Western Graduate&PostdoctoralStudies

Western University
Scholarship@Western

Electronic Thesis and Dissertation Repository

8-16-2018 11:00 AM

Automatic Image Classification for Planetary Exploration

Lei Shu
The University of Western Ontario

Supervisor
McIsaac, Kenneth
The University of Western Ontario Co-Supervisor
Osinski, Gordon R.
The University of Western Ontario

Graduate Program in Electrical and Computer Engineering
A thesis submitted in partial fulfillment of the requirements for the degree in Doctor of Philosophy
© Lei Shu 2018

Follow this and additional works at: https://ir.lib.uwo.ca/etd

Part of the Computer Engineering Commons, and the Geological Engineering Commons

Recommended Citation
Shu, Lei, "Automatic Image Classification for Planetary Exploration" (2018). *Electronic Thesis and Dissertation Repository*. 5705.
https://ir.lib.uwo.ca/etd/5705

This Dissertation/Thesis is brought to you for free and open access by Scholarship@Western. It has been accepted for inclusion in Electronic Thesis and Dissertation Repository by an authorized administrator of Scholarship@Western. For more information, please contact wlswadmin@uwo.ca.

Abstract

Autonomous techniques in the context of planetary exploration can maximize scientific return and reduce the need for human involvement. This thesis work studies two main problems in planetary exploration: rock image classification and hyperspectral image classification. Since rock textural images are usually inhomogeneous and manually hand-crafting features is not always reliable, we propose an unsupervised feature learning method to autonomously learn the feature representation for rock images. The proposed feature method is flexible and can outperform manually selected features. In order to take advantage of the unlabelled rock images, we also propose self-taught learning technique to learn the feature representation from unlabelled rock images and then apply the features for the classification of the subclass of rock images.

Since combining spatial information with spectral information for classifying hyperspectral images (HSI) can dramatically improve the performance, we first propose an innovative framework to automatically generate spatial-spectral features for HSI. Two unsupervised learning methods, K-means and PCA, are utilized to learn the spatial feature bases in each decorrelated spectral band. Then spatial-spectral features are generated by concatenating the spatial feature representations in all/principal spectral bands. In the second work for HSI classification, we propose to stack the spectral patches to reduce the spectral dimensionality and generate 2-D spectral quilts. Such quilts retain all the spectral information and can result in less convolutional parameters in neural networks. Two light convolutional neural networks are then designed to classify the spectral quilts. As the third work for HSI classification, we propose a combinational fully convolutional network. The network can not only take advantage of the inherent computational efficiency of convolution at prediction time, but also perform as a collection of many paths and has an ensemble-like behavior which guarantees the robust performance.

Keywords: autonomous techniques, planetary exploration, rock image classification, unsupervised feature learning, self-taught learning, hyperspectral image classification, spatial-spectral features, support vector machine (SVM), spectral quilts, convolutional neural network (CNN), combinational fully convolutional network (CFCN)

Co-Authorship Statement

Chapters 3 through 6 are co-authored with two supervisors.

Kenneth McIsaac is the student's primary research supervisor in the graduate program in Electrical and Computer Engineering. He provided guidance throughout the research program, including in the selection and formulation of the problems, and selection and evaluation of techniques to address them.

Gordon R. Osinski is the student's co-supervisor providing guidance especially in planetary geology. He provided insight and instruction in the practice of terrestrial and planetary field geology.

Acknowlegements

The thesis would not have been possible without the support of many people. I would like to express my sincere gratitude to my supervisors, Dr. Kenneth McIsaac and Dr. Gordon Osinski, for their continuous support, encouragement and advice. They have provided helpful and close guidance throughout the research program and the guidance has consistently been helpful, timely, and respectful.

Thanks as well to the team members of the research group for their kindness, valuable advice and enlightening discussion. Special thanks to Matthew Cross and Duane Jacques for sharing their thesis writing skills which are extremely helpful to me.

I would like also to acknowledge the funding from China Scholarship Council without which I would not have afforded the study as an international student. Finally, I would like to thank my parents for a lifetime of uncompromising support.

Contents

List of Figures

List of Tables

List of Abbreviations

AA	Average accuracy
AEGIS	Autonomous exploration for gathering increased science
AP	Attribute profile
ASTER	Advanced spaceborne thermal emission and reflection radiometer
CFCN	Combinational fully convolutional network
CK	Composite Kernel
CNN	Convolutional neural network
CRISM	Compact Reconnaissance Imaging Spectrometer for Mars
DBN	Deep belief network
EMAP	Extended multi-attribute profile
EMP	Extended morphological profile
ESA	European Space Agency
FCN	Fully convolutional network
GLCM	Gray level co-occurrence matrix
HSI	Hyperspectral imaging
κ	Kappa coefficient
MER	Mars exploration rovers
MP	Morphological profile
MRF	Markov random field
MRO	Mars Reconnaissance Orbiter
MSI	Multispectral imaging
MSL	Mars science laboratory
NASA	National Aeronautics and Space Administration
OA	Overall accuracy
OASIS	Onboard autonomous science investigation system
OMEGA	Observatoire pour la Minéralogie, l'Eau, les Glaces et l'Activité
PCA	Principal component analysis
ROSIS	Reflective Optics System Imaging Spectrometer
RRSC	Russian Roscosmos State Corporation
SSP	Stacking spectral patches
SVM	Support vector machine

Chapter 1

Introduction

The present work develops new approaches to autonomous science in the context of planetary exploration. Such missions have produced tremendous amounts of data from the instruments they carry, requiring autonomous techniques to enhance the efficiency of scientific data analysis and enlarge the scientific returns. We address two specific problems, rock image classification and hyperspectral image classification and produce new techniques to automatically model the feature representation of images and conduct the classification.

1.1 Planetary exploration

The last two decades have seen unprecedented achievements in space missions on Earth and extraterrestrial planets and we expect the scope of exploration to increase as it is powered by human's infinite curiosity. Take surface missions on Mars as examples. Mars Exploration Rovers (MER) [1] and Mars Science Laboratory (MSL) [2] are two of the most successful surface missions on Mars by NASA. MER mission (started in 2003) has two Mars rovers, Spirit and Opportunity, exploring Martian surface and searching for and characterizing a wide range of rocks and soils that hold clues to past water activity on Mars. MSL mission (began in 2012) has a car-sized rover, Curiosity, investigating the Martian climate and geology, as well as assessing the environmental conditions of the selected field site for microbial life in preparation for human exploration. Figure 1.1 illustrates the artist's conception of both MER rovers and MSL rover. In the near future, Mars2020 [3] from NASA and ExoMars [4] from ESA/RRSC are two Mars rover missions which will be launched both in 2020. Mars2020 is intended to investigate an astrobiologically relevant ancient environment on Mars and the surface geological processes and history. ExoMars rover is to search for the existence of past life on Mars.

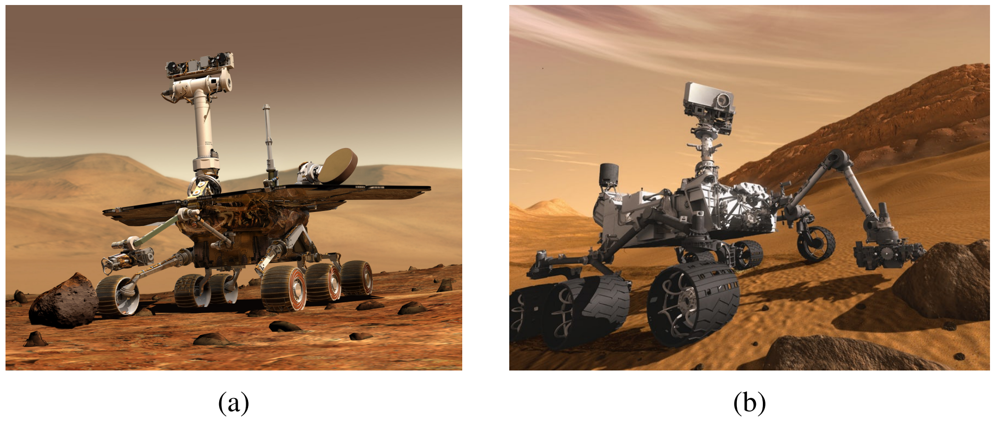
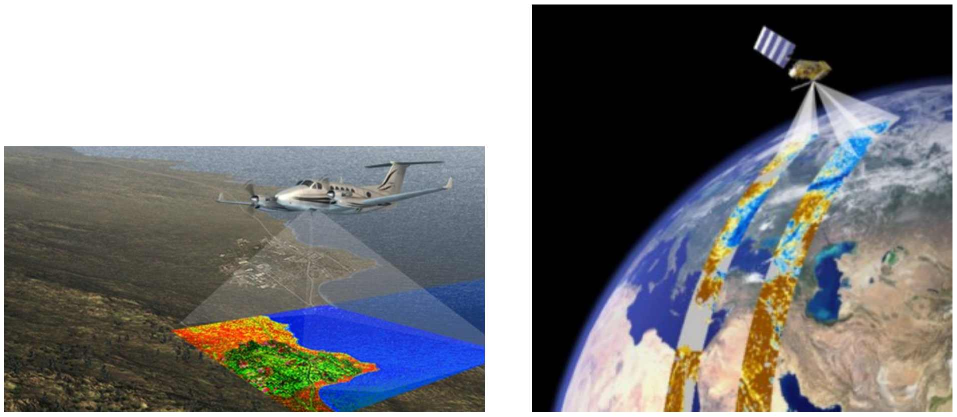


Figure 1.1: Artist's conception of (a) MER rovers (Spirit and Opportunity) and (b) MSL rover (Curiosity). Photo courtesy of NASA/JPL.

In addition to the surface missions which closely explore the planet, there have been several successful orbital missions which utilize imaging spectrometers to remotely observe the surface/atmosphere of Earth/Mars without physical contact (this is also called remote sensing). The spectrometers record the electromagnetic radiation of the materials which can be used to identify and map the materials. As indicated in Figure 1.2, there are mainly two types of remote sensing, airborne and spaceborne remote sensing. Following are several successful remote sensors/imaging systems.

Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) [5] is an airborne hyperspectral sensor in the realm of Earth remote sensing. It scans the Earth's surface and produces calibrated images of the upwelling spectral radiance in 224 contiguous spectral bands with wavelength range from 0.4 to 2.5 μm. The main objective of the AVIRIS project is to identify, measure, and monitor constituents of the Earth's surface and atmosphere based on molecular absorption and particle scattering signatures. Research with AVIRIS data is predominantly focused on understanding processes related to the global environment and climate change. Compact Recon-



(a) (b)

Figure 1.2: (a) Airborne remote sensing (b) Spaceborne remote sensing.

naissance Imaging Spectrometer for Mars (CRISM) [6] aboard NASA's Mars Reconnaissance Orbiter (MRO, launched by NASA in 2005) is a hyperspectral imaging system, operating from 3.62 to 3.92 μm wavelength with 6.55 nm per channel spacing. The best spatial resolution mapped onto Martian surface is ~12 m per pixel using along-track oversampled observations. CRISM is used to map both the surface and atmosphere of Mars with visible and infrared spectrometers and has played a significant role in the Mars expeditions, including the confirmation of the former presence of water on Mars. Observatoire pour la Minéralogie, l'Eau, les Glaces et l'Activité (OMEGA) [7] aboard Mars Express spacecraft (launched by ESA in 2003) is a visible and infrared mineralogical mapping spectrometer which provides hyperspectral images of Mars, with a spatial resolution from 300 m to 4 km, on 256 spectral channels in the near infrared range (1.0 – 5.2 μm) and 128 channels in the visible range (0.5 – 1.0 μm). The visible channel will measure the wavelength of incoming radiation to within ~7 nm and the infrared channel to within 13 – 20 nm. With the fact that different materials absorb and radiate light at different wavelengths, OMEGA builds up a map of Martian surface composition by analysing sunlight that has been absorbed and re-emitted by the surface. As radiation travelling from the surface to the instrument must pass through the atmosphere, OMEGA also detects wavelengths absorbed by some atmospheric constituents, in particular dust and aerosols.

A large amount of achievements of planetary exploration is due to the advancement in imaging and observing techniques, which has allowed new insights into the nature of Earth and extraterrestrial planets such as Mars. Instruments aboard aircrafts, satellites and surface rovers have been gathering a huge amount of data from various sensor modalities, such as magnetism, radar, laser altimetry, spectrum etc. The development and deployment of the exploring platforms will enhance the collection of scientific data. As the exploration missions continue to expand in scope, the growing volume of recorded data gradually requires more efficient and effective techniques for data analysis.

Among these scientific data, visual information provided by images in visible and other adjacent bands is particularly significant. Landforms/materials can be identified and interpreted from images [8]. Images stand in for the eyes of human specialists, acting as an initial and primary reference for the environment in the viewing field. Over many years, mission tasks have required interpretations of the image data from visible-wavelength as well as other electromagnetic spectra. Images from the visible spectrum are the key tools for navigation of mobile rovers on planetary surface. Rovers are guided towards regions of interest which are identified by the scientific interpretation of these visible images. The route of safe planning towards the targets also relies on the visible images. Images from other electromagnetic spectra (multispectral/hyperspectral) can map the distribution of materials/minerals, monitor the environmental conditions and distinguish the types of land covering. A regional or global geological map can thereafter be generated. Thus, both operational and scientific use of the images form fundamental elements of planetary mission operations.

1.2 Autonomous science

The growing volume of scientific data requires efficient and effective techniques for data analysis. Furthermore, the exploration missions on distant planets such as Mars suffer the limitation of communicating bandwidth and large delay of data transmission. To address these challenges, a possible strategy is to incorporate artificial intelligence to improve the autonomy on the process of data analysis. The enhanced autonomy will dramatically increase the scientific returns and reduce the intervention from human scientists. The high artificial intelligence will enable the planetary rovers to autonomously identify and react to serendipitous science opportunities. More concretely, with advanced computer vision and machine learning techniques, the rovers can automatically locate rocks, analyze rock properties, and identify rocks that merit further investigation. Also, by interpreting the scientific data in situ, the rovers will be capable to make decisions by themselves about which new data to gather, which instruments to use, and where to go next.

The autonomous techniques will benefit not only the distant missions on Mars, but also the orbital missions on Earth. Instruments aboard the orbital spacecrafts collect observing data and send for human analysis. The automated or computer-aided interpretation could speed up the work of human scientists. The gain can come from three aspects. The first is reliability. While humans can make mistakes, the computers can provide more reliable investigation on data and possibly see things that humans overlook. The second is efficiency. The growing volume of scientific data almost makes it impossible for humans to manually investigate and analyze the data. However, autonomous techniques can take advantage of high-performance computers and dramatically improve the efficiency. The last but not least is capability. With the autonomous techniques, the computers are capable to dig into the huge amount of data and figure out the hidden patterns, such as learning flexible feature representation for images.

1.3 Research problems

The autonomous interpretation of planetary images can enhance the performance of planetary exploration missions by reducing the data volume needed for specific observations, and the task load of scientists analyzing and interpreting image data [9]. Images of a planetary surface, either in-situ photographed or remotely sensed, provide rich and useful information about that planetary surface and enable the understanding and investigation of the nature and history of a geological setting by recognition, classification, and mapping of different types of surface materials.

The present work addresses two primary problems representative of computer vision tasks needed for autonomous science in the study of planetary surfaces – autonomous rock image classification and hyperspectral image classification. Rock image classification is to identify the specific type of rocks from the in-situ photographs. It allows the understanding of the environment in which the rocks were created and its subsequent geological history. Hyperspectral image classification attempts to recognize and map different types of surface materials from the spaceborne or airborne hyperspectral image. It enables the detection and investigation without any physical contact.

1.3.1 Rock image classification

Rock image classification refers to attempts to identify the specific type of rocks based on the visual appearance. The rock images are photographed in-situ by robotic platforms on surface missions. More autonomy on the process of detection and identification can dramatically enhance the exploration performance and the scientific returns by reducing the data volume needed for specific observation and human scientists' intervention. The autonomy for extraterrestrial exploration is especially significant because of the communicating bandwidth limit and large delay of data transmission. An autonomous geological classifier, even one which works only for specific rock types or specific environments, would be a very valuable tool for increasing the autonomy and scientific discovery rate of planetary exploration missions.

The identification of rock type is important since rocks provide information as to the environment in which they were created and the subsequent geological history [10]. For example, the size of crystals in igneous rocks can be used to estimate cooling rates and provides constraints on the depth of formation; the grain size and shape of sedimentary rocks provides information as to the mode of deposition; and the properties of rocks formed by meteorite impact craters reflects the pressure and temperature of formation and of the environment prior to impact. Thus, autonomous rock classification has the potential to provide valuable information about the origin and evolution of rocky planetary bodies throughout the Solar System.

Rock image classification consists of extracting a feature representation for rock images and conducting classification. The feature representation plays a significantly important role in the performance of classification. However, rock appearance is seldom homogeneous which makes the design of the feature representation challenging. Conventional feature modeling methods utilize hand-engineered features manually/semi-automatically selected for the specific applica-

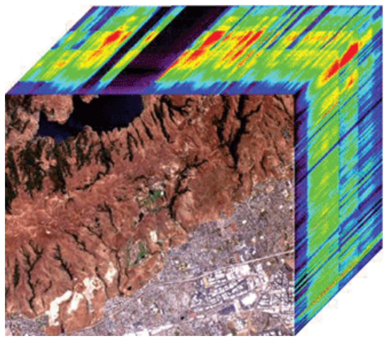


Figure 1.3: A hyperspectral image cube. This image was acquired in 1997 over Moffett Field in California by the Airborne Visible-Infrared Imaging Spectrometer (AVIRIS), developed by NASA JPL Laboratory. Photo courtesy of NASA.

tions. These features are not flexible and are usually time-consuming to choose. Thus, they are not good enough to represent inhomogeneous rock images. We will propose an unsupervised feature learning method in Chapter 3, which can automatically learn the feature representation for rock images and experiments show that our approach is more flexible and powerful than hand-crafted feature sets.

The history and current state of the literature with respect to techniques of rock image classification is given in section 2.1.

1.3.2 Hyperspectral image classification

A typical hyperspectral image has hundreds of spectral bands with fine spectral resolution (see Figure 1.3 for an example). Each pixel of the image contains spectral information (absorption, reflectance and emission of electromagnetic spectrum), which is added as a third dimension of intensity to the two-dimensional spatial image, generating a three-dimensional image cube. Since certain objects leave unique spectral signatures in the electromagnetic spectrum, these spectral signatures in hyperspectral images enable the identification of materials/objects. While the spectral information identifies the materials in the scene, spatial information provides locations.

Hyperspectral images (airborne or spaceborne) can cover an enormous area of planetary surface with rich spectral and spatial information. Over the last decades, many efforts have been directed toward geological mapping and land cover classification with hyperspectral images [11]. Conventional methods for mapping require tremendous human intervention, such as manually selecting spectral components for analysis and defining band ratios, and intense geological knowledge from specialists. Modern methods apply machine learning techniques to improve the efficiency. As with rock image classification, the feature representation is dramatically important for hyperspectral image classification. However, previous research work on designing feature representation either relies on heavily hand-crafted features, or is computationally expensive. We will propose three innovative methods in Chapter 4, 5 and 6 respectively for hyperspectral image classification, each of which inherently has its own advantage over the previous work.

The history and current state of hyperspectral image classification is described in section 2.2.

1.4 Research contribution

The work described in the subsequent chapters details several contributions. In particular, these include:

- A novel technique to autonomously learn feature representations for rock images which is more flexible and expressive than normal manual feature methods.
- A new technique to model spectral and spatial contextual information for hyperspectral images and enable high performance of hyperspectral image classification.
- A novel strategy to stack the spectral patches to form spectral quilts which represent the spectral volume in form of 1-channel grayscale image. The spectral quilts construct new feature patterns which could be useful to distinguish different materials.
- Two shallow convolutional neural networks which perform well with spectral quilts.
- A combinational fully convolutional network which has an ensemble-like behavior and is extremely efficient in predicting hyperspectral images in terms of computation and memory cost.

Bibliography

[1] Mitchell Ai-Chang, John Bresina, Len Charest, Adam Chase, JC-J Hsu, Ari Jonsson, Bob Kanefsky, Paul Morris, Kanna Rajan, Jeffrey Yglesias, et al. Mapgen: mixed-initiative planning and scheduling for the mars exploration rover mission. *IEEE Intelligent Systems*, 19(1):8–12, 2004.

[2] John P Grotzinger, Joy Crisp, Ashwin R Vasavada, Robert C Anderson, Charles J Baker, Robert Barry, David F Blake, Pamela Conrad, Kenneth S Edgett, Bobak Ferdowski, et al. Mars science laboratory mission and science investigation. *Space science reviews*, 170(1-4):5–56, 2012.

[3] Mars 2020 mission overview. `https://mars.jpl.nasa.gov/mars2020/mission/overview/`. Accessed: 2018-03-19.

[4] J Vago, O Witasse, H Svedhem, P Baglioni, A Haldemann, G Gianfiglio, T Blancquaert, D McCoy, and R de Groot. Esa exomars program: the next step in exploring mars. *Solar System Research*, 49(7):518–528, 2015.

[5] Robert O Green, Michael L Eastwood, Charles M Sarture, Thomas G Chrien, Mikael Aronsson, Bruce J Chippendale, Jessica A Faust, Betina E Pavri, Christopher J Chovit, Manuel Solis, et al. Imaging spectroscopy and the airborne visible/infrared imaging spectrometer (aviris). *Remote Sensing of Environment*, 65(3):227–248, 1998.

[6] Scott Murchie, R Arvidson, Peter Bedini, K Beisser, J-P Bibring, J Bishop, J Boldt, P Cavender, T Choo, RT Clancy, et al. Compact reconnaissance imaging spectrometer

for mars (crism) on mars reconnaissance orbiter (mro). *Journal of Geophysical Research: Planets*, 112(E5), 2007.

[7] J-P Bibring, A Soufflot, M Berthé, Y Langevin, B Gondet, P Drossart, M Bouyé, M Combes, P Puget, A Semery, et al. Omega: Observatoire pour la minéralogie, l'eau, les glaces et l'activité. In *Mars Express: the scientific payload*, volume 1240, pages 37–49, 2004.

[8] Ronald Greeley. *Introduction to planetary geomorphology*. Cambridge University Press, 2013.

[9] Raymond Francis, Kenneth McIsaac, David R Thompson, and Gordon R Osinski. Autonomous rock outcrop segmentation as a tool for science and exploration tasks in surface operations. In *SpaceOps 2014 Conference*, page 1798, 2014.

[10] V Gor, R Castano, R Manduchi, RC Anderson, and E Mjolsness. Autonomous rock detection for mars terrain. *Space*, pages 1–14, 2001.

[11] JR Harris, L Wickert, T Lynds, P Behnia, R Rainbird, E Grunsky, R McGregor, and E Schetselaar. Remote predictive mapping 3. optical remote sensing–a review for remote predictive geological mapping in northern canada. *Geoscience Canada*, 38(2), 2011.

