td>99.80</td>
</tr>
<tr>
<td>5</td>
<td>99.29</td>
<td>99.25</td>
<td>96.65</td>
<td>98.22</td>
<td>98.68</td>
<td>99.11</td>
<td>99.05</td>
<td>99.35</td>
<td><strong>99.41</strong></td>
</tr>
<tr>
<td>7</td>
<td>99.89</td>
<td>99.69</td>
<td>99.34</td>
<td>99.66</td>
<td>99.75</td>
<td>99.78</td>
<td>99.80</td>
<td>99.91</td>
<td><strong>99.89</strong></td>
</tr>
<tr>
<td>8</td>
<td>87.80</td>
<td>89.82</td>
<td>79.67</td>
<td>94.45</td>
<td>95.66</td>
<td>90.68</td>
<td>90.59</td>
<td>99.85</td>
<td><strong>99.87</strong></td>
</tr>
<tr>
<td>9</td>
<td>99.95</td>
<td>99.94</td>
<td>98.88</td>
<td>99.87</td>
<td>99.97</td>
<td>99.95</td>
<td>99.94</td>
<td><strong>100.00</strong></td>
<td>100.00</td>
</tr>
<tr>
<td>10</td>
<td>96.00</td>
<td>95.75</td>
<td>91.26</td>
<td>97.19</td>
<td>93.80</td>
<td>96.80</td>
<td>96.52</td>
<td><strong>99.82</strong></td>
<td>99.78</td>
</tr>
<tr>
<td>11</td>
<td>97.61</td>
<td>94.98</td>
<td>95.50</td>
<td>95.76</td>
<td>92.65</td>
<td>96.70</td>
<td>98.37</td>
<td><strong>100.00</strong></td>
<td>100.00</td>
</tr>
<tr>
<td>12</td>
<td>99.91</td>
<td>99.99</td>
<td>99.82</td>
<td>99.86</td>
<td>99.96</td>
<td>100.00</td>
<td>100.00</td>
<td><strong>100.00</strong></td>
<td>100.00</td>
</tr>
<tr>
<td>13</td>
<td>98.94</td>
<td>98.67</td>
<td>97.32</td>
<td>98.47</td>
<td>99.22</td>
<td>98.77</td>
<td>98.22</td>
<td>98.93</td>
<td><strong>99.05</strong></td>
</tr>
<tr>
<td>14</td>
<td>96.54</td>
<td>92.73</td>
<td>93.36</td>
<td>96.91</td>
<td>92.15</td>
<td>94.34</td>
<td>95.67</td>
<td>98.39</td>
<td><strong>98.68</strong></td>
</tr>
<tr>
<td>15</td>
<td>66.95</td>
<td>64.29</td>
<td>63.50</td>
<td>56.18</td>
<td>66.25</td>
<td>69.02</td>
<td>71.19</td>
<td>99.48</td>
<td><strong>99.57</strong></td>
</tr>
<tr>
<td>16</td>
<td>98.98</td>
<td>98.64</td>
<td>97.79</td>
<td>98.38</td>
<td>98.25</td>
<td>98.76</td>
<td>98.76</td>
<td><strong>99.98</strong></td>
<td>99.93</td>
</tr>
</tbody>
</table>

Time in Sec. (Train + Test) | 347.96 | 0.91 | 15.18 | 3517.27 | 2.72 | 1.87 | 2.3   | 87.71 | 88.22 |

MHL-PEN Net provides slightly higher accuracy in most cases. As for computation time, ELM acts as the fastest classifier among all other methods, while our PEN Net, especially SHL-PEN Net
is also providing very fast processing speed with much better accuracy than ELM. The SS based PEN Nets require longer processing time compared to spectral-only PEN Net based classification scheme, this is due to the time delay for extracting spatial features.

5.3.3 Visual Inspection on Classification Performance

For illustrative purposes, we demonstrate classification performance on the landscape using HSI. Figs. 5.2, 5.3, and 5.4 show the ground truth and classification maps obtained by the different tested methods for the Indian Pines, University of Pavia and Salinas datasets, respectively. For each method, we randomly selected one of the maps in the ten trials.

From these figures, it can be seen that our PEN Nets, especially SS-SHL-PEN and SS-MHL-PEN, exhibit spatial smoothness within all labeled ground categories, and they look almost similar to the ground truth maps which indicates high classification accuracy. In contrast, SVM, ELM, KNN, NRS, and LDA produce more noisy pixel mapping where many misclassification can be found in all three test sites. For example, in Fig. 5.3, the misclassification occurred heavily between the class of Meadows and Bare Soil, whereas PEN Net based methods significantly reduce these incorrect classification.
Figure 5.2: Classification maps of different methods on the Indian Pines dataset.
Figure 5.3: Classification maps of different methods on the University of Pavia dataset.
Figure 5.4: Classification maps of different methods on the Salinas dataset.
5.3.4 Effect of Different Number of Training Samples

In this section, we conduct several experiments to examine the influence of the amount of training data on the classification accuracy. In each test, we randomly choose various number of labeled samples for Indian Pines, University of Pavia and Salinas datasets. We execute 10 runs test for ELM, KNN, LDA, and SHL-PEN Net, and report the average OA versus the number of training samples. All other parameters in the algorithms are set as described in Section 5.2, except the number of training samples is varying from $10\% - 50\%$ with an increment of $5\%$.