Chapter 2

Literature Review and Background

2.1 Rock image classification

2.1.1 Background

Autonomous geological detection is becoming an increasingly important technique for robotic platforms exploring remote environments such as Mars (e.g. [1, 2]). It can maximize the scientific return and reduce the need for human involvement. In the case of Mars specifically, the bandwidth limit and large time delay (3 to 22 minutes one-way travel time) of data transmission make autonomous techniques even more critical and valuable. The past two decades have seen tremendous achievements in Mars exploration. Among them are Mars Exploration Rovers (MER) and Mars Science Laboratory (MSL) missions. Both missions sent rovers to the surface of Mars and explored their respective regions of interest with various scientific instruments. Two autonomous onboard systems have been developed for these rovers: the Onboard Autonomous Science Investigation System (OASIS) [3, 4, 5], and the Autonomous Exploration for Gathering Increased Science (AEGIS) system [6, 7]. Both systems are actively used and have enabled the rovers to autonomously identify and react to serendipitous science opportunities by analyzing imagery onboard with computer vision techniques. Tasks included locating rocks in the images, analyzing rock properties, and identifying rocks that merit further investigation through autonomous selection and sequencing of targeted observations. However, the rovers still heavily rely on explicit instructions given by scientists on Earth, which requires extensive communication and frequent command cycles. Therefore, there is still a long way to go before rovers will possess sufficient "intelligence" to reason about science goals, make informed decisions, and respond to discoveries autonomously [2].

An alternative approach to AEGIS and OASIS is increasingly being used in geosciences in the form of computer vision. For example, Chanou et al. [8] and Pittarello et al. [9] developed and applied quantitative image analysis methods to analyze the images of individual rock samples. In these approaches, components or particles of a rock image are first segmented, which then allows the measurement and quantification of various properties, such as shape complexity, preferred orientation, size-frequency, and so on. A different advanced technique that we focus on here is rock image classification [10]. Instead of the exact quantitative measurement of particles in rock images, the approach of rock image classification is to identify the specific type of rock(s) based on visual appearance. The identification of rock type is important as this provides information as to the environment in which the rock was created and its subsequent geological history [11].

2.1.2 Related work

A typical framework of image classification (see Figure 2.1) includes extracting feature representation for input images and feeding the feature representation into a classifier. In general, the performance of image classifiers is heavily dependent on the selection of a feature representation. Unfortunately, rock textures are seldom homogeneous. As a result, the design of a feature representation is difficult, which makes rock image classification extremely challenging. There have been a few attempts at developing feature representation for rock image classification to date. All these previous works use either hand-engineered features manually selected for the specific application, or features selected with time-consuming methods.

Prior works mostly involve manually selected features. In order to reduce the time-consuming process of manual identification of rock samples, Slipek et al. [12] and Mlynarczuket al. [13]

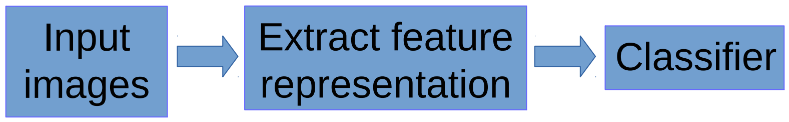


Figure 2.1: The typical framework of image classification.

conducted autonomous classification of microscopic images of rocks by four pattern recognition methods - nearest neighbour, k-nearest neighbours (k-NN), nearest mode, and optimal spherical neighbourhoods. Sharif et al. [14] built a small library of grayscale images from a total of 30 hand samples, and used Bayesian analysis to classify them with selected Haralick textural features [15]. In order to distinguish adjacent outcrops, Francis et al. [1] started with some fundamental visual “channels” such as colour and difference between colour channels, then utilized multi-class linear discriminant analysis (MDA) to identify the principal visual components. Harinie et al. [16] utilized Tamura features [17] to classify hand samples of rocks into the three major categories, namely, igneous, sedimentary and metamorphic. Dunlop et al. [18] studied features such as shape, albedo, colour and textures, then conducted rock classification with different feature combinations. Singh et al. [19] compared 7 well-established image texture analysis algorithms for rocks classification and the results suggested that Law’s masks [20] and co-occurrence matrices [15] were best. Lepisto et al. [21] classified rock images by methods based on textural and spectral features. The spectral features are some colour parameters and the textural features are calculated from the co-occurrence matrix. In order to improve the classification accuracy, Lepisto et al. [22] combined colour information in Gabor space [23] to the texture description. Given that various visual descriptors extracted from images are often high dimensional and non-homogenous, Lepisto et al. [24] conducted rock images classification based on k-nearest neighbour voting, which combined k-NN base classifiers for different descriptors by voting. A similar idea of combining base classifiers was presented by Lepisto et al. [25]. Each feature descriptor had a corresponding separate base classifier, and better classification accuracy can be achieved by combining opinions provided by each base classifier.

Other works have concentrated on feature selection. Chatterjee et al. [26] used the genetic algorithm to select features, and then classified limestone with multi-class support vector machine (SVM). Shang et al. [10] utilized a reliability-based method and mutual information to select features, then classified rocks images in a more general dataset. Both works showed that their own feature selection methods worked well in their dataset, but feature selection itself is time-consuming. When the dataset becomes complicated, one might have to think of what kind of feature pool to select from, or even devise a brand new feature representation.

2.2 Hyperspectral image classification

2.2.1 Background

Remote sensing can be broadly defined as the collection and interpretation of information about an object, area, or event without being in physical contact with the object [27]. Aircrafts and satellites are the common platforms for remote sensing. Though imaging in the visible portion of the electromagnetic wavelength was the original form of remote sensing, technological development has enabled the acquisition of information at other wavelengths including near infrared, thermal infrared and microwave. The measurement of the electromagnetic radiation takes place in spectral bands which are defined as discrete intervals of the electromagnetic spectrum. For example, the wavelength range of 0.5 μm to 0.6 μm is one spectral band. Based on different measurements of spectral information, remote sensing can be divided into multispectral imaging (MSI) and hyperspectral imaging (HSI).

Both MSI and HSI utilize optical spectroscopy as an analytical tool. Each pixel of the image contains spectral information, which is added as a third dimension of intensity to the two-dimensional spatial image, generating a three-dimensional image cube. Unlike the common RGB color image, where each pixel has red, green and blue color, a typical multi/hyperspectral image usually has many more spectral bands. The spectral range can extend beyond the visible range. Each pixel contains absorption and reflectance electromagnetic spectrum. Since certain objects leave unique spectral signatures in the electromagnetic spectrum, these spectral signatures enable the identification of materials/objects. Note that there is difference between MSI and HSI. MSI contains spectral information at a smalll set of discrete and narrow spectral bands, while HSI records spectra over a continuous spectral range and the sampling wavelengths are equally distributed with fine resolution. Landsat [28] and ASTER [29] are two excellent examples of multispectral imaging, AVIRIS [30] is an excellent example of hyperspectral imaging.

Given that HSI usually acquires digital images in many continuous and very narrow spectral bands, it enables the construction of an essentially continuous radiance spectrum for every pixel in the scene. Thus, HSI makes possible the remote identification of materials of interest based on their spectral signature [31]. HSI has been widely used in areas such as planetary science, agriculture, land use classification etc. For example, Combe et al. [32] analyzed the mineralogical composition of the Martian surface with the hyperspectral data from OMEGA mineralogical mapping spectrometer onboard Mars Express. Moussaoui et al. [33] studied the chemical species on surface and the atmosphere of Mars with hyperspectral images also from OMEGA spectrometer. Since hyperspectral remote sensing provides valuable information about vegetation type, leaf area index, biomass, chlorophyll, and leaf nutrient concentration, it can be used to understand ecosystem functions, vegetation growth, and nutrient cycling [34] and estimate the biochemical and biophysical parameters of wetland vegetation [35]. HSI can also be applied to urban area classification [36] and automatic target detection [37].

2.2.2 Related work

Remotely sensed hyperspectral images, either spaceborne or airborne, can cover relatively large areas of Earth's surface with rich spectral information and enable the accurate and robust classification of land cover [38]. Detailed spectral information is helpful to discriminate materials of interest [39]. Over the last decades, many efforts have been directed toward using machine learning techniques to automatically classify hyperspectral imagery.

As with rock image classification, the feature representation for hyperspectral image is also significant. Because of the existence of hundreds of spectral bands, the feature representation techniques for hyperspectral images are different from the ones for rock images. Conventional pixel-wise classification methods process each pixel independently without exploiting the spatial information [40]. However, spatially adjacent pixels usually share similar spectral characteristics and combing spectral information with spatial contextual information can reduce the uncertainty of the samples. As a result, the spatial contextual information is as important as the spectral information [41].

There have been significant works on jointly combining spatial and spectral information to analyze spectral imageries [42]. Early works, such as [43], [44] and [45] have sought to model spatial contextual information to classify the multispectral imageries. More recently, Coburn et al. [46] utilized the first and second order statistical textures to model the spatial information for forest stands. Pesaresi et al. [47] presented the concept of morphological profiles (MPs) to model the spatial information which has proven to be effective. MPs are constructed by applying a set of mathematical morphological operations (i.e., opening and closing) [48, 49] on spectral images. They simultaneously attenuate some spatial details while preserving the geometric characteristics of the other regions. Based on MP, the derivative of the morphological profile (DMP) was also proposed in [47]. A DMP is useful for visual inspection of the scene since it shows the differences between adjacent levels of the MP. Benediktsson et al. [36] and Fauvel et al. [50] later proposed extended morphological profiles (EMPs) which use MPs to model the spatial information in the top principal spectral bands after reducing spectral dimensionality.

As an extension of MPs, attribute profiles (APs)were proposed in [51]. APs provide a multilevel characterization of an image by using the sequential application of a morphological attribute filter (AF). AFs are connected morphological operators that process an image by considering the connected components at different levels in the image. APs model the spatial information more precisely than MPs, since the input images can be processed according to many attributes, which can be defined with great flexibility. Dalla et al. [52] proposed the extended attribute profile (EAP) and the extended multi-attribute profile (EMAP), which rely on the application of the APs to hyperspectral data and to a straightforward further extension to a multi-attribute scenario, respectively.

In addition to using morphological operations to generate spatial-spectral features, composite kernels (CKs) [53, 54, 55] were applied to combine spatial information and spectral information. In the CKs method, a local spatial feature extraction method is used to extract spatial features (e.g., mean or standard deviation of the pixel intensities in the spatial neighbourhood of the pixels), then the extracted spatial features and original spectral features are used to compute the spatial and spectral kernels to form a CK. With this CK containing both spatial and spectral information, classifiers such as support vector machine (SVM) [53], multinomial logistic regression [54] or extreme learning machine [55] can be used to conduct classification.

Rather than directly generating spatial-spectral features, Markov Random Field (MRF) based methods first conduct pixel-wise classification, then integrate spatial contextual information to refine (smooth) the classification [56, 57, 58]. Based on the assumption that the neighboring pixels have the same class labels as the central pixel, the maximum a posteriori decision rule is typically formulated as the minimization of a suitable energy function. The pixel-wise classification can be conducted with probabilistic classifiers, such as probabilistic SVM [56, 59], subspace Multinomial Logistic Regression [60], or sparse Multinomial Logistic Regression [61] etc. In order to preserve small structures and edges, Tarabalka et al. [56] proposed to integrate the edge information into the spatial energy function with a gradient map, Zhang et al. [62] proposed an adaptive-MRF to adjust the weighting coefficient of the spatial energy and Sun et al. [61] also presented a weighted MRF.

Although MP-based methods are an effective technique for extracting spatial contextual information, they are heavily hand-crafted and are therefore less flexible for different datasets. CK-based methods are a strong competitor, but the spatial features extracted by CKs are usually so simple that they do not accurately represent complex spatial structures. MRF-based methods have shown great performance, but the performance heavily depends on the initial pixel-wise classification which still require manual feature selection. Thanks to the fast development of neural networks [63, 64, 65] in image classification in the past few years, a lot of researchers have been focusing on utilizing neural networks as an end-to-end method to automatically model the spatial-spectral feature representation for hyperspectral images.

Li et al. [66] first applied principal component analysis (PCA) to preserve the top 3 principal components of spectral bands, then extracted 7×7 neighbor regions over the 3 principal bands as the training samples. After flattening each sample as a column vector, a deep belief network

(DBN) was utilized for classification. Similarly, Chen et al. [67] stacked the column vector with the original spectrum vector and fed the stacked spectral-spatial vectors into DBN for classification. Instead of DBN, Makantasis et al. [68] utilized a convolutional neural network (CNN) to directly classify the cropped 5×5 neighbor regions after the spectral dimensionality reduction. Other than using the neural network as an end-to-end method to directly classify the samples, there are some attempts to extract the high-level features with neural networks and then conduct classification with other classifiers. For example, Zhao et al. [69] and Zhao et al. [70] applied a logistic regression classifier to classify the extracted features. Zhao et al. [69] utilized a multiscale convolutional neural network (MCNN) to extract multiscale spatial features, then fused spatial features with spectral features to obtain the spectral-spatial features. Zhao et al. [70] applied balanced local discriminant embedding (BLDE) algorithm to extract the spectral features with lower dimensionality and used a CNN framework to extract spatial-related deep features. Then the spectral-spatial features are obtained by simply stacking the BLDE-based spectral features with CNN-based spatial features. All the methods above first require compression of the hyperspectral image with dimensionality reduction methods such as PCA. The compression step, however, will inevitably cast away a certain amount of useful spectral information.

Instead of reducing the spectral dimensionality, Lee et al. [71, 72] proposed a contextual deep CNN that can jointly exploit spatial and spectral features directly from the original hyperspectral image. Given that usually there are hundreds of spectral bands for a hyperspectral image, such a neural network will end up with a relatively large amount of parameters (especially the early layers) and therefore the risk of overfitting is increased. Chen et al. [73] proposed a 3-D CNN model to extract spectral-spatial features. Basically, the convolution and pooling are conducted in both spatial and spectral dimensions rather than only in the spatial dimensions as in conventional approaches. This model can have fewer parameters to train but the computation is tremendously increased because of the convolution along spectral bands.

2.3 Machine learning

Machine learning is a method of data analysis that can automatically find hidden insights from data. Machine learning techniques are typically classified into two broad categories - unsupervised learning and supervised learning, depending on whether there is a learning feedback available to the learning system. In the following subsections, we discuss several techniques from both categories which we will utilize to solve rock image and hyperspectral image classification problems in this thesis.

2.3.1 K-means

K-means is by far the most widely used clustering algorithms. It aims to group data points into K clusters in which each data point belongs to the cluster with the nearest mean [74]. The main advantage of K-means is that it is fast and easily implemented at large scale. Other than its use in market segmentation, social network analysis etc., it is also identified as a successful method to learn feature representation for images by computer vision researchers [75].

The classic K-means algorithm finds cluster centroids that minimize the distance between data points and the nearest centroid. Given a set of data points $\{x_1, x_2, ..., x_n\}$, where each data point is a d-dimensional vector, K-means aims to partition n data points into K ($\leq n$) clusters so as to minimize the optimization objective:

$$J = \frac{1}{n} \sum_{i=1}^{n} ||x_i - \mu_{c^i}||^2 \tag{2.1}$$

where μ_{c^i} is the cluster centroid of the cluster to which data point x_i has been assigned to, $c^i \in \{1, 2, ..., K\}$ is the cluster number. The entire optimizing process is summarized as follows:

a. Randomly allocate K points as the cluster centroids;

b. Go through each data point and evaluate the distance between the point and every cluster centroid, then assign the point to the closest cluster;

c. Move each centroid to the average of the correspondingly assigned data points;

d. Repeat step b and c until convergence.

Note that K-means can converge to different solutions depending on the initialization of centroids. So there is a risk of local optimum. A typical solution is to do multiple random initializations and see if they lead to the same result, since many same results are likely to indicate a global optimum.

2.3.2 Principal component analysis

Principal component analysis (PCA) is a method for dimensionality reduction [76, 77]. When the data is redundant, PCA is useful to compress the data into lower dimensional space with a tiny reconstruction error. Data compression usually leads to efficient computation since the data volume is reduced. Besides, the compressed data is represented by new features which could be more abstract and expressive than the original features. Thus, PCA is also used to extract feature representation in some cases. For example, Turk et al. [78] applied PCA to extract feature bases (called EigenFaces) from human faces and then represented faces as the linear combination of the EigenFaces.

Assume $X = [x_1, x_2, ..., x_n]^T$ contains all the data points. $x_i \in \mathbb{R}^d$ is a d-dimensional vector. X is a $n \times d$ matrix. Each row of X corresponds to one point and each column of X corresponds to one feature. The full routine of PCA is summarized as follows:

a. Normalize the data points. $X := \frac{X - mean(X)}{\sqrt{var(X)}}$, where the average and standard deviation are conducted along the first dimension. This will scale the features into a comparable range and move the mean of data points to 0.

b. Compute the covariance matrix: $\Sigma = \frac{1}{n}X^T X$.

c. Calculate eigenvectors using singular value decomposition: $[U, S, V] = svd(\Sigma)$. Columns of U are eigenvectors.

d. Select the top P columns (in the case of extracting feature representation, these principal eigenvectors are the feature bases) in U and use them to project each point x_i to the lower dimensional sub-space $\mathbb{R}^P$:

$$x_i := U(:, 1 : P)^T * x_i \tag{2.2}$$

2.3.3 PCA-whitening

While PCA is to rotate and compress data points with principal components, PCA-whitening is to rotate and rescale the data to reduce correlations among features and assure that features all have the same variance [79]. PCA-whitening is a common method to preprocess the data points. The full routine of PCA-whitening is summarized here:

a. Center and standardize the data: $X := \frac{X - mean(X)}{\sqrt{var(X)}}$. X is an $n \times d$ matrix, where n is the number of data points and d is the number of features.

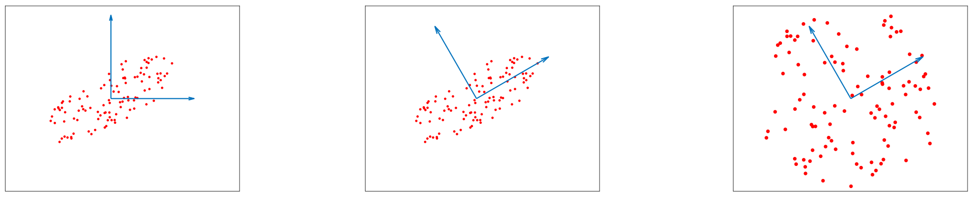
b. Compute the covariance matrix: $\Sigma = \frac{1}{n} X^T X$.

c. Calculate eigenvectors using singular value decomposition: $[U, S, V] = svd(\Sigma)$. Columns of U are eigenvectors.

d. Rotate the data with eigenvectors: $x_i := U^T * x_i$. This is to reduce the correlation between features.

e. Rescale each feature by $1/\sqrt{\lambda_i + \epsilon}$ to make it have unit variance, where λ_i is an eigenvalue of the covariance matrix. A small constant ϵ is added in case that the eigenvalues are numerically close to 0.

Fig. 2.2 illustrates the process and effect of PCA-whitening. We see that after rotating and scaling, both features are less correlated and have the same variance.



(a) (b) (c)

Figure 2.2: Process and effect of PCA-whitening. (a) unwhitened data, where the two features are clearly correlated with each other. (b) rotate the data to reduce the correlation. (c) whitened data, where features are less correlated and both have the same variance.

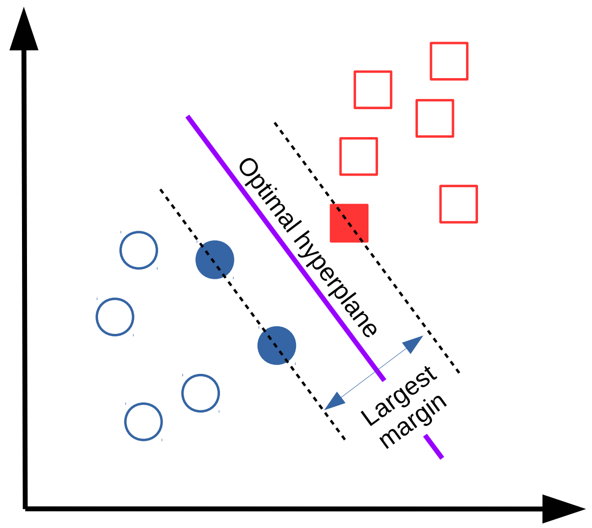


Figure 2.3: SVM learns a hyperplane with the largest margin.

2.3.4 Support vector machine

Support vector machine (SVM) [80, 81] is one of the most powerful and widely used classification algorithms. It learns a hyperplane or set of hyperplanes in high-dimensional space, which separate data points with the largest margin (see Figure 2.3). For a typical binary classification problem, assume the learned hyperplane is $w^T x + b = 0$, where x is a d-dimensional vector and each dimension corresponds to one feature of data points, w and b are the parameters defining the hyperplane. The distance (margin) from each data point to the hyperplane is

$$\frac{|w^T x + b|}{||w||} \tag{2.3}$$

Intuitively, a good classifier is the one having as large distance to every data point as possible, i.e., the closest points have as large distance as possible. So, the overall goal is to maximize the margin. Since the hyperplane and the closest data points will be determined for the set of data points as long as w (direction of hyperplane) is determined, w is the only unknown variable for the margin formula. For any hyperplane, we can always have $|w^T x + b| = 1$ for any data point by dividing a coefficient while having the hyperplane unchanged. So, the margin formula can be simplified as

$$\frac{1}{||w||} \tag{2.4}$$

with the constraints $w^T x + b \geq 1$ for positive data and $w^T x + b \leq -1$ for negative data (or $y_i(w^T x_i + b) \geq 1$ for each data point (x_i, y_i), where y_i is label). In this case, the closest distance from data points to hyperplane is exactly $\frac{1}{||w||}$. The maximization of margin can be convert to the minimization of a quadratic function,

$$\begin{aligned} &\min \frac{1}{2}||w||^2 \\ &s.t., y_i(w^T x_i + b) \geq 1, i = 1, ..., n \end{aligned} \tag{2.5}$$

To solve this optimization problem efficiently, we can add the constraints into the loss function with Lagrange duality. Then the loss function becomes

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{n} \alpha_i \left(y_i(w^T x_i + b) - 1 \right) \tag{2.6}$$

where $\alpha_i \geq 0, i = 1, ..., n$. Then the optimization function becomes

$$\min_{w,b} \max_{\alpha_i \geq 0} \mathcal{L}(w, b, \alpha) \tag{2.7}$$

Here, the optimization function is equivalent to the one switching the position of minimization and maximization, as shown in

$$\max_{\alpha_i \geq 0} \min_{w,b} \mathcal{L}(w, b, \alpha) \tag{2.8}$$

In order to first get the optimal solution for the minimization of $\mathcal{L}$, we set

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial w} &= 0 \Rightarrow w = \sum_{i=1}^{n} \alpha_i y_i x_i \\ \frac{\partial \mathcal{L}}{\partial b} &= 0 \Rightarrow \sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned} \tag{2.9}$$

Then we have

$$\begin{aligned} \mathcal{L}(w, b, \alpha) &= \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j - \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j b \sum_{i=1}^{n} \alpha_i y_i + \sum_{i=1}^{n} \alpha_i \\ &= \sum_{i=1}^{n} \alpha_i \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \end{aligned} \tag{2.10}$$

Now, our optimization problem is simplified as

$$\begin{aligned} &\max_{\alpha} \sum_{i=1}^{n} \alpha_i \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \\ &s.t., \alpha_i \geq 0, i = 1, \ldots, n \\ &\sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned} \tag{2.11}$$

where $\langle \cdot, \cdot \rangle$ indicates the inner product of two vectors. After optimizing the function to get the hyperplane, the inference is only to compute

$$\begin{aligned} f(x) &= \left(\sum_{i=1}^{n} \alpha_i y_i x_i \right)^T x + b \\ &= \sum_{i=1}^{n} \alpha_i y_i \langle x_i, x \rangle + b \end{aligned} \tag{2.12}$$

Since it is the closest points (vectors) who determine the hyperplane and the inference computation is only related to the closest points (because for all the other points their $y_i(w^T x_i + b) - 1 > 0$, so their corresponding α_i have to be 0 to maximize Formula 2.6), we call the closest points *support vectors*.

Above we have derived the optimization formula for linear SVM. Linear SVM only works for the data points which are linearly separable. For the data points which are not linearly separable, we can project the original data points to a higher dimensional space (assume $\phi(\cdot)$ is the projection). Similarly, the new optimization problem becomes

$$\begin{aligned} &\max_{\alpha} \sum_{i=1}^{n} \alpha_i \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \phi(x_i), \phi(x_j) \rangle \\ &s.t., \alpha_i \geq 0, i = 1, \ldots, n \\ &\sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned} \tag{2.13}$$

where the inner product $\langle \phi(x_i), \phi(x_j) \rangle$ can be noted as kernel function $\kappa(x_i, x_j)$. Compared with the inner product in the higher dimensional space, kernel function can be more efficiently computed in the lower dimensional space. Thus, kernel functions have been widely used to solve the nonlinear classification with SVM. With kernel function, the learned hyperplane (classifier) can be noted as

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i \kappa(x_i, x) + b \tag{2.14}$$

One of the most popular kernel functions is Gaussian (radial basis function) kernel, which projects the original data points to the infinite dimensional space. The Gaussian kernel is noted as

$$\kappa(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2} \right) \tag{2.15}$$

where σ is standard deviation. It is to control the shape of Gaussian kernel function. Large σ will make the high order feature components decay fast, which results in the projection to a approximately low dimensional space. On the other hand, small σ will allow projecting any data to be linearly separable but have a risk of overfitting.

In order to make the classifier less sensitive to outliers, we can add slack variables to the loss function to allow the data points to have the authority to move out of their own group a little bit. Take the linear SVM as example, the optimization problem can be denoted as

$$\begin{aligned}
&\min \frac{1}{2}\|w\|^2 + \frac{C}{2}\sum_{i=1}^{n}\xi_i^2 \\
&s.t.\ y_i(w^T x_i + b) \geq 1 - \xi_i, i = 1, ..., n \\
&\xi_i \geq 0, i = 1, ..., n
\end{aligned} \tag{2.16}$$

where ξ_i is the slack variable and C is the regularization parameter. C is to make a trade-off between training error and margin size. If C is large, training error will be small. Otherwise, training error and margin will be large. With the similar derivation indicated above, we can modify the linear SVM optimization problem as

$$\begin{aligned}
&\max_{\alpha} \sum_{i=1}^{n}\alpha_i \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j\langle x_i, x_j\rangle \\
&s.t., 0 \leq \alpha_i \leq C, i = 1, \ldots, n \\
&\sum_{i=1}^{n}\alpha_i y_i = 0
\end{aligned} \tag{2.17}$$

With careful choice of regularization parameter (e.g., C) and kernel parameter (e.g., σ in Gaussian kernel function), SVM is highly resistant to overfitting. Besides, it is effective in high dimensional spaces, even in cases where the number of dimensions is greater than the number of data points (e.g., hyperspectral image classification). Last but not least, it is also memory efficient since the decision function is only related to the support vectors.

2.3.5 Convolutional neural network

Convolutional neural networks (CNNs) have been fast developed to solve computer vision problems (e.g., image classification and object detection) in recent years. Among the earliest work, LeNet [82] is the most popular CNN which was designed for handwritten and machine-printed character recognition. Although LeNet had achieved great success almost two decades ago, it has seen a surge of recent interest due to the availability of a large amount of data through the digitization of society and the rapid development of high performance computing devices (e.g., GPU). AlexNet [63] is the first work which combined the innovation of CNN and efficient computation of GPU. It won the ILSVRC-2012 competition and performed way better than the second-best method. AlexNet opened a new era for CNNs. Since then, researchers have made great achievements in designing innovative network structures and developing the state-of-the-art techniques. For example, Simonyan et al. [65] developed VGG-net which simplified the neural networks structures and found that deeper networks can lead to better performance. In order to overcome the problem of vanishing gradient and speed up the training of deep network, ResNet [83] adds short cuts to the main path of the original convolutional network.

Conventional CNNs consist of alternating convolutional layers and pooling layers and fully connected layers in the end [84]. Figure 2.4 illustrates a typical framework of CNN.

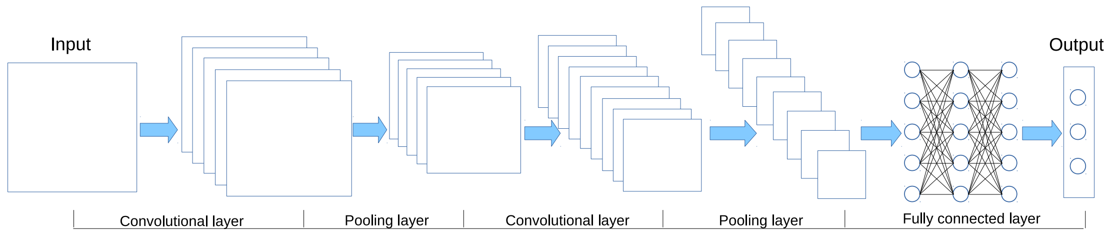


Figure 2.4: A typical framework of CNN.

2.3.5.1 Convolutional layer

Convolutional layers take the inner product of the linear filters and the underlying receptive fields followed by a nonlinear activation function at every local portion of the input. Concretely, during the forward pass, we slide and convolve each filter across the width and height of the input and compute dot products between the entries of the filter and the input at any position. As we slide the filter over the width and height of the input we will produce an activation map which gives the responses of that filter at every spatial position. Intuitively, the network will learn filters that activate when they see some type of visual patterns such as an edge of certain orientation. By stacking all the activation maps along the depth dimension, we will get the output volume for each convolutional layer. If the input is $n \times n$, convolution filter size is $f \times f$, padding is p and stride is s, then the size of output will be $[\frac{n+2p-f}{s}+1] \times [\frac{n+2p-f}{s}+1]$.

Technically, the convolution in neural networks is actually called cross-correlation, not mathematically true convolution which needs flipping the filters common in signal processing areas. Convolutions have two main advantages. The first is parameter sharing. A feature detector (filter) that is useful in one part of the image is probably useful in another part of the image. By sharing the parameters, convolutions can detect the shifted objects and make the network translation invariant. The second is the sparsity of connections. In each layer, each output value depends only on a small number of inputs from previous layer (within the receptive fields). This leads the network to remain fewer parameters. As a result, it can be trained with smaller training datasets and is less prone to overfitting.

The nonlinear activation function is necessary because it is more expressive than the linear one. With the nonlinear activation function, the overall network can end up to be complicate enough for complicate classification problem. Rectified linear unit (Relu) is the most common activation function to use. It allows the network to be trained fast since it is linear when entries are positive.

2.3.5.2 Pooling layer

Pooling layer is used to downsample the feature maps along the spatial dimensions (width and height) to reduce the amount of parameters and computation in the network, and hence to control overfitting. The commonly used pooling method is max-pooling, which retains the max values within the receptive fields. Because the main features usually correspond to the max values in feature maps, retaining max values will preserve the main features. In addition to max-pooling, average-pooling is sometimes used in very deep neural networks to largely collapse the feature map dimension.

2.3.5.3 Fully-connected layer

While convolutional layers extract feature representations and pooling layers downsample the feature maps, fully-connected layers work as the classifier. Each neuron in this layer will be connected to all the activations in the previous layer. The output activations can hence be computed with a matrix multiplication followed by a bias offset.

It is worth noting that the only difference between fully-connected layers and convolutional layers is that the neurons in convolutional layer are connected only to a local region in the input and many of them share parameters. However, the neurons in both layers still compute dot products, so their functional form is identical. Therefore, it is possible to convert a fully-connected layer to a convolutional layer. This conversion is particularly useful in practice since it allows to build a fully convolutional network which is computationally efficient in prediction.

2.3.5.4 Regularization methods

Neural networks usually have a large amount of model parameters and are prone to overfitting. There are several techniques to regularize the model and prevent overfitting during training.

Data augmentation means to augment the training data. In the case of computer vision problems, it is usually difficult to get enough data. The image data augmentation can be done by flipping, rotating or cropping images. Although data augmentation is not as good as collecting an additional set of brand new images because of the redundancy, it will still help generalize the model.

L1/L2-regularization is to penalize the values of the weights in the loss function based on the assumption that a model with small weights is simpler than a model with large weights (small weights tend to deactivate many neurons, which makes a simpler neural network). The added weight penalization term into the loss function can be written as Formula 2.18.

$$\begin{cases} \frac{\lambda}{2}\sum_i |w_i|, & \text{L1-regularization} \\ \frac{\lambda}{2}\sum_i w_i^2, & \text{L2-regularization} \end{cases} \tag{2.18}$$

w_i indicates any weight in the network and λ is the regularization parameter which is to control the degree of regularization. With such penalty terms, all the weights are driven to be small during training iteration. Although L1-regularization makes the weights sparse and needs less memory, L2-regularization is used more often in practice because it is differentiable so that the updating of weights with gradient descent will have a decay term making the weights smaller in each iteration step.

Dropout is to go through each of the layers of the network and eliminate (shut down or set to zero) each neuron with a probability at each training iteration. The dropped neurons do not contribute to the training in both forward and backward propagations of the iteration. This ends up with a smaller and diminished network in each iteration. Since different iteration dropouts/eliminates a different set of neurons, the network cannot rely on any one specific feature and have to spread out the weights. This will tend to have effect of shrinking the weights, which is similar to what L1/L2-regularization does.

Early stopping is to stop the training in half way. As the training iterates, the model might gradually overfit the data. By stopping the training in a proper position, we can prevent the overfitting. Unlike L1/L2-regularization, early stopping does not require tuning of the regularization parameter, which makes training computationally efficient. However, it mixes together two opposite things, optimizing loss function and preventing overfiting, and thus we cannot deal with these two things independently.

2.4 Summary

In this chapter, we first talked about the background and conventional methods for rock image classification and hyperspectral image classification. Then we talked about the machine learning techniques which will be utilized in the following chapters.

Bibliography

[1] R Francis, K McIsaac, DR Thompson, and GR Osinski. Autonomous mapping of outcrops using multiclass linear discriminant analysis. In *Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space, Montreal, Canada*, 2014.

[2] Raymond Francis, Kenneth McIsaac, David R Thompson, and Gordon R Osinski. Autonomous rock outcrop segmentation as a tool for science and exploration tasks in surface operations. In *SpaceOps 2014 Conference*, page 1798, 2014.

[3] Rebecca Castano, Michele Judd, Tara Estlin, Robert C Anderson, Lucas Scharenbroich, Lin Song, Daniel Gaines, Forest Fisher, Dominic Mazzoni, and Andres Castano. Autonomous onboard traverse science system. In *Aerospace Conference, 2004. Proceedings. 2004 IEEE*, volume 1. IEEE, 2004.

[4] Rebecca Castano, Tara Estlin, Robert C Anderson, Daniel M Gaines, Andres Castano, Benjamin Bornstein, Caroline Chouinard, and Michele Judd. Oasis: Onboard autonomous science investigation system for opportunistic rover science. *Journal of Field Robotics*, 24(5):379–397, 2007.

[5] Rebecca Castano, Tara Estlin, Dan Gaines, B Bomstein, Robert C Anderson, Brian Bue, and Michele Judd. Experiments in onboard rover traverse science. In *Aerospace Conference, 2008 IEEE*, pages 1–11. IEEE, 2008.

[6] Tara Estlin, Rebecca Castano, Benjamin Bornstein, Daniel Gaines, Robert C Anderson, Charles De Granville, David Thompson, Michael Burl, Michele Judd, and Steve Chien.

Automated targeting for the mer rovers. In *Space Mission Challenges for Information Technology, 2009. SMC-IT 2009. Third IEEE International Conference on*, pages 257–263. IEEE, 2009.

[7] Tara A Estlin, Benjamin J Bornstein, Daniel M Gaines, Robert C Anderson, David R Thompson, Michael Burl, Rebecca Castano, and Michele Judd. Aegis automated science targeting for the mer opportunity rover. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(3):50, 2012.

[8] a. Chanou, G. R. Osinski, and R. a F Grieve. A methodology for the semi-automatic digital image analysis of fragmental impactites. *Meteoritics & Planetary Science*, 49(4):621–635, apr 2014.

[9] Lidia Pittarello and Christian Koeberl. Clast size distribution and quantitative petrography of shocked and unshocked rocks from the el'gygytgyn impact structure. *Meteoritics & Planetary Science*, 48(7):1325–1338, 2013.

[10] Changjing Shang and Dave Barnes. Support vector machine-based classification of rock texture images aided by efficient feature selection. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*, pages 1–8. IEEE, 2012.

[11] V Gor, R Castano, R Manduchi, RC Anderson, and E Mjolsness. Autonomous rock detection for mars terrain. *Space*, pages 1–14, 2001.

[12] B Ślipek and M Młynarczuk. Application of pattern recognition methods to automatic identification of microscopic images of rocks registered under different polarization and lighting conditions. *Geology, Geophysics and Environment*, 39, 2013.

[13] Mariusz Młynarczuk, Andrzej Górszczyk, and Bartłomiej Ślipek. The application of pattern recognition in the automatic classification of microscopic rock images. *Computers & Geosciences*, 60:126–133, 2013.

[14] Helia Sharif, Maxim Ralchenko, Claire Samson, and Alex Ellery. Autonomous rock classification using bayesian image analysis for rover-based planetary exploration. *Computers & Geosciences*, 83:153–167, 2015.

[15] Robert M Haralick, Karthikeyan Shanmugam, and Its' Hak Dinstein. Textural features for image classification. *Systems, Man and Cybernetics, IEEE Transactions on*, (6):610–621, 1973.

[16] Thiagarajan Harinie, I Janani Chellam, SB Sathya Bama, S Raju, and V Abhaikumar. Classification of rock textures. In *Proceedings of the International Conference on Information Systems Design and Intelligent Applications 2012 (INDIA 2012) held in Visakhapatnam, India, January 2012*, pages 887–895. Springer, 2012.

[17] Hideyuki Tamura, Shunji Mori, and Takashi Yamawaki. Textural features corresponding to visual perception. *Systems, Man and Cybernetics, IEEE Transactions on*, 8(6):460–473, 1978.

[18] Heather Dunlop. *Automatic rock detection and classification in natural scenes*. PhD thesis, Carnegie Mellon University, 2006.

[19] Maneesha Singh, Akbar Javadi, and Sameer Singh. A comparison of texture teatures for the classification of rock images. In *Intelligent Data Engineering and Automated Learning–IDEAL 2004*, pages 179–184. Springer, 2004.

[20] Kenneth I Laws. Textured image segmentation. Technical report, DTIC Document, 1980.

[21] Leena Lepistö, Iivari Kunttu, Jorma Autio, and Ari Visa. Rock image classification using non-homogenous textures and spectral imaging. 2003.

[22] Leena Lepistö, Iivari Kunttu, and Ari Visa. Rock image classification using color features in gabor space. *Journal of Electronic Imaging*, 14(4):040503–040503, 2005.

[23] Jing Yi Tou, Yong Haur Tay, and Phooi Yee Lau. Gabor filters and grey-level co-occurrence matrices in texture classification. In *MMU International Symposium on Information and Communications Technologies*, pages 197–202, 2007.

[24] Leena Lepistö, I Kunttu, and Ari Visa. Rock image classification based on k-nearest neighbour voting. *IEE Proceedings-Vision, Image and Signal Processing*, 153(4):475–482, 2006.

[25] Leena Lepistö, Iivari Kunttu, and Ari Visa. Classification of natural rock images using classifier combinations. *Optical Engineering*, 45(9):097201–097201, 2006.

[26] Snehamoy Chatterjee. Vision-based rock-type classification of limestone using multi-class support vector machine. *Applied intelligence*, 39(1):14–27, 2013.

[27] George Joseph. *Fundamentals of remote sensing*. Universities press, 2005.

[28] Landsat missions. `https://landsat.usgs.gov`. Accessed: 2018-03-15.

[29] Advanced spaceborne thermal emission and reflection radiometer. `https://asterweb.jpl.nasa.gov`. Accessed: 2018-03-15.

[30] Airborne visible/infrared imaging spectrometer. `https://aviris.jpl.nasa.gov/`. Accessed: 2018-03-15.

[31] Dimitris Manolakis and Gary Shaw. Detection algorithms for hyperspectral imaging applications. *IEEE signal processing magazine*, 19(1):29–43, 2002.

[32] J-Ph Combe, S Le Mouélic, C Sotin, A Gendrin, JF Mustard, L Le Deit, P Launeau, J-P Bibring, B Gondet, Y Langevin, et al. Analysis of omega/mars express data hyperspectral data using a multiple-endmember linear spectral unmixing model (melsum): Methodology and first results. *Planetary and Space Science*, 56(7):951–975, 2008.

[33] Saïd Moussaoui, Hafrun Hauksdottir, Frédéric Schmidt, Christian Jutten, Jocelyn Chanussot, David Brie, Sylvain Douté, and Jon Atli Benediktsson. On the decomposition of mars hyperspectral data by ica and bayesian positive source separation. *Neurocomputing*, 71(10-12):2194–2208, 2008.

[34] Jungho Im and John R Jensen. Hyperspectral remote sensing of vegetation. *Geography Compass*, 2(6):1943–1961, 2008.

[35] Elhadi Adam, Onisimo Mutanga, and Denis Rugege. Multispectral and hyperspectral remote sensing for identification and mapping of wetland vegetation: a review. *Wetlands Ecology and Management*, 18(3):281–296, 2010.

[36] Jón Atli Benediktsson, Jón Aevar Palmason, and Johannes R Sveinsson. Classification of hyperspectral data from urban areas based on extended morphological profiles. *IEEE Transactions on Geoscience and Remote Sensing*, 43(3):480–491, 2005.

[37] Dimitris Manolakis, David Marden, and Gary A Shaw. Hyperspectral image processing for automatic target detection applications. *Lincoln laboratory journal*, 14(1):79–116, 2003.

[38] Philip H Swain. Fundamentals of pattern recognition in remote sensing. *Remote Sensing: The Quantitative Approach*, pages 136–187, 1978.

[39] Mathieu Fauvel, Yuliya Tarabalka, Jon Atli Benediktsson, Jocelyn Chanussot, and James C Tilton. Advances in spectral-spatial classification of hyperspectral images. *Proceedings of the IEEE*, 101(3):652–675, 2013.

[40] David A Landgrebe. *Signal theory methods in multispectral remote sensing*, volume 29. John Wiley & Sons, 2005.

[41] Pedram Ghamisi, Jon Atli Benediktsson, and Johannes R Sveinsson. Automatic spectral–spatial classification framework based on attribute profiles and supervised feature extraction. *IEEE Transactions on Geoscience and Remote Sensing*, 52(9):5771–5782, 2014.

[42] P Ghamisi, M Dalla Mura, and J A Benediktsson. A Survey on Spectra-Spatial Classification Techniques Based on Attribute Profiles. *IEEE Transactions on Geoscience and Remote Sensing*, 53(5):2335–2353, 2015.

[43] Robert M Haralick and K Sam Shanmugam. Combined spectral and spatial processing of erts imagery data. *Remote Sensing of Environment*, 3(1):3–13, 1974.

[44] David A Landgrebe. The development of a spectral-spatial classifier for earth observational data. *Pattern Recognition*, 12(3):165–175, 1980.

[45] Philip H Swain, Stephen B Vardeman, and James C Tilton. Contextual classification of multispectral image data. *Pattern Recognition*, 13(6):429–441, 1981.

[46] CA Coburn and Arthur CB Roberts. A multiscale texture analysis procedure for improved forest stand classification. *International Journal of Remote Sensing*, 25(20):4287–4308, 2004.

[47] Martino Pesaresi and Jon Atli Benediktsson. A new approach for the morphological segmentation of high-resolution satellite imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 39(2):309–320, 2001.

[48] Pierre Soille. *Morphological image analysis: principles and applications*. Berlin, Germany: Springer-Verlag, 2003.

[49] Pierre Soille and Martino Pesaresi. Advances in mathematical morphology applied to geoscience and remote sensing. *IEEE Transactions on Geoscience and Remote Sensing*, 40(9):2042–2055, 2002.

[50] Mathieu Fauvel, Jón Atli Benediktsson, Jocelyn Chanussot, and Johannes R Sveinsson. Spectral and spatial classification of hyperspectral data using svms and morphological profiles. *IEEE Transactions on Geoscience and Remote Sensing*, 46(11):3804–3814, 2008.

[51] Mauro Dalla Mura, Jón Atli Benediktsson, Björn Waske, and Lorenzo Bruzzone. Morphological attribute profiles for the analysis of very high resolution images. *IEEE Transactions on Geoscience and Remote Sensing*, 48(10):3747–3762, 2010.

[52] Mauro Dalla Mura, Jon Atli Benediktsson, Björn Waske, and Lorenzo Bruzzone. Extended profiles with morphological attribute filters for the analysis of hyperspectral data. *International Journal of Remote Sensing*, 31(22):5975–5991, 2010.

[53] Gustavo Camps-Valls, Luis Gomez-Chova, Jordi Muñoz-Marí, Joan Vila-Francés, and Javier Calpe-Maravilla. Composite kernels for hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, 3(1):93–97, 2006.

[54] Jun Li, Prashanth Reddy Marpu, Antonio Plaza, José M Bioucas-Dias, and Jon Atli Benediktsson. Generalized composite kernel framework for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 51(9):4816–4829, 2013.

[55] Yicong Zhou, Jiangtao Peng, and C. L Philip Chen. Extreme Learning Machine with Composite Kernels for Hyperspectral Image Classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(6):2351–2360, 2015.

[56] Yuliya Tarabalka, Mathieu Fauvel, Jocelyn Chanussot, and Jón Atli Benediktsson. Svm- and mrf-based method for accurate classification of hyperspectral images. *IEEE Geoscience and Remote Sensing Letters*, 7(4):736–740, 2010.

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

[58] Gabriele Moser and Sebastiano B Serpico. Combining support vector machines and markov random fields in an integrated framework for contextual image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 51(5):2734–2752, 2013.

[59] Xiangrong Zhang, Zeyu Gao, Licheng Jiao, and Huiyu Zhou. Multifeature hyperspectral image classification with local and nonlocal spatial information via markov random field in semantic space. *IEEE Transactions on Geoscience and Remote Sensing*, 2017.

[60] Jun Li, José M Bioucas-Dias, and Antonio Plaza. Spectral–spatial hyperspectral image segmentation using subspace multinomial logistic regression and markov random fields. *IEEE Transactions on Geoscience and Remote Sensing*, 50(3):809–823, 2012.

[61] Le Sun, Zebin Wu, Jianjun Liu, Liang Xiao, and Zhihui Wei. Supervised spectral–spatial hyperspectral image classification with weighted markov random fields. *IEEE Transactions on Geoscience and Remote Sensing*, 53(3):1490–1503, 2015.

[62] Bing Zhang, Shanshan Li, Xiuping Jia, Lianru Gao, and Man Peng. Adaptive markov random field approach for classification of hyperspectral imagery. *IEEE Geoscience and Remote Sensing Letters*, 8(5):973–977, 2011.

[63] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[64] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833. Springer, 2014.

[65] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[66] Tong Li, Junping Zhang, and Ye Zhang. Classification of hyperspectral image based on deep belief networks. *2014 IEEE International Conference on Image Processing (ICIP)*, pages 5132–5136, 2014.