Figure 5.5 illustrates the OA (averaged over ten runs) for each classifier under different numbers of training samples in the three datasets. It is obvious that the performance for all the classifiers generally improves as the numbers of training samples increases. Additionally, the proposed PEN Net (here we use SHL-PEN with 2 terms in the Taylor expansion of $ln(1 + x)$) consistently yields higher OA against the other three methods regardless of smaller or larger amount of training sets. Besides, classification performance of all methods behaves differently for different datasets. For example, in Fig. 5.5(a), OA goes higher as the number of training samples increases, whereas the OA of all algorithms toward to be stable when the number of training samples is more than $30\%$ when using the University of Pavia dataset as shown in Fig. 5.5(b).

5.3.5 Effect of Multiscale Features in MS-CLBP

In the proposed SS based classification framework, selecting optimal number of scale for MS-CLBP is important. In order to investigate the effects of the scaling factor (i.e., radius $R$) on the classification performance, we first randomly choose training samples and take the rest of the data as testing samples. Then we fix the value $P = 8$ and vary the scale $R$ from 1 to 5 to conduct experiments. Since the scale is set from 1 to 5, there are 5 choices for MS-CLBP: $[1], [1 : 2], [1 : 3], [1 : 4], [1 : 5]$, where $[1 : \sigma]$ indicates the feature combination of the scales from 1 to $\sigma$ for
σ = 1, 2, ..., 5. In classification, we use SS-SHL-PEN model with 100 units in the hidden layer and 2 terms in the expansion of function \( \ln(1 + x) \).

Figure 5.5: OA of PEN Net (SHL model), ELM, KNN, LDA versus the number of training samples in (a) Indian Pines dataset, (b) University of Pavia dataset, and (c) Salinas dataset.

Figures 5.6 and 5.7 present the classification results in terms of OA (averaged over 10 trials) and processing time using different number of scales for the three datasets. From these results, it is observed that the relationship between the highest accuracy and scales varies, this is due to these datasets contain different landscape and materials which will affect the choice of scales in
MS-CLBP feature as well as classification accuracy. For example, the scale set [1 : 3] works best for Indian Pines dataset, [1 : 2] for the Pavia University dataset, and [1 : 5] for the Salinas dataset.
Furthermore, it can be seen that using more scales leads to higher computational cost as depicted in Fig. 5.7, although it sometimes provide higher accuracy.

5.3.6 **Comparison with the other spectral-spatial based classification methods**

Finally, we compare our PEN Net with very recently published spectral-spatial based HSI classification algorithms. The competing methods include contextual SVM (CSVM) [89], contextual ELM (CELM) [89], local receptive field based ELM (L-ELM) [90], hierarchical local receptive field based ELM (HL-ELM) [89], CNN [91], spatial-aware dictionary learning (SADL) [92], edge-preserving filtering (EPF) [43], SAE with logistic regression (SAE-LR) [35], guided filter based fine-tuning SAE (GF-FSAE) [93]. The OA of CSVM, CELM, L-ELM, HL-ELM, CNN and SADL in Table 5.11 (a) are taken from [89], whereas the results of EPF, SAE-LR and GF-FSAE in Table 5.11(b) are from [93]. For fair comparison, we used the same amount of training and testing samples for each competed methods, and all experiments are repeated 10 times and averaged OA is presented. From Table 5.11, it is evident that our proposed SS-SHL-PEN is much more effective on HSI classification. It is worth mentioning that we only use 2 scales in MS-CLBP features and 2 terms in the Taylor series of function \( \ln(1 + x) \). Thus, higher accuracy may be achieved by various combination of different scales in MS-CLBP and different number of terms in the expansion.

5.3.7 **Summary**

In this chapter, we introduced a NN model named PEN Net. Throughout performance examination in terms of accuracy and processing speed, we found that PEN Net is robust for HSI classification regardless of sensor differences, and it provides better accuracy and faster or competitive processing speed compared to the state-of-the-arts. Additionally, our spectral-spatial features based HSI classification scheme is proven to be more compelling than spectral-only based HSI classification strategy. Furthermore, it is observed that when we use more than one term in the expansion
Table 5.11: Performance Comparison of spectral-spatial features based HSI classification methods.

<table>
<thead>
<tr>
<th>Methods</th>
<th>CSVM</th>
<th>CELM</th>
<th>L-ELM</th>
<th>HL-ELM</th>
<th>CNN</th>
<th>SADL</th>
<th>SS-SHL-PEN (Proposed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>OA (Univ. of Pavia dataset)</td>
<td>96.01</td>
<td>94.60</td>
<td>97.76</td>
<td>98.59</td>
<td>97.24</td>
<td>98.47</td>
<td><strong>99.72</strong></td>
</tr>
</tbody>
</table>

(a) Comparison with spectral-spatial based classification methods in [89].