[67] Yushi Chen, Xing Zhao, and Xiuping Jia. Spectral–spatial classification of hyperspectral data based on deep belief network. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(6):2381–2392, 2015.

[68] Konstantinos Makantasis, Konstantinos Karantzalos, Anastasios Doulamis, and Nikolaos Doulamis. Deep Supervised Learning for Hyperspectral Data Classification through Convolutional Neural Networks. *IGARSS 2015. 2015 IEEE International Geoscience and Remote Sensing Symposium. Proceedings*, pages 4959–4962, 2015.

[69] Wenzhi Zhao and Shihong Du. Learning multiscale and deep representations for classifying remotely sensed imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 113:155–165, 2016.

[70] Wenzhi Zhao and Shihong Du. Spectral Spatial Feature Extraction for Hyperspectral Image Classification : A Dimension Reduction and Deep Learning Approach. *IEEE Transactions on Geoscience and Remote Sensing*, 54(8):4544–4554, 2016.

[71] Hyungtae Lee and Heesung Kwon. Contextual Deep CNN Based Hyperspectral Classification. *arXiv*, 1604.03519:2–4, 2016.

[72] Hyungtae Lee and Heesung Kwon. Going deeper with contextual cnn for hyperspectral image classification. *IEEE Transactions on Image Processing*, 26(10):4843–4855, 2017.

[73] Yushi Chen, Hanlu Jiang, Chunyang Li, Xiuping Jia, and Pedram Ghamisi. Deep feature extraction and classification of hyperspectral images based on convolutional neural

networks. *IEEE Transactions on Geoscience and Remote Sensing*, 54(10):6232–6251, 2016.

[74] Tapas Kanungo, David M Mount, Nathan S Netanyahu, Christine D Piatko, Ruth Silverman, and Angela Y Wu. An efficient k-means clustering algorithm: Analysis and implementation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):881–892, 2002.

[75] Adam Coates and Andrew Y Ng. Learning feature representations with k-means. In *Neural Networks: Tricks of the Trade*, pages 561–580. Springer, 2012.

[76] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 2(1-3):37–52, 1987.

[77] Jonathon Shlens. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*, 2014.

[78] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[79] Patrik O. Hoyer Aapo Hyvrinen, Jarmo Hurri. *Natural Image Statistics*. Springer, 2009.

[80] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[81] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

[82] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[83] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[84] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

Chapter 3

Unsupervised Feature Learning for Autonomous Rock Image Classification

Published as Shu, L., McIsaac, & K., Osinski, G. R. in Computers & Geosciences, 2017(106), 10-17.

3.1 Introduction

Conventional feature methods for rock images consist either of an entirely manually crafted feature set or a set of features automatically selected from a set of manually crafted features. These manual features are not good enough to represent inhomogeneous rock images and are time-consuming to get. In this chapter, we present an unsupervised feature learning method for rock image classification. The proposed method can automatically learn the feature representations. The experimental results demonstrate that the learned feature representations have the potential to be more flexible and powerful.

We have approached the problem of feature selection for geological classification in two ways in this chapter. First, we propose an unsupervised feature learning technique [1] to extract features for rock images. The approach is to autonomously learn the feature representation from a large amount of data rather than manually choosing the features. This has the benefit of making the feature representation much more flexible when using different datasets. The feature learning method we utilized is based on K-means [2], which is fast and easily implemented. We applied this method to the classification of rock images with support vector machine (SVM).

The second autonomous feature selection method we propose in this chapter is called self-taught learning [3, 4]. The concept behind self-taught learning is to learn a feature representation from unlabelled images of *mixed-class* and then train a classifier on a *subset of the data* that has been labelled to identify certain subclasses represented within the original data set. For image classification, having enough labelled images is important. Basically, the more images you have, the better learning you get. However, it is usually difficult and expensive to label images. Though researchers have resorted to tools such as AMT (Amazon Mechanical Turk) to have a large number of people help with labelling, there are still financial costs and concerns about the quality of labelling. Thus the ability to use *unlabelled images* would greatly enhance an autonomous feature identification technique. In addition, it is highly unlikely that a particular dataset will only contain the classes of the images we are interested in. It is much more likely that a dataset will comprise a mix of all kinds of possible rock classes. Thus, we utilized self-taught learning to directly learn feature representation from unlabelled rock images of mixed-class and then applied the feature representation to labelled rock images which we are interested in for classification. In such an approach, the unlabelled images do not have to follow the same distribution as the labelled images, and the labelled images for classification can belong to merely subclasses of the unlabelled images [5]. This attribute is particularly important for applications such as planetary exploration where the potential rock types will be uncertain.

Below, we first present the rock image dataset. Next we provide background on the set of manually selected features, the K-means feature learning approach and the self-taught learning approach. Finally, we show the effects of parameter selection for the feature learning methods as well as the results of classification with both the manual features and both types of learned features.

3.2 Rock image dataset

We photographed 9 different types of rock hand samples to generate a rock image dataset. The samples are provided by Department of Earth Science in Western University. These rocks are randomly selected. Table 3.1 lists all these rocks and the brief descriptions. Each type of rock reveals different appearance such as colour, structure, texture and grain size.

Table 3.1: 9 types of rocks used for classification in this study.

Rock types	**Description**
Limestone	It is a sedimentary rock consisting largely of calcium carbonate. It is light grey and smooth to touch.
Volcanic breccia	It is formed from angular gravel and boulder-sized clasts cemented together in a matrix. The angular nature of the clasts indicates that they have not been transported very far from their source. The texture is coarse-grained. Clasts are poorly sorted.
Oolitic limestone	It is made up mostly of ooliths which are sand-sized carbonate particles that have concentric rings of calcium carbonate. The colour is grey and texture is fine grained and porous.
Dolostone	It is a sedimentary carbonate rock that contains a high percentage of the mineral dolomite. It has a stoichiometric ratio of nearly equal amounts of magnesium and calcium.
Rhyolite	It is a silica-rich volcanic rock. Its texture is porphyritic and very compact. The groundmass with varying amounts of glass is also dense and fine grained. The colour is light reddish.
Granite	It is a felsic plutonic rock. It contains high percentage of light coloured constituents and low percentage of dark minerals. So the colour is basically light and texture is phaneritic. The size of the individual constituents is very varied.
Andesite	It is an extrusive rock intermediate in composition between rhyolite and basalt. It is basically grey and lighter coloured than basalt. Texture is porphyritic and interweaved. The groundmass is fine grained and glassy.
Peridotite	It is a very dense and coarse-grained igneous rock. The colour is generally dark greenish-grey and the texture is phaneritic. It is olivine-rich and has low silica content and very little feldspar.
Red granite	It has an equigranular texture with much pink orthoclase, grey quartz and biotite. It is coarse grained and the grains are developed enough to be recognised by the naked eye.

A dataset of approximately 700 textural images was generated from these 9 different types of rocks. There are roughly 80 images in total for each type of rock. Each image has size of

$128 \times 128 \times 3$ pixels and is between 1 and 2 cm across in reality (Figure 3.1). Note that, the scale of the rocks was not accurately measured. Thus, we didn't research on how different scales will affect the classification in this study.

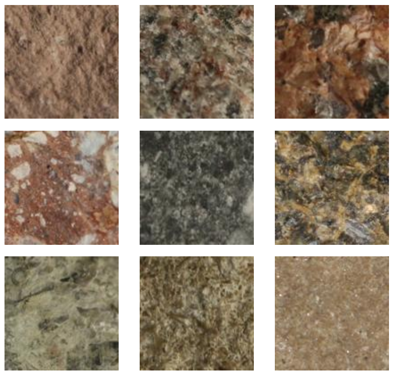


Figure 3.1: Sample images from dataset. Each sample stands for one class. All images have size of $128 \times 128 \times 3$ pixels and are between 1 and 2 cm across . From top to bottom, the first column – rhyolite, volcanic breccia, limestone; the second column – granite, andesite, oolitic limestone; the third column – red granite, peridotite, dolostone.

3.3 Methods

Our first set of experiments compares the behaviour of SVM classifier using two different feature sets: Manually selected features, and autonomously learned features. We first present the two feature sets in this section. The final part of this section presents the needed background to the concept of self-taught learning, which is used in our second set of experiments.

3.3.1 Feature representations

3.3.1.1 Manual features

Among manual features, texture is commonly used to describe and represent the rock images [6, 7]. It is determined by the way in which the gray levels are distributed over the pixels and describes an image as orderly or coarse, smooth or irregular, homogeneous or inhomogeneous [8]. It has been shown that first and second order statistics of texture can reasonably provide a small number of relevant and distinguishable features [9]. Note that, these features do not have any geological meaning. They are carefully hand-crafted by computer scientists.

First-order statistics describes the pixel intensity distribution of the image. If $I(i)$ stands for intensity of pixel i in image, and N is the number of pixels in the whole image, then five of the first-order statistics can be represented as

- Mean
$$\bar{I} = \frac{1}{N}\sum_{i=1}^{N} I(i) \tag{3.1}$$

- Median - intensity value which separates the higher half of pixel intensity from the lower half

- Standard deviation
$$I_{sd} = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(I(i)-\bar{I})^2} \tag{3.2}$$

- Skewness
$$skewness = \frac{\frac{1}{N}\sum_{i=1}^{N}(I(i)-\bar{I})^3}{(\sqrt{\frac{1}{N}\sum_{i=1}^{N}(I(i)-\bar{I})^2})^3} \tag{3.3}$$

- Kurtosis
$$kurtosis = \frac{\frac{1}{N}\sum_{i=1}^{N}(I(i)-\bar{I})^4}{(\sqrt{\frac{1}{N}\sum_{i=1}^{N}(I(i)-\bar{I})^2})^4} \tag{3.4}$$

Here, *skewness* is used to measure the symmetry of the histogram distribution and *kurtosis* is used to describe the flatness of the histogram distribution.

Second-order Statistics describes the information about relative positions of the various intensities. It is calculated from a gray level co-occurrence matrix (GLCM), which describes how frequently two gray levels of pixels appear [10]. For two-dimensional images, there are four gray level co-occurrence matrices in total, by orientation of 0°, 45°, 90°, 135° respectively. Among the second-order statistics calculated from GLCM, four of them are used in this chapter,

- Angular Second Moment(ASM)

$$ASM = \sum_{I_1,I_2} P(I_1,I_2)^2 \tag{3.5}$$

- Entropy

$$Entropy = -\sum_{I_1,I_2} P(I_1,I_2) log P(I_1,I_2) \tag{3.6}$$

- Contrast

$$Contrast = \sum_{I_1,I_2} |I_1 - I_2|^2 log P(I_1,I_2) \tag{3.7}$$

- Correlation

$$Correlation = \sum_{I_1,I_2} \frac{(I_1 - \mu_1)(I_2 - \mu_2)P(I_1,I_2)}{\delta_1 \delta_2} \tag{3.8}$$

Here, $P(I_1, I_2)$ is the frequency of co-occurrence matrix [10]. Four directions(i.e., 0°, 45°, 90°, 135°) are averaged out, which could make it rotate invariantly. *ASM* is to measure the smoothness or uniformity of the image region. *Entropy* is to measure the disorderliness. *Contrast* is a measure of local level variations which takes high values for image of high contrast. *Correlation* is a measure of correlation between pixels in two different directions.

With both first and second order statistics, there will be 9 textural features. If calculated in all colour channels, there will be 27 features in total for each image. In order to show how different feature configurations affect classification results, both the whole feature set and 4 subsets were used to represent the images during our experiments. The configurations of these feature sets are denoted as

- MF I - all first and second order statistics;
- MF II - only first-order statistics;
- MF III - only second-order statistics;
- MF IV - 5 features including Mean, Skewness, Kurtosis, Entropy, Correlation;
- MF V - 5 features including Skewness, Kurtosis, ASM, Entropy, Correlation.

3.3.1.2 Unsupervised feature learning based on K-means

Among various feature learning methods [11, 12], an approach based on K-means has previously been identified as fast and easily implemented [2]. The basic framework is as follows. First, n random sub-patches (i.e., small red squares on *Input image* in Figure 3.3) are extracted from the unlabelled dataset. Each sub-patch has a size of $w \times w \times d$ pixels, where w refers to receptive field size (width of sub-patch) and d is the number of colour channels. Hence, in terms of pixel intensity, the extracted patches can be represented as vectors $(x_1, ..., x_n)$ in $\mathbb{R}^N$, with $N = w \cdot w \cdot d$. Next, K-means [13] algorithm is used to generate K centroids $C_1, ..., C_\mathrm{K}$, where each centroid C_i is also a vector in $\mathbb{R}^N$. All these centroids are the feature filters for the whole dataset and represent a basis set for all images. Figure 3.2 shows the 60 centroids learned from training dataset of rock images. The parameter configuration is provided in Table 3.2. It is worthwhile to note here that these extracted features may not intuitively make much sense, and do not necessarily represent geologically meaningful properties. However, on visual inspection, some of them (e.g., the one in first row and third column) are clearly to detect edges, which are fundamental elements for rock textural images.

With K learned centroids, each patch x_j can be mapped to a $\mathbb{R}^\mathrm{K}$ vector $f(x_j)$, and each element of the vector can be represented as

$$f_i(x_j) = \max\{0, \mu - z_i\}, 1 \leq i \leq \mathrm{K} \tag{3.9}$$

where, $z_i = \|x_j - C_i\|_2$, μ is the mean of all z_i. Essentially, this operation represents all image patches as a weighted combination of the set of identified features. The max operation is used so patches "too far" from a particular centroid are treated as independent of that centroid.

The above steps explain how to transform an input patch from $\mathbb{R}^N$ to $\mathbb{R}^\mathrm{K}$. With this transformation complete, we can now extract a representation of an entire image by applying the transformation to many sub-patches in the whole image. The framework of this process is shown in Figure 3.3. The whole image is first cropped into many sub-patches of the same size $(w \times w \times d)$ with stride (step size) s. If we assume that the entire image has a size of $a \times b \times d$ and stride s is 1, then there will be $(a - w + 1) * (b - w + 1)$ sub-patches in total mapped to $\mathbb{R}^\mathrm{K}$

for each image. After reducing the feature dimensionality with pooling (e.g., split the y^{ij} into four equal-sized quadrants and compute the sum of the y^{ij} in each quadrant), each image will be represented as a feature vector $[\phi_1, \phi_2, ..., \phi_{4K}]^T$ in the same feature space.

Table 3.2: Parameter configuration for feature learning. stride - step size between two adjacent sub-patches, rfsize - receptive field size, K - number of centroids, numPatches - number of sub-patches extracted for training.

stride	1
rfsize	12
K	60
numPatches	50,000

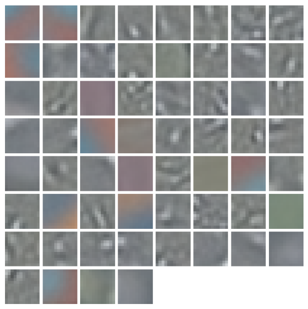


Figure 3.2: 60 centroids learned from training dataset, each centroid has size of $12 \times 12 \times 3$, configuration of parameters refers to Table 6.1.

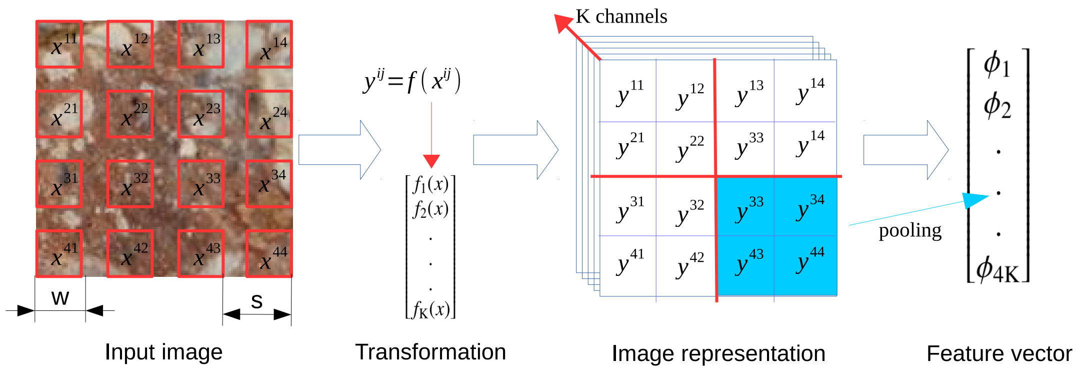

K channels
$y^{ij} = f(x^{ij})$
$f_1(x)$ $f_2(x)$ $f_K(x)$
x^{11} x^{12} x^{13} x^{14} x^{21} x^{22} x^{23} x^{24} x^{31} x^{32} x^{33} x^{34} x^{41} x^{42} x^{43} x^{44}
w
s
y^{11} y^{12} y^{13} y^{14} y^{21} y^{22} y^{33} y^{14} y^{31} y^{32} y^{33} y^{34} y^{41} y^{42} y^{43} y^{44}
pooling
ϕ_1 ϕ_2 ϕ_{4K}
Input image
Transformation
Image representation
Feature vector

Figure 3.3: Framework of representing a rock image with learned features.

3.3.2 Classification method

Support Vector Machine [14, 15] was used to classify rock images with both manual features and learned features. It is one of the most powerful and widely used methods for classification.

It learns a hyperplane or set of hyperplanes in high-dimensional space, which separate data points with the largest margin.

The rock classification here is a multi-class classification problem. Thus, we used one-vs-all linear SVM in the experiments. The objective function is L2-SVM as shown in formula (3.10), where w defines the classifier and ξ_i is the slack variable to deal with outliers of data. We used L2-SVM regularization term because it is differentiable and imposes a bigger loss for data which violate the margin [16].

$$\begin{aligned} &\min \frac{1}{2}\|w\|^2 + \frac{C}{2}\sum_{i=1}^{n}\xi_i^2 \\ &s.t.\ y_i(w^T x_i + b) \geq 1 - \xi_i, i = 1, ..., n \\ &\quad \xi_i \geq 0, i = 1, ..., n \end{aligned} \tag{3.10}$$

The regularization parameter C was determined by 5-fold cross-validation. In 5-fold cross-validation, we first divided the training set into 5 subsets of equal size. Sequentially each subset was tested using the classifier trained on the remaining 4 subsets. Thus, the cross-validation accuracy is the percentage of images which are correctly classified in all of the subsets. Various C values were tried and the one with the best cross-validation accuracy was picked.

3.3.3 Self-taught learning

The method discussed above, using K-means clustering and SVM, requires a fully labelled data set. Unfortunately, in machine learning settings, labelled datasets are significantly harder to obtain than unlabelled ones [17, 18]. Thus, our goal in this second part of the work was to utilize an algorithm (known as *self-taught learning*) that is able to take as much advantage of unlabelled data as possible. Self-taught learning consists of two stages [3]. In the first step, a feature representation is learned from *unlabelled images*. In the second step, the learned features are then used to train a classifier on a *smaller, labelled data set*. Once the general feature representation has been learned in the first stage, it can be used repeatedly for different classification tasks. Unlike semi-supervised learning [19], we do not assume that the unlabelled images were drawn from the same distribution as labelled images. Hence, self-taught learning is more general and powerful [4].

As an example, suppose the task is to identify basalt and limestone. It is time-consuming to gather a large dataset which consists only of basalt and limestone. A more common case and an easier approach is to collect a dataset containing various types of rocks (e.g., breccia, gneiss, sandstone, etc.) including basalt and limestone, be it from the internet or manually photography. Given that labelling images is costly (requires time of a trained geologist), it would be inefficient to label all images in the dataset if the goal is only the identification of basalt and limestone. The self-taught learning approach would be to use the entire large initial data set for feature learning. The large size of the initial data set means that we learn a generic set of features about rocks of all types. Then, we apply the learned feature representation to another small labelled dataset containing *only* basalt and limestone to train a classifier. The reason why this approach works is that the other types of rocks contain some basic visual patterns ("basic elements" as mentioned in [3]) similar to ones in basalt and limestone, such as edges. Therefore, self-taught learning learns how to represent images in terms of these basic elements. By applying this learned representation to labelled images, we can obtain a higher level representation of labelled data as well, thus an easier supervised learning task.

The formalism of self-taught learning is as follows. We have unlabelled dataset X_u of n examples drawn from k classes, $X_u = \{x_1^{C_1}, ..., x_i^{C_j}, ..., x_n^{C_k}\}$. In addition, there is also a set of m labelled examples $X_l = \{x_1^{C_u}, ..., x_i^{C_v}, ..., x_m^{C_w}\}$. These labelled examples come from classes $\{C_u, ..., C_v, ..., C_w\}$, which is a subset of original classes, $\{C_u, ..., C_v, ..., C_w\} \in \{C_1, ..., C_j, ..., C_k\}$. The task is to learn feature representation from X_u, and then apply the feature representation to X_l for further classification. Figure 3.4 shows the general framework.

3.4 Experimental design

We conducted experiments to test both approaches discussed in this chapter. To test the performance of unsupervised feature learning, we first compared classification performance using various combinations of manual features (i.e. first and second order statistics) and with classi-

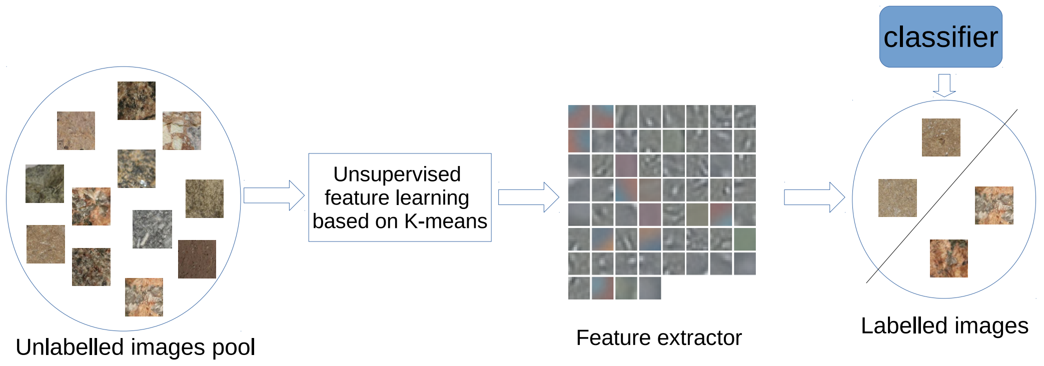


Figure 3.4: Framework of self-taught learning.

fication performance using a feature set learned using the K-means approach. Figure 3.5 shows how we separated the dataset into a training set and a testing set. For all manual features and unsupervised feature learning, the whole dataset was split into 70% as training data and 30% as testing data.

In the second part of the work, we explore the concept of self-taught learning. In this case, the whole initial dataset was split half-and-half into a labelled part and an unlabelled part ("unlabelled" here means we ignored the labels). We used the unlabelled half of the data to perform the feature identification step. In the labelled half, only samples from rock type #1 ~ rock type #4 (which are rhyolite, volcanic breccia, limestone and granite) were picked to classify, and this sub-dataset was separated as 60% for training and 40% for testing.

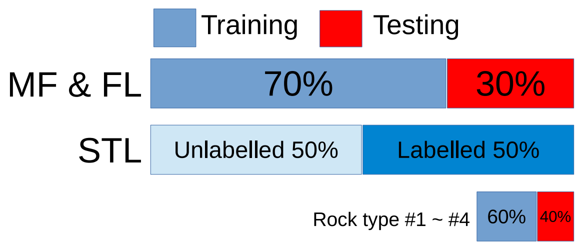


Figure 3.5: Dataset separation. MF-manual features, FL-feature learning, STL-self-taught learning.