<table>
<thead>
<tr>
<th>Methods</th>
<th>Datasets</th>
<th>Salinas</th>
<th>Univ. of Pavia</th>
</tr>
</thead>
<tbody>
<tr>
<td>EPF</td>
<td></td>
<td>99.09</td>
<td>99.28</td>
</tr>
<tr>
<td>SAE-LR</td>
<td></td>
<td>97.57</td>
<td>97.76</td>
</tr>
<tr>
<td>GF-FSAE</td>
<td></td>
<td>99.64</td>
<td>99.51</td>
</tr>
<tr>
<td>SS-SHL-PEN (Proposed)</td>
<td></td>
<td><strong>99.93</strong></td>
<td><strong>99.83</strong></td>
</tr>
</tbody>
</table>

(b) Comparison with spectral-spatial based classification methods in [93].

of PEN Net, it produces better accuracy than without using any expansion (i.e., raw input). However, using more number of terms and MS-CLBP scales will increase computational cost. In our future work, the research focus will on the balancing of classification accuracy and computational efficiency.
In this dissertation, two novel approaches were presented for HSI analysis: 1) class-associative spectral fringe-adjusted joint transform correlation (CSFJTC) for multi-class object detection, and 2) progressively expanded neural network (PEN Net) for object classification. The effectiveness of the methods was examined using different real-world HSI datasets, as well as comparison with the state-of-the-art methods.

The first part of this research focused on development of a deterministic object detection algorithm capable of detecting multiclass objects in HSI. The main challenge was to simultaneously detect objects that are from multiple landscape categories without any training. To tackle this problem, a technique named CSFJTC is developed. The key process of CSFJTC is based on joint signal correlation and filtering. The output of CSFJTC provides a pair of sharp and high correlation peaks for a matched unknown input and negligible or no correlation peaks for a mismatch. In other words, if the desired patterns present in the scene, CSFJTC yields distinctive correlation peaks simultaneously irrespective of tasks for detecting single class or multiclass targets. From evaluation results, it is found that CSFJTC algorithm can successfully perform multi-class object detection tasks and exceed the state-of-the-art detectors. In our future work, improvements may be achieved by adaptively adjusting JPS coefficients and the parameters in CSGFAF.
The second part of this research concentrated on constructing a new type of neural work for object classification. The goal is to achieve an effective and efficient machine learning algorithm. From studies of neuroscience, two major observations were made and a new structured of neural network, i.e., PEN Net, was formed. The underlying idea of PEN Net is to efficiently transform input data to a higher dimensional feature space for better pattern differentiation. Based on this concept, Taylor series of a nonlinear function is introduced into neural network, which does not require complex mathematical formulation but reaches satisfactory classification accuracy and processing speed. Detailed comparisons were made with state-of-the-art techniques qualitatively and quantitatively, and it is observed that PEN Net shows sound performance in terms of accuracy and computational efficiency. Furthermore, a spectral-spatial HSI classification framework is also developed on the basis of PEN Net. The combination of spectral-spatial features significantly improved overall accuracy compared to spectral-only based classification strategy. The limitation is that the spatial feature extraction process slows down the processing speed. In future work, an efficient spectral-spatial HSI classification scheme will be investigated and an advanced multi-feature fusion technique will be researched.

In future exploration, it would be a great interest to apply PEN Net for various applications, such as handwritten digit recognition, automatic target tracking, speech recognition, image enhancement, etc., so that it will extensively show its robustness and usefulness, and further express significance of this research.
CHAPTER VII

RESEARCH PUBLICATIONS

Peer-Reviewed Journal Publications:


**Journal Papers in Review:**


**Journal Papers in Preparation:**


12. Evan Krieger, Sidike Paheding, and Vijayan K. Asari, “Robust object tracking through multi-

13. ALmabrok Essa, Sidike Paheding, and Vijayan K. Asari, “Volumetric textural features for

*Conference Proceedings:*

features for robust object tracking,” IEEE Applied Imagery Pattern Recognition Workshop:
Imaging and Artificial Intelligence: Intersection and Synergy - AIPR 2016, Washington DC,
USA, October 18-20, 2016.

15. Sidike Paheding, Chen Chen, Vijayan K. Asari, Yan Xu, and Wei Li, “Classification of
hyperspectral image using multiscale spatial texture features,” *IEEE 8th Workshop on Hyper-
spectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, August
2016.

change detection through adaptive local textural features and sequential background removal,”
*IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, Beijing, China,

objects using ringlet-based pyramid Fourier histogram of oriented gradients,” *SPIE Defense


**Abstracts and Posters Presentation:**


Thesis Publications:

BIBLIOGRAPHY