3.5 Results and Discussion

3.5.1 Parameters for unsupervised feature learning

There are several tunable parameters for unsupervised feature learning, such as the receptive field size (rfsize), the step size (stride) and the number of centroids (K). Below, we present the results of experiments that investigated how these parameters affect the performance and how to choose these parameters.

3.5.1.1 Number of centroids

We extracted feature representations with 10, 15, 20, 30, 45 and 60 centroids and fixed the receptive field size (6 pixels) and stride (1 pixel). Figure 3.6 clearly shows that the test accuracy generally goes up as the number of the centroids (K) increases. This is reasonable because a larger dictionary of feature bases is usually better able to capture structures and patterns inherent in the images. This is consistent with the work of [1] and [20], who also observed that learning large numbers of features can substantially improve supervised classification results. As such, it is best to set K as large as computing resources will allow.

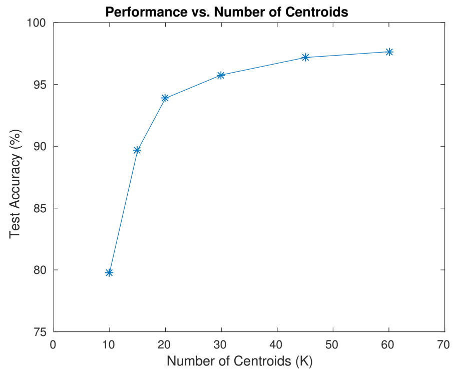


Figure 3.6: Performance vs. number of centroids, rfsize = 6, stride = 1.

3.5.1.2 Stride

Stride is the space between sub-patches where features will be extracted (See Figure 3.3). In this experiment, we fixed the number of centroids (60) and receptive field size (6 pixels), and then chose the stride over 1, 2, 4 and 8 pixels (Figure 3.7). There is a clear downward trend in performance with increasing step size as expected. The smaller stride is able to cover more details in the images, so it will provide a better representation of the images. However, a small stride is also computationally more expensive. Thus, as with our recommendations for the number of centroids, it is best to set stride as small as compute resources will allow.

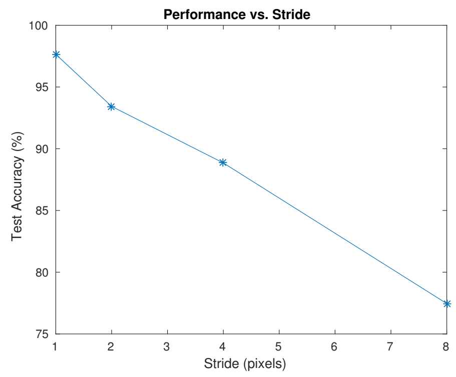


Figure 3.7: Performance vs. stride, K = 60, rfsize = 6.

3.5.1.3 Receptive field size

Receptive field size represents the size of an area in an image from which the features are extracted. In general, a larger receptive field size should result in the learning of more complex features that cover a larger region of the images. However, this will increase the dimensionality of the feature space (see section 3.1.2) and may require the learning of more feature bases or require more images in order to get the same performance. We evaluated the effect of receptive field size by testing it on 6, 8, 10 and 12 pixels. For the other parameters, we used stride of 1 pixel and 60 centroids. Figure 3.8 shows that all the numbers performed similarly and the 6 pixels outperformed others slightly. It is unclear as to whether there is, or could be, a general rule for choosing the receptive field size. However, given that the small receptive field size produces a low dimensionality of the feature space, which in turn reduces computation and also that our experiment showed the small size can work reasonably well, it is suggested to use a small receptive field size if the computing resources allow a large K and a small stride.

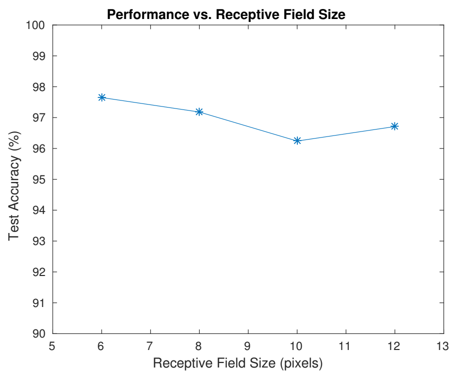


Figure 3.8: Performance vs. receptive field size, K = 60, stride = 1.

3.5.2 Performance comparison

We used one group of parameters (Table 3.2) for unsupervised feature learning, and compared the classification result based on it with the results based on using manual features. Among the parameters, numPatches is the number of extracted sub-patches for learning features. The value of it depends on the size of dataset and the size of individual image sample.

Table 3.3 shows the testing accuracy for all the methods. Note that accuracy varies considerably for different combinations of manual features. MF I has accuracy as high as 96.24%, while MF III only achieves 66.20%. It is apparent that, for our dataset, pure first order statistics (MF II) outperforms second order statistics (MF III) considerably in representing rock images - 95.77% VS 66.20%. However, adding second order statistics to first order statistics can further improve the performance (from 95.77% (MF II) to 96.24% (MF I). The goal, however, was not to compare the manual feature combinations and see which one provides the best result, but rather to show how much variability there is in the results depending on the features used. As it turns out, even slight changes in feature combinations may cause large difference in performance, such as MF IV (92.02%) and MF V (74.18%). So, it clearly indicates that manually selecting appropriate features is difficult.

In this case, one may resort to automatic feature selection methods such as filter and wrapper methods [21]. With these feature selection methods, one may know what features contribute much to representing images. However, there still exist limitations such as what feature selection method to use and what features to choose from. In addition, there is no guarantee that well-selected features for one image dataset A can be applied to another dataset B. For example, the dataset we used here appears to yield good results with the first-order statistics, but there may be other datasets that instead require the second-order statistics. Therefore, one has to conduct different feature selections for different datasets and to make sure the pool containing the features is large enough. If not, hand-crafting new and complicated features such as SIFT [22] might be needed.

While selecting manual features is time-consuming, unsupervised feature learning is more straightforward . The feature learning based on K-means we implemented in this chapter can autonomously learn feature representation from training data, and get a relatively higher testing accuracy as high as 96.71%. Although we cannot guarantee this feature learning method would outperform any manual feature setting other than first and second order statistics, this flexible and easily implemented method is capable of working well.

Table 3.3: Performance of different methods. MF-manual features, FL-feature learning, STL-self-taught learning.

Features	Test accuracy
MF I	**96.24%**
MF II	95.77%
MF III	66.20%
MF IV	92.02%
MF V	74.18%
FL	**96.71%**
STL	**90.32%**

Self-taught learning gets test accuracy as high as 90.32%. Features are learned from “unlabelled” data (first half of the whole dataset) with the same feature learning method and parameter configuration as in Table 3.2. The reason why the accuracy for this approach is not as good as FL is that we are using fewer data to both learn feature representation and train the classifier. This is not unexpected, because the more data available for learning features will result in a more generalized representation and more data for training will also result in a better classifier. Another reason is we are applying the feature representation learned from one subset (“unlabelled”) to another subset (“labelled”), rather than to the same subset as in FL, where learning feature and training classifier share the same subset (70% of the whole dataset). So, just as classifier typically performs better with the training data than testing data, feature representation performs better on the same training subset in FL.

3.6 Conclusion

In the first part of the work, we conducted rock image classification with various combinations of manual features as well as unsupervised feature learning. The results of these experiments show that different combinations of manual features affected classification substantially; whereas unsupervised feature learning based on K-means performed pretty well. While there is no guarantee that this feature learning method can absolutely outperform any manual features configuration, it is easily implemented and more flexible than the manual features.

We also explored the use of self-taught learning based on unsupervised feature learning for classification of rock images. The approach proved promising. It can learn the feature representation directly from unlabelled images of mixed rock types, and then repeatedly apply the feature representation to different sub-classes of rocks. We suggest that the fundamental reason as to why this approach works is that rock images share some basic visual patterns or elements. As such, as long as these basic patterns can be learned from the whole mixed dataset, they can be well utilized for representing the new groups of images belonging to the sub-class.

Bibliography

[1] Adam Coates, Andrew Y Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *International conference on artificial intelligence and statistics*, pages 215–223, 2011.

[2] Adam Coates and Andrew Y Ng. Learning feature representations with k-means. In *Neural Networks: Tricks of the Trade*, pages 561–580. Springer, 2012.

[3] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng. Self-taught learning: transfer learning from unlabeled data. In *Proceedings of the 24th international conference on Machine learning*, pages 759–766. ACM, 2007.

[4] Hua Wang, Feiping Nie, and Heng Huang. Robust and discriminative self-taught learning. In *Proceedings of The 30th International Conference on Machine Learning*, pages 298–306, 2013.

[5] Rajat Raina. *Self-taught learning*. Stanford University, 2009.

[6] Leena Lepistö, Iivari Kunttu, Jorma Autio, and Ari Visa. Rock image classification using non-homogenous textures and spectral imaging. 2003.

[7] Pavel Paclík, Serguei Verzakov, and Robert PW Duin. Improving the maximum-likelihood co-occurrence classifier: a study on classification of inhomogeneous rock images. In *Image Analysis*, pages 998–1008. Springer, 2005.

[8] Changjing Shang and Dave Barnes. Support vector machine-based classification of rock texture images aided by efficient feature selection. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*, pages 1–8. IEEE, 2012.

[9] Namita Aggarwal and RK Agrawal. First and second order statistics features for classification of magnetic resonance brain images. *Journal of Signal and Information Processing*, 3(2):146–153, 2012.

[10] Robert M Haralick, Karthikeyan Shanmugam, and Its' Hak Dinstein. Textural features for image classification. *Systems, Man and Cybernetics, IEEE Transactions on*, (6):610–621, 1973.

[11] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y Ng. Efficient sparse coding algorithms. In *Advances in neural information processing systems*, pages 801–808, 2006.

[12] Quoc V Le, Alexandre Karpenko, Jiquan Ngiam, and Andrew Y Ng. Ica with reconstruction cost for efficient overcomplete feature learning. In *Advances in Neural Information Processing Systems*, pages 1017–1025, 2011.

[13] Tapas Kanungo, David M Mount, Nathan S Netanyahu, Christine D Piatko, Ruth Silverman, and Angela Y Wu. An efficient k-means clustering algorithm: Analysis and implementation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):881–892, 2002.

[14] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[15] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

[16] Yoshiaki Koshiba and Shigeo Abe. Comparison of l1 and l2 support vector machines. In *Neural Networks, 2003. Proceedings of the International Joint Conference on*, volume 3, pages 2054–2059. IEEE, 2003.

[17] Richard O Duda, Peter E Hart, et al. *Pattern classification and scene analysis*, volume 3. Wiley New York, 1973.

[18] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100. ACM, 1998.

[19] Xiaojin Zhu, Zoubin Ghahramani, John Lafferty, et al. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, volume 3, pages 912–919, 2003.

[20] Jan C Van Gemert, Jan-Mark Geusebroek, Cor J Veenman, and Arnold WM Smeulders. Kernel codebooks for scene categorization. In *European conference on computer vision*, pages 696–709. Springer, 2008.

[21] Girish Chandrashekar and Ferat Sahin. A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28, 2014.

[22] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

Chapter 4

Learning Spatial-spectral Features for Hyperspectral Image Classification

Published as Shu, L., McIsaac, K., & Osinski, G. R. in IEEE Transactions on Geoscience and Remote Sensing, 2018. ©2018 IEEE.

4.1 Introduction

As we describe in section 2.2, combining spatial contextual information with spectral information can dramatically improve the performance of hyperspectral image classification. There have been a lot of conventional methods to do so, such as morphological profile (MP) based methods [1, 2, 3], composite kernel (CK) based methods [4, 5, 6] and Markov random field (MRF) based methods [7, 8, 9]. However, these methods have their own limits. MP-based methods are heavily hand-crafted and are therefore less flexible for different datasets. CK-based methods are a strong competitor, but the spatial features extracted by CKs are usually so simple that they do not accurately represent complex spatial structures. MRF–based methods have shown great performance, but the performance heavily depends on the initial pixel-wise classification which still require manual feature selection.

In this chapter, we propose a simple but innovative framework to automatically generate spatial-spectral features. Our approach is more flexible than MP-based feature methods, and can cover more complicated spatial structures than CK-based methods. Compared with the two steps on MRF-based methods - pixel-wise classification and smoothing classification, our approach's direct classification with spatial-spectral features is more straightforward and the spatial-spectral features are also more advanced than the features used by MRF's initial pixel-wise classification.

The rest of chapter is organized as follows. Section 4.2 presents the proposed framework for generating the spatial-spectral features. Two different feature sets are proposed. The first is based on principle component analysis (PCA) and the second is based on K-means clustering. Section 4.3 presents results from classification experiments and discusses trade-offs between the different approaches. We present some concluding remarks in Section 4.4.

4.2 Method

A typical process for spectral imagery classification includes extracting feature representation for the pixels in the spectral imagery and feeding the feature representation into a classifier. In general, the performance of a classifier is heavily dependent on the selection of the feature representation. We first design a flexible and powerful technique for extracting a spatial-spectral feature representation. In this step, we have investigated two unsupervised learning methods: K-means and PCA. Once we have learned a set of representative features, we use the radial basis function kernel support vector machine to conduct the classification.

4.2.1 Spatial-spectral feature representation

Figure 4.1 illustrates the framework for generating the proposed spatial-spectral features. Instead of representing each pixel only by its own spectral intensity, we crop a $W \times W \times B$ cuboid of which the host pixel lies in the middle to extract the feature representation. The patch size - W - in each band is odd (e.g., 3 or 5). Larger values of W allow capture of more spatial details.

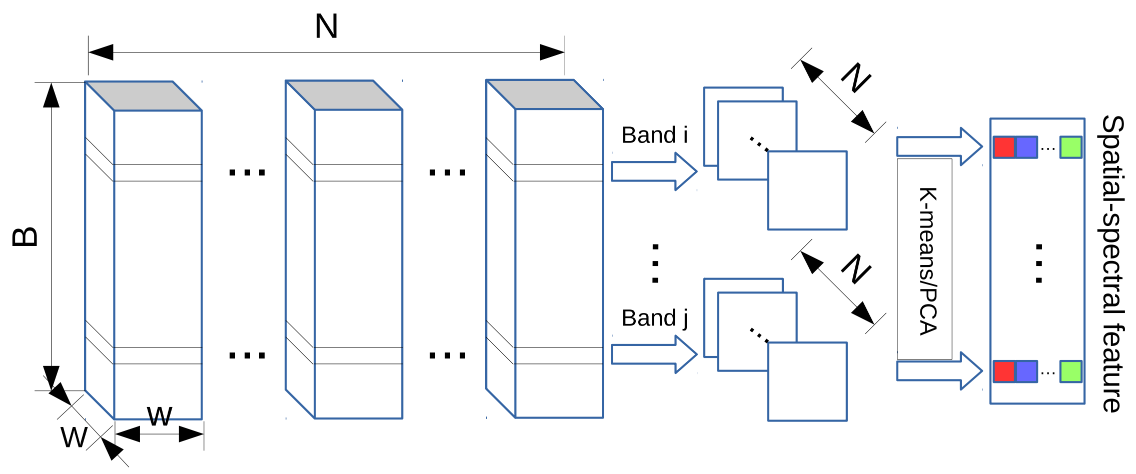


Figure 4.1: Framework for generating spatial-spectral features. B is the number of spectral bands, W is the patch size and N is the number of spectral cuboids cropped from the spectral image. In each cuboid, the host pixel lies in the middle of the W by W neighboring patch.

B is the number of spectral bands. For pixels on the border of the spectral image, we pad the image with a mirror reflection of itself. With *N* cropped spectral cuboids, we can use K-means or PCA as the feature extractor to compress and extract representative spatial features in each band (The original spectral image will be decorrelated along spectral bands but the dimensionality is not necessarily reduced. We discuss the preprocessing of the original spectral image in Section 4.3). Each band will have its own spatial feature elements (shown as coloured squares in Figure 4.1). Thus, the whole output for each band in spectral imagery is a feature map, of which the number of channels equals to the number of Spatial-centroids/Spatial-PCs. By concatenating the feature elements in all/principal spectral bands, we get the spatial-spectral feature for each pixel. Assuming that the number of spectral bands is *B* and the number of Spatial-centroids/Spatial-PCs is K, then the final spatial-spectral feature for each pixel will be a vector with $B *$ K dimensions.

Note that the idea of concatenating spatial feature elements (extracted by K-means/PCA) to generate the spatial-spectral features is actually inspired by the MP-based methods, which concatenate the multilevel attribute profiles from each spectral band. What is primarily different in our work is the method used to extract the spatial features. Our proposed method can automatically learn the features while the MP-based methods resort to manual morphological operations and require trial and error for setting the parameters.

4.2.1.1 Parallel computing

In the proposed framework, the spatial-spectral features are learned independently in each spectral band. Therefore, parallel computing [10, 11] can be carried out to do the calculations simultaneously, with the potential of dramatic reductions in computing time. There have been some works on applying graphics processing unit (GPU) to process hyperspectral images [12, 13]. In our experiments, we have limited our investigation of parallel computing to multicore desktop with MATLAB parallel computing toolbox. The toolbox executes calculation on workers (MATLAB computational engines) which are associated with cores in a multicore machine. Porting the process to a massively parallel computing unit like a GPU is likely to lead to a further improvement in processing speed, but we have left this as future work.

4.2.1.2 Feature Extraction Approach #1: Spatial-Kmeans

K-means is a widely used unsupervised feature learning method [14, 15, 16]. In each spectral band, the patch can be represented as a vector x_j in $\mathbb{R}^S$, with $S = W \cdot W$. We run the K-means algorithm to generate K centroids from N patches. Each centroid C_i is also a vector in $\mathbb{R}^S$. All these centroids are the spatial feature extractors in each band. The feature representation for each patch is then extracted by evaluating the Euclidean distance from the patch to these centroids. Concretely, each patch x_j can be mapped to a $\mathbb{R}^{\mathrm{K}}$ vector $f(x_j)$, and each element of $f(x_j)$ can be represented as

$$f_i(x_j) = \max\{0, \mu_j - z_{ij}\}, \quad 1 < i \leq \mathrm{K} \tag{4.1}$$

here, $z_{ij} = \|x_j - C_i\|_2$, μ_j is the mean of z_{ij} over i. The definitions of mathematical symbols in Formula (4.1) are summarized as follows: x_j stands for jth patch and $x_j \in \mathbb{R}^S$; $f(x_j) \in \mathbb{R}^{\mathrm{K}}$, it is the feature representation of patch x_j; $f_i(x_j)$ is the ith element in $f(x_j)$ and $1 < i \leq \mathrm{K}$; C_i is the ith centroid vector learned by K-means; z_{ij} is the Euclidean distance from patch x_j to centroid C_i; μ_j is the average distance from patch x_j to all the centroids.

For the case of K = 1, where the mean vector is right the centroid vector, we use the reciprocal of the distance from each patch to the centroid as the feature element:

$$f_i(x_j) = \frac{1}{z_{ij} + \epsilon}, \quad i = 1 \tag{4.2}$$

where ϵ is a small number (we use 10^{-6}) which is to avoid the risk of distance being zero.

In Figure 4.1, the coloured squares denote the feature elements. By concatenating the feature elements in all the bands we construct a final spatial-spectral feature representation.

Including preprocessing, the full routine of K-means feature extractor (Spatial-Kmeans) is summarized here:

a. Normalize spatial patches in each spectral band among the training dataset: $X := \frac{X - mean(X)}{\sqrt{var(X)}}$. X is $N \times S$ matrix, where N is the number of samples (or patches) and $S = W \cdot W$.

b. Run K-means to generate K centroids from N patches.

c. Project patches from a $\mathbb{R}^S$ vector x_j to a $\mathbb{R}^{\mathrm{K}}$ vector $f(x_j)$. Each element of $f(x_j)$ can be represented as:

$$\underset{1 \le i \le \mathrm{K}}{f_i(x_j)} = \begin{cases} \dfrac{1}{z_{ij} + \epsilon}, & \mathrm{K} = 1 \\ \max\{0, \mu_j - z_{ij}\}, & \mathrm{K} > 1 \end{cases}$$

Since K-means may be subject to finding a local optimum based on different initialization of the clustering centroids, we repeat the clustering for 10 times with randomly initialized cluster centroid positions. The solution adopted is the one with the lowest within-cluster sums of point-to-centroid distance.

4.2.1.3 Feature Extraction Approach #2: Spatial-PCA

Inspired by the concept of EigenFaces [17], we utilize PCA [18, 19] to generate Spatial-PCs (we call them *EigenPatches*) as the feature bases for each band. Just as faces can be represented as a linear combination of the EigenFaces, the spatial patches in each band can be represented as a linear combination of the EigenPatches.

As with our K-means feature extractor, the patches in each spectral band can be represented as a vector x_i in $\mathbb{R}^S$, with $S = W \cdot W$. We conduct PCA to extract the top P principal components (Spatial-PCs) in each band, with which each patch x_i can be projected from $\mathbb{R}^S$ to $\mathbb{R}^P$. The coloured squares in Figure 4.1 denote the PCA feature elements.

Including preprocessing, the full routine of PCA feature extractor (Spatial-PCA) is summarized here:

a. Normalize spatial patches in each spectral band among the training dataset: $X := \frac{X - mean(X)}{\sqrt{var(X)}}$. X is $N \times S$ matrix, where N is the number of training data (or patches) and $S = W \cdot W$.

b. Compute the covariance matrix: $\Sigma = \frac{1}{N} X^T X$.

c. Calculate eigenvectors using singular value decomposition: $[U, S, V] = svd(\Sigma)$. Columns of U are eigenvectors.

d. Select the top P columns (eigenvectors) in U as the Spatial-PCs and use them to project each patch x_i to the lower space $\mathbb{R}^P$:

$$x_i := U(:, 1 : P)^T * x_i \tag{4.3}$$

4.2.2 Classifier

Support vector machine (SVM) is a well-known classifier which learns a hyperplane or set of hyperplanes in high-dimensional space to separate the data points with the largest margin [20]. SVM has been particularly preferred in the field of hyperspectral analysis due to its capability to generalize from a limited amount of training samples [21]. The SVM with Gaussian radial basis function (RBF) kernel is utilized to conduct the classification with LIBSVM tool [22] in our experiments. The regularization parameters are determined by 5-fold cross-validation. In 5-fold cross-validation, we first divide the training set into 5 subsets of equal size. Sequentially each subset is tested using the classifier trained on the remaining 4 subsets. Thus, the cross-validation accuracy is the percentage of images which are correctly classified in all of the subsets. Various parameter values are tried and the one with the best cross-validation accuracy is picked.

4.3 Experiments and results

4.3.1 Hyperspectral datasets

4.3.1.1 Indian pines

The hyperspectral data of Indian pines was acquired by Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) to support soils research in 1992 over Purdue University Agronomy farm northwest of West Lafayette and the surrounding area in Indiana, USA [23]. The area is a mixed agricultural/forest area, where some of the crops present, corn, soybeans, were in early stages of growth.

Figure 4.2a shows a false-color composition of the AVIRIS Indian Pines scene. This dataset consists of 145×145 pixels covering a 2 mile by 2 mile area at 20 m spatial resolution and 224 spectral bands covering the spectral range from 400 nm to 2.5 μm. Several spectral bands were removed from the dataset due to noise and water absorption phenomena, leaving a total of 200 bands to be used in the experiments. The ground-truth of the Indian Pines scene has been designated into sixteen classes (see Figure 4.2b). Table 4.1 shows the specific classes and the respective number of samples.

4.3.1.2 Pavia University

The hyperspectral data of Pavia University was acquired by Reflective Optics System Imaging Spectrometer (ROSIS) during a flight campaign over Pavia, nothern Italy. Figure 4.3 illustrates the true-color image of Pavia Univeristy. It contains 610×340 pixels and the geometric resolution is 1.3 m. There are 103 spectral bands covering the spectral range from 430 to 860 nm. The groud-truth map differenciates 9 classes as shown in Table 4.2.

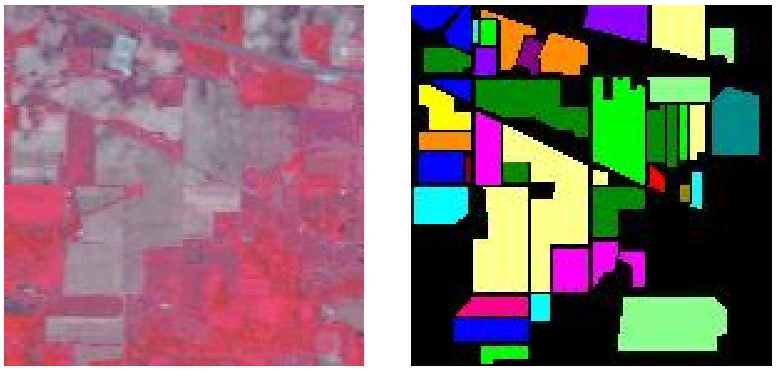
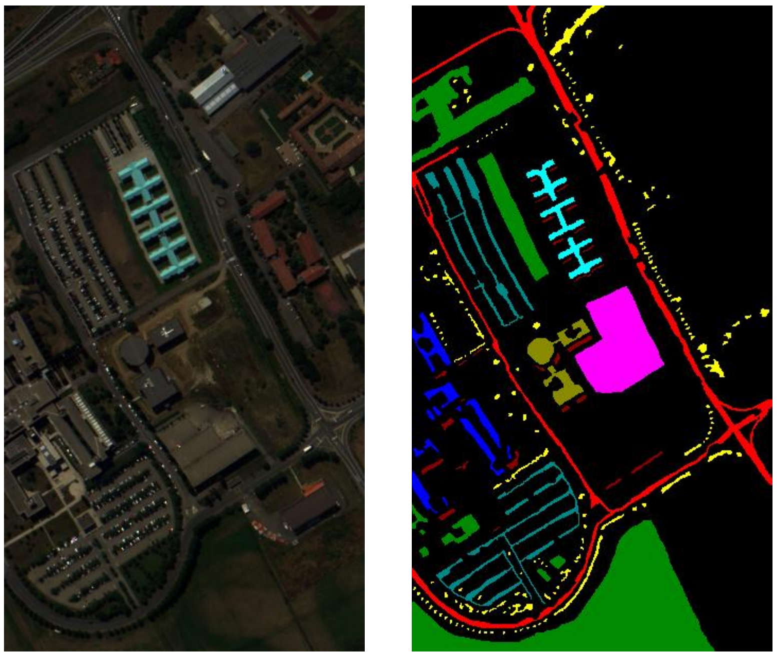


Figure 4.2: Indian Pines dataset. (a) False-color composition of bands 50, 27 and 17. (b) Ground-truth map. The specific classes denoted by different colors refer to Table 4.1.

Table 4.1: Classes for the Indian Pines scene and their respective number of samples.

#	**Class**	**Num. of samples**
1	Alfalfa	46
2	Corn-notill	1428
3	Corn-mintill	830
4	Corn	237
5	Grass-pasture	483
6	Grass-trees	730
7	Grass-pasture-mowed	28
8	Hay-windrowed	478
9	Oats	20
10	Soybean-notill	972
11	Soybean-mintill	2455
12	Soybean-clean	593
13	Wheat	205
14	Woods	1265
15	Buildings-Grass-Trees-Drives	386
16	Stone-Steel-Towers	93
	Total	10249

From both Table 4.1 and Table 4.2 we see that two datasets have unbalanced numbers of labelled samples per class. Therefore, the classification is challenging. To investigate the performance of the proposed methods, we randomly select 5% of the labelled samples per class for training (minimum 10 samples for the classes with extremely limited numbers of samples). The rest of the labelled samples are used for testing. The classification performance is measured by the overall accuracy (OA), average accuracy (AA) and Kappa coefficient (κ). Cohen's



(a) (b)

Figure 4.3: Pavia University scene. (a) True-color composition of bands 53, 31 and 8. (b) Ground-truth map. The specific classes denoted by different colors refer to Table 4.2.

Table 4.2: Classes for the Pavia University scene and their respective number of samples.

#	**Class**	**Num. of samples**
1	Asphalt	6631
2	Meadows	18649
3	Gravel	2099
4	Trees	3064
5	Painted metal sheets	1345
6	Bare Soil	5029
7	Bitumen	1330
8	Self-Blocking Bricks	3682
9	Shadows	947
	Total	42776

Kappa [24] is a metric that measures how much better the classifier is performing over the performance of a classifier that simply guesses at random according to the frequency of each class. It is considered a good metric for imbalanced and multi-class classification problems. A higher Kappa score indicates better performance. Since the performance varies with different selection of limited training samples, the mean and standard deviation of OA, AA and κ over 10 Monte Carlo runs are recorded for evaluating the performance.

4.3.2 Analysis and discussion

In the rest of this section, we first show our approach to preprocessing the spectral image to enhance the performance. We then show that a parallel computing scheme can dramatically accelerate the calculation of the proposed spatial-spectral features. Next we illustrate two sets of feature bases that have been automatically learned by our two proposed methods as long as the parameters are well set. After that, two tunable parameters (i.e., patch size and number of Spatial-centroids/Spatial-PCs) are analyzed with experiments on different parameter settings. Last but not least, we carry out a performance comparison between our methods and seven other methods taken from prior work.

4.3.2.1 Preprocessing the spectral image

More often than not, the spectral bands of hyperspectral image are highly correlated. Though there have been some works on band selection [25, 26], using a dimensionality reduction method (e.g., PCA) to compress the spectral bands is more common [27, 28, 29]. With PCA, the spectral dimensionality can be dramatically reduced with tiny errors by selecting the top principal components. This can decrease computation and resolve the problem of 'curse of dimensionality'. However, since reducing dimensionality will inevitably cast away a small amount of useful spectral information, as our experiments will show, it is not necessary to do so if computing resources allow. Our proposed methods use Gaussian kernel SVM as the classifier, which projects the features into an infinite high-dimensional space. It is powerful to generalize the features as long as the model parameters are carefully tuned [21]. As a matter of fact, our experiments show that performance improves with the inclusion of more principal components.

Table 4.3 and Figure 4.4 show the performance of different ways of preprocessing the original spectral image of Indian Pines. *Raw* means the original spectral data with 200 bands is intact; the other settings correspond to retaining different numbers of spectral principal components (Spectral-PCs) after decorrelating with Spectral-PCA. The spatial-spectral feature extraction method used here is Spatial-PCA. The number of Spatial-PCs is set to 5 and the patch size is set to 15. We can see that as the number of the retained Spectral-PCs increases, the performance is generally improved (the mean of OA, AA and κ goes up and the standard deviation goes down). From 10 to 40 spectral-PCs, the performance dramatically bumps up. After that, the performance slightly ramps up since the rest of the spectral components preserve a tiny portion of variance. We note, however, that although the *mean* of the three measurements are almost saturating near the end, the *standard deviation* continues to trend downwards. What can be taken away from this phenomenon is that even small amounts of variance in the last tens of spectral principal components can contribute to the classification and make the feature representations more robust. Of course, retaining more Spectral-PCs will lead to more expensive computation. Therefore, one can make a compromise between the higher performance and faster computation in a specific situation. If computing resources will allow, it is better not to reduce the spectral dimensionality since compressing the data will always inevitably cast away useful information.

Another thing to note is that using PCA to decorrelate the spectral bands as preprocessing before calculating features is better than directly calculating features with the raw spectral bands (see the comparison between *Raw* and *200 Spectral-PCs* in Table 4.3). We believe this is because that decorrelating spectral bands can highlight the spatial-spectral features which are useful to distinguish the classes.

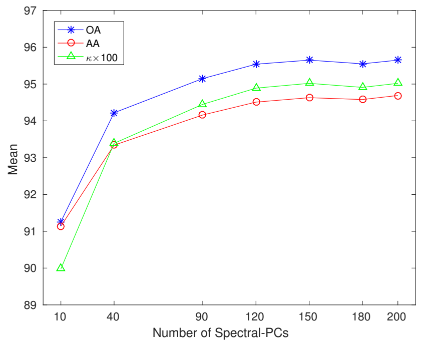
(We make a brief aside here. This preprocessing step uses principle component analysis (PCA), but it should not be confused with the use of PCA to extract a feature representation as discussed above. In an attempt to avoid confusion of terms, in what follows, we will use the term *Spectral-PCA* to refer to the preprocessing of spectral bands and *Spatial-PCA* to refer to the feature extraction step.)

4.3.2.2 Parallel computing

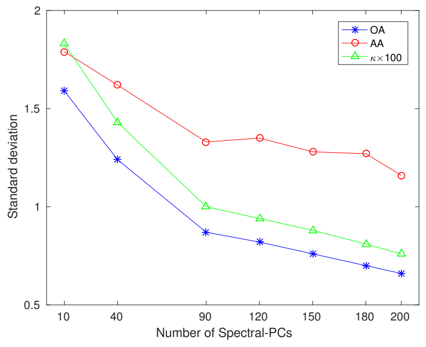
Since the proposed features can be learned independently in each spectral band, computing can simply be done in parallel. We run MATLAB 2015b on a desktop with Intel Core i7-3930K 3.20GHz CPU and 32 GB memory. The MATLAB parallel computing toolbox creates

Table 4.3: Performance of different ways of preprocessing the original spectral image of Indian Pines.

	10 Spectral-PCs	40 Spectral-PCs	90 Spectral-PCs	120 Spectral-PCs
OA	91.26 ± 1.59	94.22 ± 1.24	95.15 ± 0.87	95.54 ± 0.82
AA	91.14 ± 1.79	93.34 ± 1.62	94.16 ± 1.33	94.51 ± 1.35
$\kappa \times 100$	89.98 ± 1.83	93.39 ± 1.43	94.45 ± 1.00	94.89 ± 0.94
	150 Spectral-PCs	180 Spectral-PCs	200 Spectral-PCs	Raw
OA	95.65 ± 0.76	95.55 ± 0.70	95.65 ± 0.66	88.15 ± 1.15
AA	94.63 ± 1.28	94.58 ± 1.27	94.69 ± 1.16	87.41 ± 2.15
$\kappa \times 100$	95.02 ± 0.88	94.91 ± 0.81	95.02 ± 0.76	86.40 ± 1.34



(a)



(b)

Figure 4.4: (a) Mean and (b) standard deviation of three measurements - OA, AA and κ - over 10 random runs with different number of Spectral-PCs retained for Indian pines dataset.

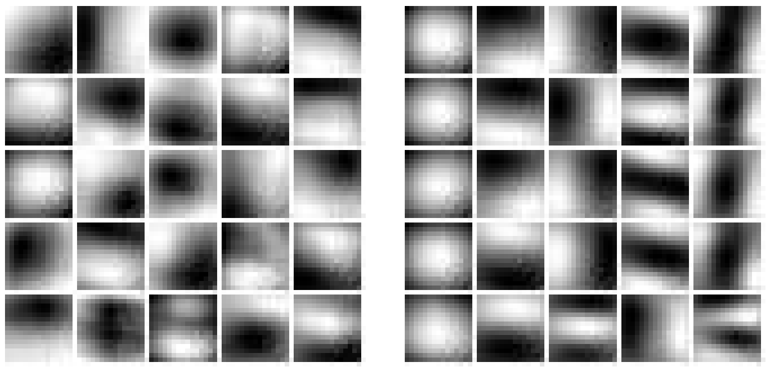
a parallel pool with 6 computational engines in the multicore machine. The running time of both serial computing and parallel computing has been recorded in Table 4.4. In this experiment, 150 Spectral-PCs are retained in preprocessing, patch size for both Spatial-Kmeans and Spatial-PCA is 15 and the number of Spatial-centroids/Spatial-PCs is 2. Clearly, the parallel computing can dramatically speed up the computation of the spatial-spectral features. Since calculating Spatial-PCA features is not an iterative process, Spatial-PCA is much faster than Spatial-Kmeans.

Table 4.4: Time cost of serial/parallel computing for Spatial-Kmeans and Spatial-PCA for Indian pines.

	Serial	Parallel
Spatial-Kmeans	665 s	352 s
Spatial-PCA	32 s	18 s

4.3.2.3 Feature bases

With patch size fixed to 15 pixels, we show the feature bases of 5 bands of Indian pines (i.e., #10, #20, #30, #50 and #100) for both Spatial-Kmeans and Spatial-PCA feature methods in Figure 4.5. These feature bases are automatically learned as long as the parameters are well set, providing more flexibility than the conventional feature methods. It is fascinating that the feature bases appear similar between different bands. For example, the first Spatial-PCs (the first column in Figure 4.5b) in all 5 bands are all blob shaped. Also worth noting is that Spatial-Kmeans features are similar to Spatial-PCA features to some extent.



(a) (b)

Figure 4.5: Feature bases learned from Indian Pines dataset by both Spatial-Kmeans and Spatial-PCA feature methods. The patch size is set to 15 pixels. From top to bottom, each row corresponds to one spectral band and 5 rows correspond to the top 5 Spectral-PCs. (a) K-means features bases - Spatial-centroids. Each row lists 5 centroids for one certain band. (b) PCA feature bases - Spatial-PCs. Each row lists top 5 Spatial-PCs for one certain band.

4.3.2.4 Parameter sensitivity analysis

There are two tunable parameters for Spatial-Kmeans and Spatial-PCA, i.e., patch size and number of Spatial-centroids/PCs. We first evaluate the effect of patch size by fixing the number of Spatial-centroids/PCs to 5 and varying patches from 3×3 to 15×15. Figure 4.6 and Table 4.5 show the results. Evidently, the performance improves as the patch size increases. This is what we would expect because larger patches are better able to capture more complex features that cover a larger region of the images.

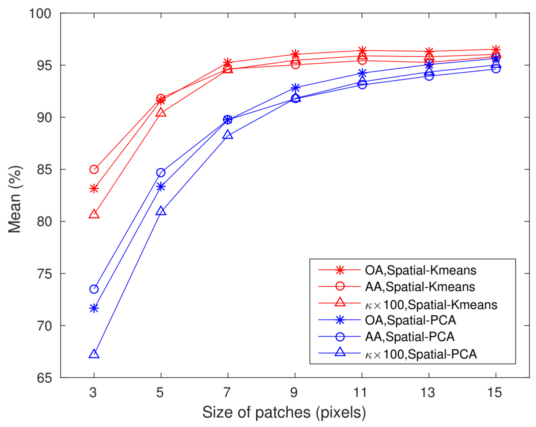


Figure 4.6: Performance with different patch size for Indian pines. Spatial-centroids/Spatial-PCs is set to 5 and top 150 Spectral-PCs are retained. The measurements refer to Table 4.5.

Table 4.5: Performance with different patch size for Indian pines. Spatial-centroids/PCs is set to 5 and top 150 Spectral-PCs are retained.

Patch size		3	5	7	9	11	13	15
Spatial-Kmeans	OA	83.15 ± 0.93	91.62 ± 0.80	95.26 ± 0.51	96.04 ± 0.56	96.42 ± 0.47	96.33 ± 0.68	96.53 ± 0.54
	AA	84.96 ± 1.78	91.86 ± 0.92	94.64 ± 0.51	95.03 ± 0.98	95.44 ± 1.09	95.27 ± 1.35	95.81 ± 0.92
	$\kappa \times 100$	80.66 ± 1.08	90.42 ± 0.91	94.58 ± 0.58	95.47 ± 0.64	95.9 ± 0.54	95.81 ± 0.77	96.04 ± 0.62
Spatial-PCA	OA	71.63 ± 1.00	83.36 ± 0.62	89.74 ± 0.67	92.84 ± 0.84	94.24 ± 0.78	95.06 ± 0.77	95.65 ± 0.76
	AA	73.52 ± 1.69	84.72 ± 1.14	89.76 ± 1.22	91.76 ± 1.40	93.10 ± 1.39	93.95 ± 1.40	94.63 ± 1.28
	$\kappa \times 100$	67.22 ± 1.22	80.91 ± 0.72	88.25 ± 0.76	91.8 0± 0.97	93.41 ± 0.90	94.35 ± 0.88	95.02 ± 0.88

We then conduct experiments by fixing the patch size to 15 and varying the number of Spatial-centroids/PCs from 1 to 7. In Figure 4.7, we see the performance of Spatial-PCA goes all the way down as the number of Spatial-PCs increases and the performance of Spatial-Kmeans also slowly ramps down after a sudden bumping up from 1 to 2. This phenomena is abnormal since intuitively more feature bases should be better able to capture structures and patterns inherent in the image which would lead to a better performance. Given that we are retaining top 150 Spectral-PCs during preprocessing and using Gaussian RBF kernel SVM as the classifier, we wonder whether this downgrade trend is due to overfitting. Thus, we conduct another experiment with more training samples (20%) and a simpler classifier (Linear-SVM) and fewer Spectral-PCs (top 10). As it turns out, the performance does goes up as the number of Spatial-centroids/PCs increases in such a situation (see Figure 4.8 and Table 4.6), which proves that overfitting really is the reason of performance's downgrade in previous experimental setting.

Therefore, if the training samples are sufficient enough, it is better to learn more spatial feature bases. If not, it is better to learn a small number of spatial feature bases for the benefit of computing efficiency and avoiding overfitting.

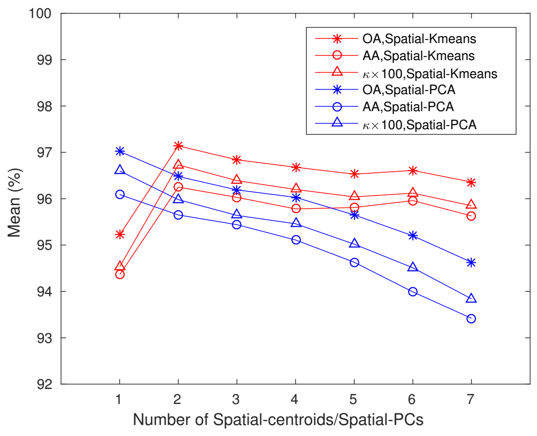


Figure 4.7: Performance with different number of Spatial-centroids/Spatial-PCs for Indian pines. Patch size is set to 15 and top 150 Spectral-PCs are retained.

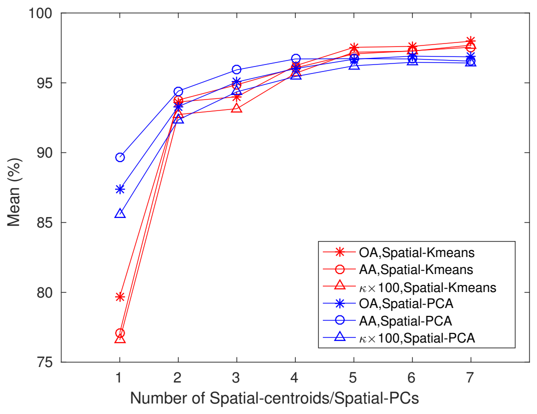


Figure 4.8: Performance with different number of Spatial-centroids/Spatial-PCs for Indian pines. Patch size is set to 15 and top 10 Spectral-PCs are retained. 20% samples are randomly selected for training with linear-SVM. Measurements refer to Table 4.6.

Table 4.6: Performance with different number of Spatial-centroids/Spatial-PCs for Indian pines. Patch size is set to 15 and top 10 Spectral-PCs are retained. 20% samples are randomly selected for training with linear-SVM.

# of Spatial-centroids/PCs		1	2	3	4	5	6	7
Spatial-Kmeans	OA	79.70 ± 0.82	93.63 ± 0.59	94.00 ± 0.48	96.22 ± 0.26	97.54 ± 0.45	97.61 ± 0.29	97.98 ± 0.29
	AA	77.10 ± 1.45	93.77 ± 1.69	94.86 ± 1.06	96.07 ± 1.19	97.07 ± 0.96	97.29 ± 0.52	97.55 ± 0.64
	$\kappa \times 100$	76.59 ± 0.96	92.73 ± 0.67	93.15 ± 0.55	95.68 ± 0.29	97.19 ± 0.51	97.27 ± 0.34	97.70 ± 0.33
Spatial-PCA	OA	87.37 ± 0.70	93.30 ± 0.66	95.06 ± 0.42	96.02 ± 0.31	96.69 ± 0.30	96.91 ± 0.46	96.86 ± 0.45
	AA	89.68 ± 0.98	94.41 ± 0.62	95.93 ± 0.75	96.71 ± 0.55	96.73 ± 0.84	96.71 ± 0.71	96.55 ± 0.90
	$\kappa \times 100$	85.58 ± 0.80	92.35 ± 0.76	94.36 ± 0.48	95.45 ± 0.36	96.22 ± 0.34	96.47 ± 0.53	96.42 ± 0.51

4.3.2.5 Performance comparison

We compared our two proposed methods with 7 other methods. They are Spectral (use only spectral bands as the features), LADA [30], LBP [8], EMAP [3, 2], GCK [5], 3DDWT [31] and SMLR-SpATV [32]. As with previous experiments, 5% training samples are randomly selected for 10 Monto Carlo runs. The number of Spatial-centroids is set to 2 for Spatial-Kmeans and the number of Spatial-PCs is set to 1 for Spatial-PCA. Patch size is set to 15. For the other methods, all the corresponding parameter setting is the same as stated in the references. Preprocessing spectral bands with PCA decorrelation is applied for all the methods.

Table 4.7 illustrates the performance with different methods for Indian pines. We see that Spectral performs badly since it misses the spatial information. By modeling pixels in neighbor regions, LADA performs slightly better than Spectral. Due to the more advanced ways of combining spatial and spectral information, the other methods perform dramatically better than the previous two. Although the overall accuracy of 3DDWT is high, the average accuracy is relatively low and the accuracy for class #4 and #7 is quite variant over 10 Monto Carlo runs. This indicates that 3DDWT is not robust. The proposed Spatial-Kmeans and Spatial-PCA can outperform all the other methods, nearly about 2%~6% improvement. Figure 4.9 shows the classified maps of all the methods. Both Spatial-Kmeans and Spatial-PCA have fewest misclassified pixels. Spatial-Kmeans is slightly better than Spatial-PCA in terms of the classification accuracy, but it is much more computationally expensive than Spatial-PCA. Hence, one can make selection among two of them by compromising between the higher performance and faster computation in a specific situation.

Table 4.8 illustrates the performance with different methods for Pavia University dataset. Since there are more training samples in Pavia University and the data is cleaner than Indian pines, the classification accuracies of all the methods are all higher than 93%. Among all the methods, LBP, 3DDWT and two proposed methods perform very close to each other and the overall accuracies are all above 98%. Though 3DDWT's numbers indicate that 3DDWT performs the best in Pavia University dataset, experiment on Indian pines has shown that its performance is not stable when there are fewer training samples and data is less clean. It is interesting to see that both Spatial-Kmeans and Spatial-PCA seem bad at identifying the 9th class - shadows

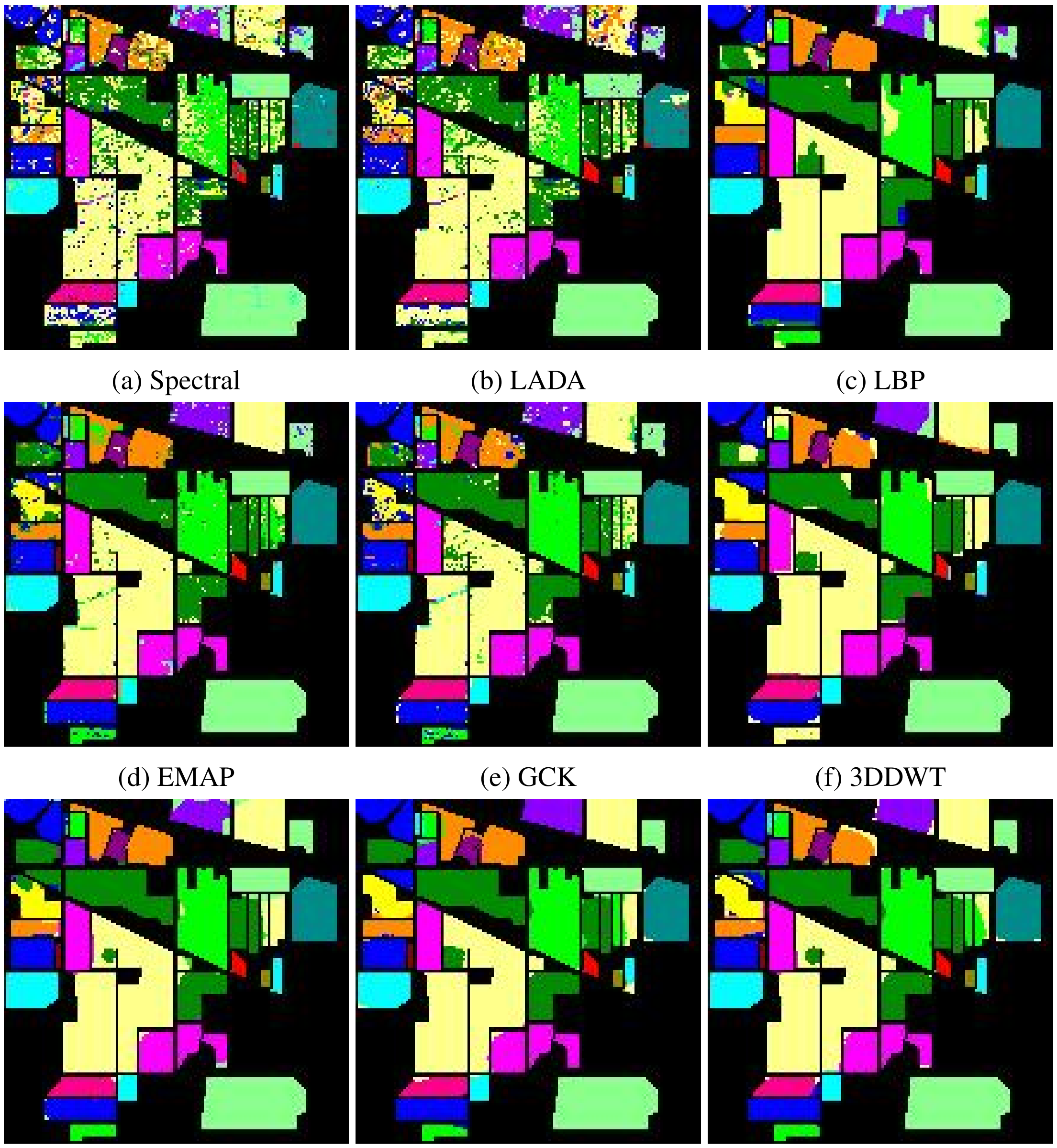
Table 4.7: Performance comparison for Indian pines with different methods - Spectral, LADA [30], LBP [8], EMAP [3, 2], GCK [5], 3DDWT [31] and SMLR-SpATV [32], Spatial-Kmeans and Spatial-PCA. Top 150 Spectral-PCs are retained for both Spatial-Kmeans and Spatial-PCA.

Class	# of samples		Spectral	LADA	LBP	EMAP	GCK	3DDWT	SMLR_SpTV	Spatial-Kmeans	Spatial-PCA
	train	test									
1	10	36	56.94 ± 27.22	75.69 ± 10.78	95.83 ± 1.96	95.56 ± 1.43	92.95 ± 4.21	95.83 ± 5.28	98.06 ± 2.29	97.22 ± 3.93	96.94 ± 3.06
2	72	1356	66.33 ± 3.18	72.38 ± 2.16	88.35 ± 3.23	88.04 ± 2.61	89.51 ± 1.69	92.21 ± 2.71	93.17 ± 2.60	97.26 ± 1.41	96.84 ± 1.47
3	42	788	50.56 ± 8.41	56.30 ± 3.79	78.83 ± 5.77	91.15 ± 2.99	91.62 ± 3.00	92.98 ± 2.85	91.27 ± 7.17	97.12 ± 1.89	96.65 ± 1.60
4	12	225	31.82 ± 9.77	42.22 ± 8.89	66.18 ± 23.58	70.44 ± 13.18	86.44 ± 6.67	76.62 ± 28.34	85.33 ± 9.63	91.82 ± 8.10	91.69 ± 7.42
5	25	458	81.44 ± 5.60	84.31 ± 5.75	88.49 ± 4.71	91.86 ± 3.49	92.99 ± 2.95	91.53 ± 4.19	91.29 ± 4.20	96.29 ± 2.60	97.23 ± 2.04
6	37	693	95.05 ± 1.20	96.01 ± 0.79	99.80 ± 0.25	98.74 ± 0.54	98.79 ± 1.03	97.53 ± 1.13	97.98 ± 0.86	96.75 ± 2.94	96.35 ± 3.12
7	10	18	86.11 ± 5.40	90.28 ± 3.93	98.89 ± 2.34	91.11 ± 5.97	96.88 ± 3.29	57.78 ± 49.84	100.00 ± 0.00	98.33 ± 3.75	98.89 ± 3.51
8	24	454	92.47 ± 5.19	93.28 ± 3.41	99.34 ± 1.02	99.74 ± 0.25	99.50 ± 0.25	99.52 ± 0.82	100.00 ± 0.00	97.95 ± 2.84	97.84 ± 2.62
9	10	10	100.00 ± 0.00	98.75 ± 3.54	100.00 ± 0.00	100.00 ± 0.00	100.00 ± 0.00	98.00 ± 6.32	100.00 ± 0.00	100.00 ± 0.00	99.00 ± 3.16
10	49	923	59.12 ± 9.06	52.51 ± 8.67	86.57 ± 5.27	86.61 ± 2.29	89.30 ± 2.61	84.02 ± 7.11	90.52 ± 4.26	95.83 ± 1.51	95.75 ± 1.56
11	123	2332	83.74 ± 3.08	83.62 ± 2.41	93.61 ± 2.02	94.42 ± 1.65	96.63 ± 1.12	96.44 ± 2.21	98.20 ± 0.44	98.62 ± 0.83	98.48 ± 0.80
12	30	563	57.02 ± 8.03	68.36 ± 5.83	90.98 ± 4.12	84.72 ± 5.11	89.19 ± 3.14	88.15 ± 8.10	92.72 ± 2.58	94.37 ± 2.81	94.65 ± 3.12
13	11	194	91.08 ± 4.05	94.85 ± 1.97	99.54 ± 0.29	98.92 ± 0.66	99.45 ± 0.16	93.25 ± 7.09	99.02 ± 0.82	93.45 ± 3.73	92.89 ± 4.28
14	64	1201	95.78 ± 1.33	96.09 ± 2.09	97.29 ± 1.75	97.60 ± 1.44	98.01 ± 2.73	96.73 ± 2.60	99.81 ± 0.28	99.45 ± 0.53	99.53 ± 0.40
15	20	366	48.39 ± 6.65	59.12 ± 7.38	70.87 ± 6.65	87.73 ± 5.43	88.56 ± 4.02	91.37 ± 7.04	86.20 ± 6.57	94.43 ± 4.11	94.86 ± 4.08
16	10	83	86.39 ± 4.65	85.54 ± 5.54	89.88 ± 5.09	90.96 ± 6.33	84.71 ± 9.38	82.41 ± 8.14	96.27 ± 3.38	91.08 ± 5.54	89.76 ± 6.51
OA			74.87 ± 0.95	77.25 ± 1.04	90.39 ± 1.43	92.04 ± 0.80	93.81 ± 0.70	93.02 ± 0.89	95.08 ± 0.95	97.14 ± 0.41	97.02 ± 0.42
AA			73.89 ± 1.84	78.08 ± 1.37	90.28 ± 1.60	91.73 ± 1.41	93.41 ± 1.02	89.65 ± 4.45	94.99 ± 0.83	96.25 ± 0.72	96.09 ± 0.86
$\kappa \times 100$			70.97 ± 1.12	73.70 ± 1.27	88.99 ± 1.65	90.91 ± 0.92	92.94 ± 0.80	92.01 ± 1.03	94.37 ± 1.10	96.73 ± 0.47	96.60 ± 0.49

Table 4.8: Performance comparison for Pavia University with different methods - Spectral, LADA [30], LBP [8], EMAP [3, 2], GCK [5], 3DDWT [31] and SMLR-SpATV [32], Spatial-Kmeans and Spatial-PCA. All the Spectral-PCs are retained for both Spatial-Kmeans and Spatial-PCA.

Class	# of samples		Spectral	LADA	LBP	EMAP	GCK	3DDWT	SMLR_SpTV	Spatial-Kmeans	Spatial-PCA
	train	test									
1	332	6299	93.03 ± 1.25	94.63 ± 0.44	98.96 ± 0.35	98.57 ± 0.55	98.64 ± 0.41	99.40 ± 0.45	97.91 ± 0.51	97.71 ± 0.62	97.47 ± 0.69
2	933	17716	97.59 ± 0.31	97.64 ± 0.34	99.88 ± 0.09	98.99 ± 0.24	99.27 ± 0.10	99.79 ± 0.13	100.00 ± 0.00	99.90 ± 0.08	99.93 ± 0.06
3	105	1994	76.16 ± 2.81	77.39 ± 2.48	84.73 ± 3.79	91.97 ± 1.77	92.01 ± 1.55	86.24 ± 1.62	91.44 ± 3.49	96.49 ± 1.34	94.72 ± 1.58
4	154	2910	93.48 ± 1.00	94.54 ± 1.24	96.22 ± 1.21	96.98 ± 1.05	96.50 ± 0.85	97.62 ± 0.81	85.10 ± 2.38	92.56 ± 1.40	94.55 ± 1.33
5	68	1277	99.03 ± 0.32	99.25 ± 0.35	99.22 ± 0.40	99.70 ± 0.15	99.27 ± 0.09	99.26 ± 0.51	99.74 ± 0.27	98.78 ± 0.94	99.63 ± 0.29
6	252	4777	88.08 ± 0.89	89.13 ± 0.99	99.25 ± 0.76	94.07 ± 1.62	97.48 ± 0.78	99.37 ± 0.65	100.00 ± 0.00	100.00 ± 0.00	99.99 ± 0.03
7	67	1263	81.49 ± 3.30	82.17 ± 1.37	92.57 ± 2.18	94.80 ± 1.70	94.96 ± 1.31	95.06 ± 1.89	97.36 ± 1.56	99.26 ± 0.45	99.05 ± 0.89
8	185	3497	89.80 ± 0.72	89.13 ± 1.53	96.74 ± 1.28	96.25 ± 1.44	95.89 ± 0.90	96.61 ± 1.44	98.08 ± 0.65	95.04 ± 1.29	94.35 ± 1.23
9	48	899	99.82 ± 0.12	99.77 ± 0.12	99.71 ± 0.36	99.86 ± 0.07	99.69 ± 0.08	99.47 ± 0.38	92.78 ± 3.33	87.95 ± 1.86	89.81 ± 2.30
OA			93.34 ± 0.27	93.84 ± 0.14	98.14 ± 0.27	97.54 ± 0.32	97.99 ± 0.21	98.42 ± 0.16	97.77 ± 0.27	98.14 ± 0.23	98.17 ± 0.27
AA			90.94 ± 0.46	91.52 ± 0.19	96.36 ± 0.47	96.80 ± 0.42	97.08 ± 0.33	96.98 ± 0.30	95.82 ± 0.69	96.41 ± 0.38	96.61 ± 0.55
$\kappa \times 100$			91.15 ± 0.35	91.81 ± 0.18	97.53 ± 0.37	96.73 ± 0.43	97.40 ± 0.28	97.90 ± 0.21	97.04 ± 0.37	97.54 ± 0.30	97.58 ± 0.36

(testing accuracy is less than 90%), which drags down the average accuracy for both methods. We believe this is due to the surrounding areas of Shadows are less consistent than the other classes and have more variant neighboring structures, which is exactly what the proposed methods value more on than the other methods. However, it does not downgrade the overall performance of the proposed methods since their Kappa metrics are still in a high level. Figure 4.10 depicts the classified maps with all the methods. The maps classified by two proposed methods are among the maps with the fewest misclassified pixels.



(a) Spectral (b) LADA (c) LBP

(d) EMAP (e) GCK (f) 3DDWT

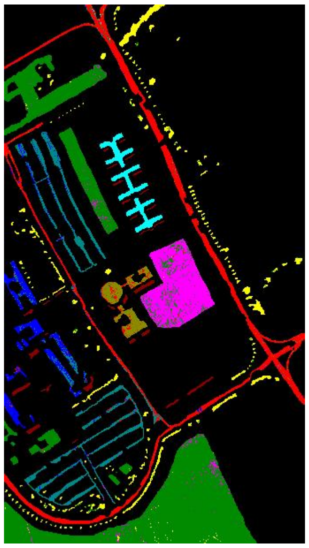
(g) SPTV (h) S-Kmeans (i) S-PCA

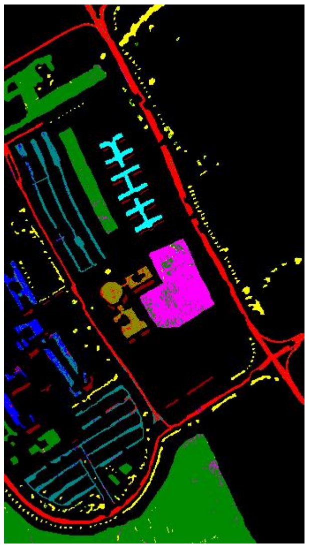
Figure 4.9: Classified maps of Indian Pines with different methods. S-Kmeans is Spatial-Kmeans and S-PCA is Spatial-PCA.

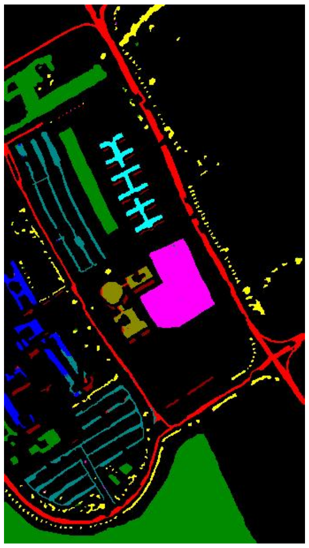
4.4 Conclusion

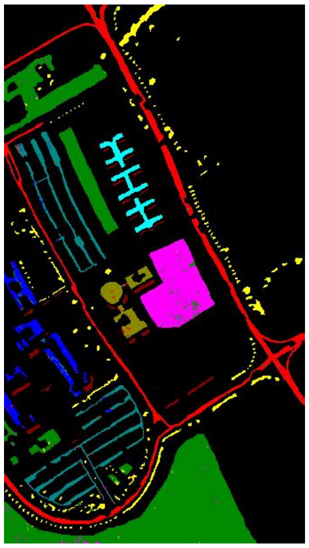
This chapter proposes a simple but innovative framework to automatically learn spatial-spectral features for hyperspectral image classification. Two unsupervised learning methods - K-means and PCA - are utilized to learn the spatial feature bases in each retained spectral band after decorrelation. Then the spatial feature representations are extracted with these spatial feature bases. By concatenating the spatial feature representations in all/principal spectral bands, we get the spatial-spectral features.

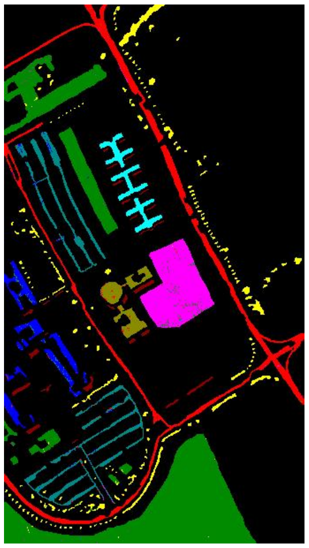
Decorrelating the spectral bands as the preprocessing step can dramatically enhance the performance. Retaining more Spectral-PCs also results in better performance since even the small amount of variance in the last tens of spectral principal components can contribute to the classification and make the feature representations more robust. However, this comes with the cost of increased computation time. As the proposed features can be learned independently in each spectral band, a simple parallel computing scheme is conducted and the computing time is significantly reduced. An even more dramatic reduction in computation time should be possible with the use of GPU processing. The proposed features are automatically learned. As a result, the feature set is more flexible compared with the features used in other methods and can cover more complicated spatial structures. Moreover, the proposed feature methods can outperform the other state-of-the-art methods as shown in experiments.

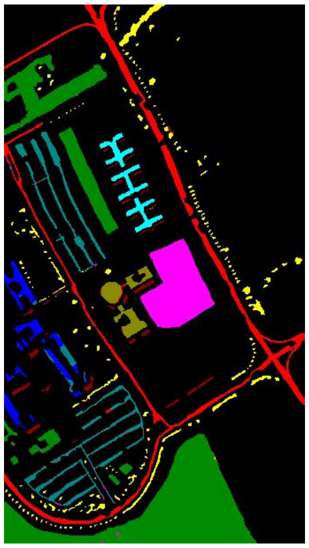

(a) Spectral

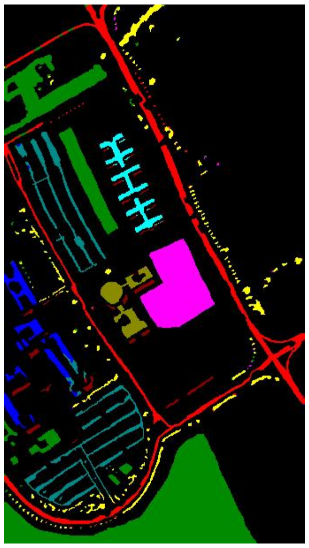

(b) LADA

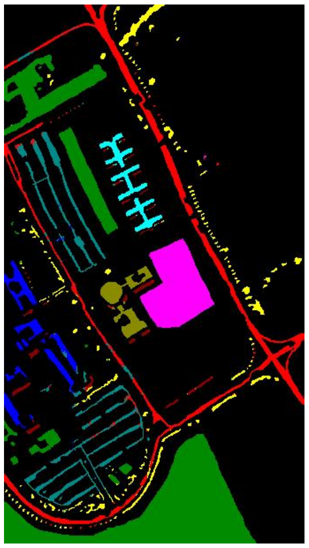

(c) LBP


(d) EMAP


(e) GCK


(f) 3DDWT


(g) SPTV


(h) S-Kmeans

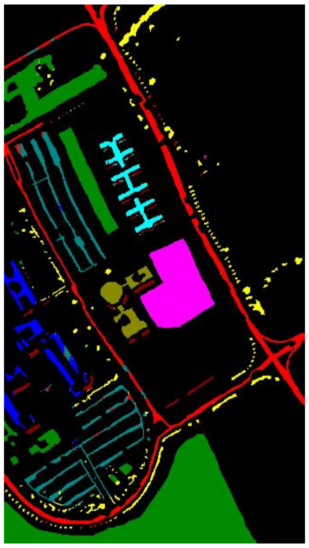

(i) S-PCA

Figure 4.10: Classified maps of Pavia University with different methods. S-Kmeans is Spatial-Kmeans and S-PCA is Spatial-PCA.

Bibliography

[1] Martino Pesaresi and Jon Atli Benediktsson. A new approach for the morphological segmentation of high-resolution satellite imagery. *IEEE Transactions on Geoscience and Remote Sensing*, 39(2):309–320, 2001.

[2] Mauro Dalla Mura, Jón Atli Benediktsson, Björn Waske, and Lorenzo Bruzzone. Morphological attribute profiles for the analysis of very high resolution images. *IEEE Transactions on Geoscience and Remote Sensing*, 48(10):3747–3762, 2010.

[3] Mauro Dalla Mura, Jon Atli Benediktsson, Björn Waske, and Lorenzo Bruzzone. Extended profiles with morphological attribute filters for the analysis of hyperspectral data. *International Journal of Remote Sensing*, 31(22):5975–5991, 2010.

[4] Gustavo Camps-Valls, Luis Gomez-Chova, Jordi Muñoz-Marí, Joan Vila-Francés, and Javier Calpe-Maravilla. Composite kernels for hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, 3(1):93–97, 2006.

[5] Jun Li, Prashanth Reddy Marpu, Antonio Plaza, José M Bioucas-Dias, and Jon Atli Benediktsson. Generalized composite kernel framework for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 51(9):4816–4829, 2013.

[6] Yicong Zhou, Jiangtao Peng, and C. L Philip Chen. Extreme Learning Machine with Composite Kernels for Hyperspectral Image Classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(6):2351–2360, 2015.

[7] Yuliya Tarabalka, Mathieu Fauvel, Jocelyn Chanussot, and Jón Atli Benediktsson. Svm- and mrf-based method for accurate classification of hyperspectral images. *IEEE Geoscience and Remote Sensing Letters*, 7(4):736–740, 2010.

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

[9] Gabriele Moser and Sebastiano B Serpico. Combining support vector machines and markov random fields in an integrated framework for contextual image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 51(5):2734–2752, 2013.

[10] George S Almasi and Allan Gottlieb. Highly parallel computing. 1988.

[11] David B Kirk and W Hwu Wen-Mei. *Programming massively parallel processors: a hands-on approach*. Morgan Kaufmann, 2016.

[12] Kun Tan, Junpeng Zhang, Qian Du, and Xuesong Wang. Gpu parallel implementation of support vector machines for hyperspectral image classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(10):4647–4656, 2015.

[13] Lucheng Wu, Xiaoming Xie, Wei Li, and Qian Du. Parallel collaborative representation for hyperspectral image classification on gpus. In *Geoscience and Remote Sensing Symposium (IGARSS), 2016 IEEE International*, pages 2438–2441. IEEE, 2016.

[14] Adam Coates, Honglak Lee, and Andrew Y Ng. An analysis of single-layer networks in unsupervised feature learning. *Ann Arbor*, 1001(48109):2, 2010.

[15] Adam Coates and Andrew Y Ng. Learning feature representations with k-means. In *Neural Networks: Tricks of the Trade*, pages 561–580. Springer, 2012.

[16] Lei Shu, Kenneth McIsaac, Gordon R. Osinski, and Raymond Francis. Unsupervised feature learning for autonomous rock image classification. *Computers & Geosciences*, 106:10–17, 2017.

[17] Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[18] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[19] Jonathon Shlens. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*, 2014.

[20] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[21] Giorgos Mountrakis, Jungho Im, and Caesar Ogole. Support vector machines in remote sensing: A review. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(3):247–259, 2011.

[22] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[23] M Baumgardner, L Biehl, and D Landgrebe. 220 band aviris hyperspectral image data set: June 12, 1992 indian pine test site 3. *Purdue University Research Repository*, 2015.

[24] J Richard Landis and Gary G Koch. The measurement of observer agreement for categorical data. *Biometrics*, pages 159–174, 1977.

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

[26] Yuan Yuan, Jianzhe Lin, and Qi Wang. Hyperspectral image classification via multitask joint sparse representation and stepwise mrf optimization. *IEEE Transactions on Cybernetics*, 46(12):2966–2977, 2016.

[27] Jón Atli Benediktsson, Jón Aevar Palmason, and Johannes R Sveinsson. Classification of hyperspectral data from urban areas based on extended morphological profiles. *IEEE Transactions on Geoscience and Remote Sensing*, 43(3):480–491, 2005.

[28] Yushi Chen, Zhouhan Lin, Xing Zhao, Gang Wang, and Yanfeng Gu. Deep learning-based classification of hyperspectral data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(6):2094–2107, 2014.

[29] Wenzhi Zhao and Shihong Du. Spectral–spatial feature extraction for hyperspectral image classification: A dimension reduction and deep learning approach. *IEEE Transactions on Geoscience and Remote Sensing*, 54(8):4544–4554, 2016.

[30] Qi Wang, Zhaotie Meng, and Xuelong Li. Locality adaptive discriminant analysis for spectral–spatial classification of hyperspectral images. *IEEE Geoscience and Remote Sensing Letters*, 14(11):2077–2081, 2017.

[31] Xiangyong Cao, Lin Xu, Deyu Meng, Qian Zhao, and Zongben Xu. Integration of 3-dimensional discrete wavelet transform and markov random field for hyperspectral image classification. *Neurocomputing*, 226:90–100, 2017.

[32] Le Sun, Zebin Wu, Jianjun Liu, Liang Xiao, and Zhihui Wei. Supervised spectral–spatial hyperspectral image classification with weighted markov random fields. *IEEE Transactions on Geoscience and Remote Sensing*, 53(3):1490–1503, 2015.

Chapter 5

Hyperspectral Image Classification with Stacking Spectral Patches and Convolutional Neural Networks

Published as Shu, L., McIsaac, K., & Osinski, G. R. in IEEE Transactions on Geoscience and Remote Sensing, 2018(99), 1-10. ©2018 IEEE.

5.1 Introduction

Previous methods for hyperspectral image classification typically consist of two separated parts - modeling spatial-spectral feature representation and conducting classification. In this chapter, we discuss how to use the neural network as an end-to-end method to classify hyperspectral image. In the past few years, the techniques of neural networks (e.g., convolutional neural networks) have been dramatically developed [1, 2, 3]. Researchers have been focusing on utilizing neural networks as the end-to-end methods to automatically model the feature representation and conduct classification as well.

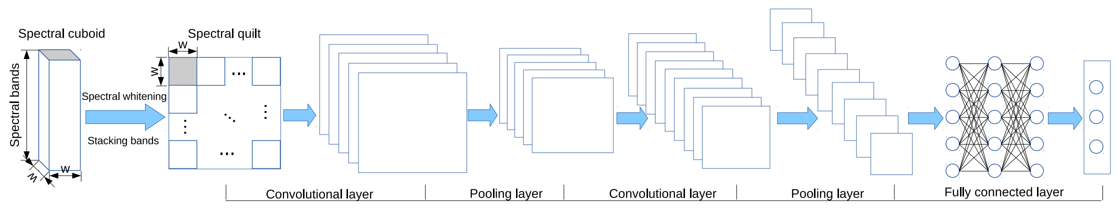
In the area of hyperspectral image classification, there have been some attempts of applying convolutional neural networks, such as [4, 5, 6]. However, those methods require compression of hyperspectral image with dimensionality reduction methods (e.g., PCA), which will inevitably cast away a certain amount of useful spectral information. Although some methods apply no dimensionality reduction, such as [7, 8], they either have ended up with a relatively large amount of network parameters or have increased the convolutional computation.

In order to retain all the spectral information and simultaneously reduce the computational complexity, this chapter proposes an innovative strategy to compress the data dimensionality. In the proposed approach, we stack the spectral patches to form a 1-channel spectral quilt. Subsequently, two shallow neural networks are proposed for classification.

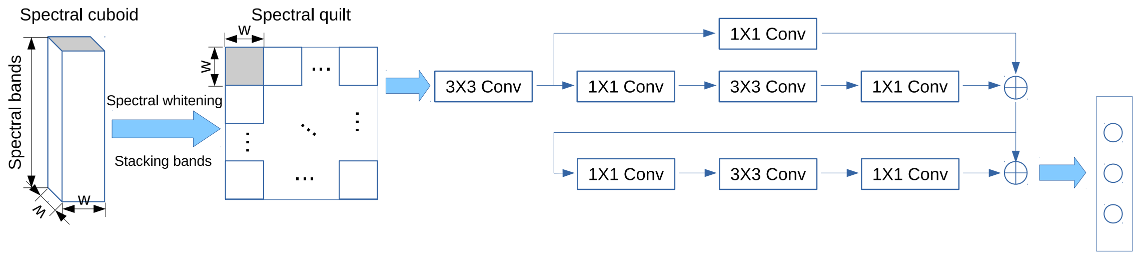
The chapter is organized as follows. Section 5.2 presents the proposed framework for preprocessing HSI as well as two neural networks. Section 5.3 presents the experimental results. Section 5.4 presents some concluding remarks.

5.2 Method

Fig. 5.1 illustrates the proposed framework of hyperspectral image classification, including the preprocessing strategy and two neural networks. In order for the neural network to classify the hyperspectral image, the neighbor region within a certain patch size W of each pixel is extracted as a training sample. Here, we call the training sample a *spectral cuboid*. The spectral cuboid is first preprocessed by PCA whitening, then all the spectral patches are stacked to form a spectral quilt. The spectral quilt is then input to two neural networks for classification. CNN-1 (Fig. 5.1.a) is inspired by VGG-16, a conventional convolutional neural network [3], while CNN-2 (Fig. 5.1.b) is inspired by ResNet which adds a shortcut path to the hidden layers [9]. The details of the networks will be discussed in Section 5.2.2.



(a)



(b)

Figure 5.1: Framework of hyperspectral image classification. (a) CNN-1, (b) CNN-2.

5.2.1 Preprocessing hypespectral data

5.2.1.1 PCA whitening

The raw hyperspectral data is largely redundant, since adjacent spectral bands are highly correlated. One useful preprocessing strategy to reduce the correlation is by whitening (also called "sphering"). Whitening rotates and rescales the data to reduce correlations among features and ensures that features all have the same variance [10]. A simple choice of whitening transform is the PCA whitening transform. The full routine of PCA whitening is summarized here:

a. Center and standardize the data: $X := \frac{X - mean(X)}{\sqrt{var(X)}}$. X is an $N \times B$ matrix, where N is the number of spectral pixels and B is the number of spectral bands.

b. Compute the covariance matrix: $\Sigma = \frac{1}{N} X^T X$.

c. Calculate eigenvectors using singular value decomposition: $[U, S, V] = svd(\Sigma)$. Columns of U are eigenvectors.

d. Rotate the data with eigenvectors: $x_i := U^T * x_i$. This step reduces the correlation between features.

e. Rescale each feature by $1/\sqrt{\lambda_i + \epsilon}$ to make it have unit variance, where λ_i an eigenvalue of the covariance matrix. A small constant ϵ is added in case the eigenvalues are numerically close to 0.

Fig. 5.2 illustrates the process and effect of PCA whitening. We see that after rotating and scaling, the data is less correlated and both features have the same variance.

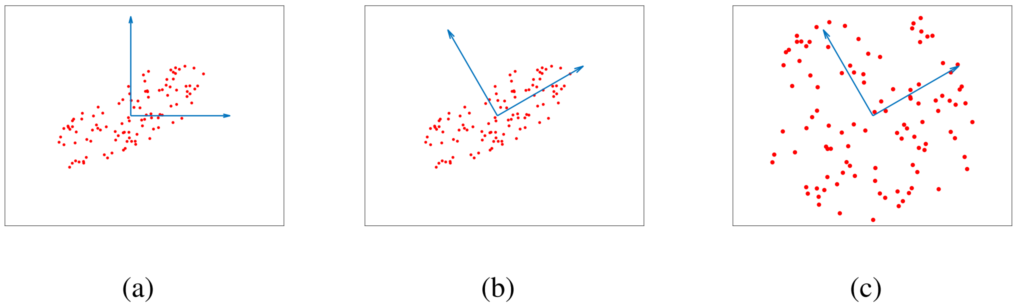


Figure 5.2: Process and effect of PCA whitening. (a) unwhitened data, where the two features are clearly correlated with each other. (b) rotate the data to reduce the correlation. (c) whitened data, where features are less correlated and both have the same variance.

5.2.1.2 Stacking spectral patches

Hyperspectral images usually have hundreds of spectral bands and these spectral bands are redundant. Conventional methods either conduct band selection [11, 12] or compress the spectral bands with a dimensionality reduction method [13, 14] to reduce the spectral dimensionality. However, such methods will inevitably cast away useful spectral information. In order to retain all the spectral bands without increasing the computational cost, we stack together the spectral patches of each spectral cuboid to form a spectral quilt with one single color channel. As long as the spectral patches of all the spectral cuboids are stacked in the same order consistently, the spectral quilt will not only retain the spatial and spectral features in each original band, but also incorporate novel textural patterns between the adjacent spectral bands, which will be useful to separate classes.

The spectral band patches are stacked in such a way that the number of vertical and horizontal patches is identical. Hence, if the number of the original spectral bands is not the square of an integer, we augment the number to the closest perfect square by interpolating a set of spectral bands at random spectral wavelengths. To the best of our knowledge, this is the first proposal for stacking spectral patches to consistently rebuild the spectral features of hyperspectral images.

Fig. 5.3 shows the benefits of stacking spectral patches. Fig. 5.3.a lists three ways of convolutional operation in the first layer of CNN. They are 1) 2-D convolution with the number of channels of convolution kernel matching the number of spectral bands, 2) 3-D convolution with fewer channels of convolution kernel and 3) 2-D convolution on spectral quilts. Due to the mechanism of the convolutional layer in CNN, there will be f^2B parameters for conventional 2-D convolution in one single filter in the first layer. The parameter sets are large because hyperspectral images typically have hundreds of spectral bands. 3-D convolution will have fewer parameters as long as the size of the kernel is small, but it will have f times more multiplications and additions. With spectral quilts, the 2-D convolution has the fewest parameters (only f^2) and the computational complexity remains the same. Table 5.1 shows the detailed numbers.

Above we have shown that 2-D convolution on spectral quilts results in fewer parameters in the first convolutional layer. However, the total number of parameters of a CNN also depends on the other layers especially the fully connected layers if there are. Concretely, if the spatial size of the feature maps adjacently ahead of the fully connected layers were large, the network would end up with a large number of parameters (most of them may come from the fully connected layers). Given that the spectral quilt has bigger spatial size than the original spectral cuboid, we can downsample the feature maps with multiple pooling layers to make the spatial size small before the fully connected layers. For simplicity, suppose the number of spectral bands $B = 2^{2n}$, each convolution kernel size is f, each convolutional layer only has one kernel and is followed by a pooling layer with stride 2×2. The spatial size of spectral quilt will be $2^n w \times 2^n w$. To get a feature map with the same spatial size (i.e., $w \times w$) as the one got from convoluting the original spectral cuboid, we need n pairs of alternating convolutional and pooling layers. While the convolutional layer on the spectral cuboid has $2^{2n} f^2$ parameters, the group of convolutional layers on the spectral quilt only has nf^2 parameters. Here, we have just proved that the spectral quilt can lead to an overall smaller sized neural network (in terms of the overall number of parameters) but with richer architecture (more convolutional layers).

Table 5.1: The number of parameters and computational complexity for three ways of convolutional operation (with "same" padding) for one filter in the first layer of CNN illustrated in Fig. 5.3.a. SQ is short for spectral quilts.

	# of parameters	computational complexity
conventional 2-D conv.	f^2B	w^2f^2B
3-D conv.	f^3	w^2f^3B
2-D conv. with SQ	f^2	w^2f^2B

Fig. 5.3.b indicates that new textural patterns (in red squares) can be created between the adjacent spectral bands by stacking spectral patches. The spatial and spectral features for the same class are similar to some extent, then as long as the spectral patches of all the spectral cuboids are stacked in the same order, the new textural patterns will reappear in the spectral quilts for the same class consistently. Such new textural patterns are effective at separating classes.

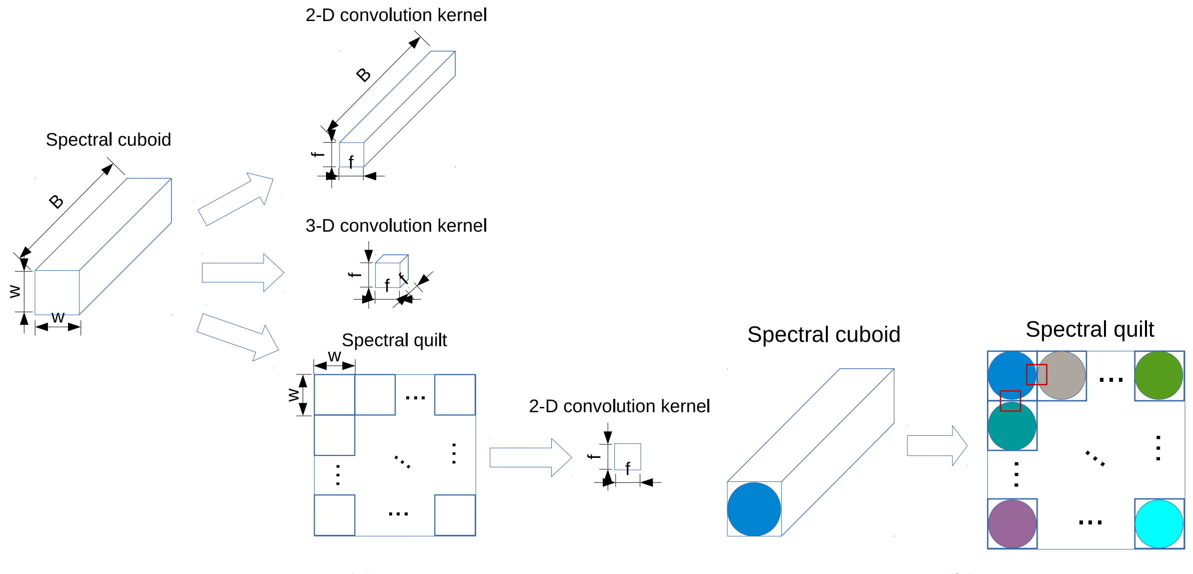


Figure 5.3: The benefits of stacking spectral patches. (a) three ways of convolutional operation in the first layer of CNN, namely 1) 2-D convolution with the number of channels of convolution kernel matching the number of spectral bands, 2) 3-D convolution with fewer channels of convolution kernel and 3) 2-D convolution on spectral quilts. Table 5.1 shows that the proposed *stacking spectral patches* has fewer convolution parameters and is computationally efficient. (b) new patterns (see the red squares) can be found in the spectral quilt and are effective at separating classes.

5.2.2 Convolutional neural networks

5.2.2.1 CNN-1

The network architecture of CNN-1 is similar to VGG-16 [3], but has much fewer layers. Table 5.2 illustrates the detailed configuration. It consists of four convolutional layers and two fully-connected layers. The second and fourth convolutional layers are followed by max-pooling layers which select the max values within the receptive fields to downsample the feature maps. Two fully-connected layers, each with 1024 nodes, are followed by a softmax classifier. In order to prevent overfitting during training, we apply Dropout [15] to some hidden layers, especially the fully-connected layers. Dropout randomly shuts down the neurons with certain probability during each training step. In this way, the network cannot rely on any one specific feature and is forced to spread out the weights. This will tend to have the effect of shrinking the weights and thus regularizes the network.

Table 5.2: Architecture of CNN-1.

Layer No.	Name	Configuration	Stride
1	Conv. + Relu	$3 \times 3 \times 32$	(1,1)
2	Conv. + Relu	$3 \times 3 \times 64$	(1,1)
3	Max-pooling	2×2	(2,2)
4	Dropout	25%	N/A
5	Conv. + Relu	$3 \times 3 \times 128$	(1,1)
6	Conv. + Relu	$3 \times 3 \times 128$	(1,1)
7	Max-pooling	2×2	(2,2)
8	Dropout	25%	N/A
9	Fully-connected	1024	N/A
10	Dropout	50%	N/A
11	Fully-connected	1024	N/A
12	Dropout	50%	N/A
13	Softmax	# of classes	N/A

5.2.2.2 CNN-2

The detailed architecture of CNN-2 is illustrated in Table 5.3. It has the same residual block as ResNet [9]. For the conventional CNN, such as CNN-1 described before, the speed of learning usually decreases very rapidly for the early layers as the network trains. This is the so-called *vanishing gradient problem*. However, by adding a shortcut to the main path of the original convolutional network, the residual block is better able to learn the identity function. Thus the network with shortcut connections overall trains fast enough without the vanishing gradient problem [9].

Table 5.3: Architecture of CNN-2. Note that *.out stands for the output of *th layer. Two bold layers are the shortcuts in residual blocks.

Layer No.	Name	Configureation	Stride
1	Conv. + Relu	$3 \times 3 \times 64$	(2,2)
2	Max-pooling	3×3	(2,2)
3	Conv. + Relu	$1 \times 1 \times 64$	(1,1)
4	Conv. + Relu	$3 \times 3 \times 64$	(1,1)
5	Conv.	$1 \times 1 \times 256$	(1,1)
6	2.out $\rightarrow$ Conv.	$1 \times 1 \times 256$	(1,1)
7	**Add(5.out, 6.out) + Relu**	N/A	N/A
8	Conv. + Relu	$1 \times 1 \times 64$	(1,1)
9	Conv. + Relu	$3 \times 3 \times 64$	(3,3)
10	Conv.	$1 \times 1 \times 256$	(1,1)
11	**Add(10.out, 7.out) + Relu**	N/A	N/A
12	Average-pooling	3×3	N/A
13	Softmax	# of classes	N/A

5.3 Experiments and results

5.3.1 Hyperspectral datasets

The performance of our proposed methods was evaluated on two public datasets: Indian Pines and Pavia University.

5.3.1.1 Indian pines

The hyperspectral data of Indian pines was acquired by Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) to support soils research in 1992 over Purdue University Agronomy farm northwest of West Lafayette and the surrounding area in Indiana, USA [16]. The area is a mixed agricultural/forest area, where some of the crops present, corn, soybeans, were in early stages of growth.

Fig. 5.4a shows a false-color composition of the AVIRIS Indian Pines scene. This dataset consists of 145×145 pixels covering a 2 mile by 2 mile area at 20 m spatial resolution and 224 spectral bands covering the spectral range from 400 nm to 2.5 μm. Several spectral bands were removed from the dataset due to noise and water absorption phenomena, leaving a total of 200 bands to be used in the experiments. The ground-truth of the Indian Pines scene has been designated into sixteen classes (see Fig. 5.4b). Table 5.4 shows the specific classes and the respective number of samples.

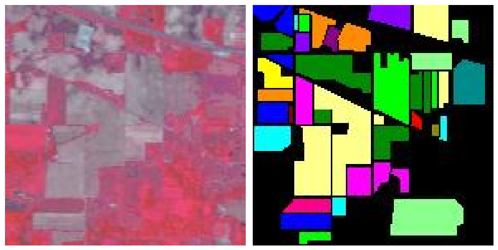

(a) (b)

Figure 5.4: Indian Pines dataset. (a) False-color composition of bands 50, 27 and 17. (b) Ground-truth map. The specific classes denoted by different colors refer to Table 5.4.

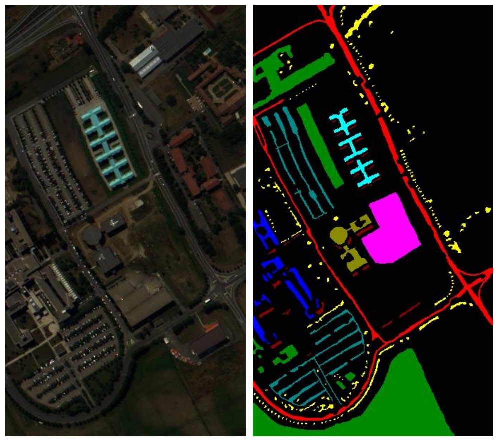
5.3.1.2 Pavia University

The hyperspectral data of Pavia University was acquired by Reflective Optics System Imaging Spectrometer (ROSIS) during a flight campaign over Pavia, northern Italy. Figure 5.5 illustrates the true-color image of Pavia University. It contains 610×340 pixels and the geometric resolution is 1.3 m. There are 103 spectral bands covering the spectral range from 430 to 860 nm. The ground-truth map differentiates 9 classes as shown in Table 5.5.

From both Table 5.4 and Table 5.5 we see that both datasets have unbalanced numbers of labelled samples per class. To investigate the performance of the proposed methods, we randomly select 10% of the labelled samples per class for training (minimum 10 samples for the classes with extremely limited numbers of samples). Cross-validation is conducted with train-

Table 5.4: Classes for the Indian Pines scene and their respective number of samples.

#	**Class**	**Num. of samples**
🟥 1	Alfalfa	46
🟩 2	Corn-notill	1428
🟦 3	Corn-mintill	830
🟨 4	Corn	237
5	Grass-pasture	483
6	Grass-trees	730
7	Grass-pasture-mowed	28
8	Hay-windrowed	478
9	Oats	20
10	Soybean-notill	972
11	Soybean-mintill	2455
🟧 12	Soybean-clean	593
13	Wheat	205
14	Woods	1265
🟪 15	Buildings-Grass-Trees-Drives	386
16	Stone-Steel-Towers	93
	Total	10249



(a) (b)

Figure 5.5: Pavia University scene. (a) True-color composition of bands 53, 31 and 8. (b) Ground-truth map. The specific classes denoted by different colors refer to Table 6.2.

ing data to tune the hyperparameters of the networks, such as learning rate, number of epochs etc. The rest of the labelled samples are used for testing. The classification performance is measured by the overall accuracy (OA), average accuracy (AA) and Kappa coefficient (κ). Cohen's Kappa [17] is a metric that measures how much better the classifier is performing over the performance of a classifier that simply guesses at random according to the frequency of each

Table 5.5: Classes for the Pavia University scene and their respective number of samples.

#	**Class**	**Num. of samples**
🟥 1	Asphalt	6631
🟩 2	Meadows	18649
🟦 3	Gravel	2099
🟨 4	Trees	3064
🟦 5	Painted metal sheets	1345
🟪 6	Bare Soil	5029
🟫 7	Bitumen	1330
🟦 8	Self-Blocking Bricks	3682
🟥 9	Shadows	947
	Total	42776

Table 5.6: Image size after stacking spectral patches over different patch size.

patch size	7	9	11	13	15
Indian Pines	105×105	135×135	165×165	195×195	225×225
Pavia University	77×77	99×99	121×121	143×143	165×165

class. It is considered a good metric for imbalanced and multi-class classification problems. A higher Kappa score indicates better performance. Since the performance varies along with variation of our selection of training samples, the mean and standard deviation of OA, AA and κ over 10 Monte Carlo runs are recorded for evaluating the performance.

5.3.1.3 Stacking spectral patches

Indian Pines dataset has 200 spectral bands available after eliminating some bands due to noise and water absorption phenomena. As discussed above, we augment the data set by interpolating 25 bands at randomly chosen positions. After interpolation, we have 225 bands in total. If we set the spectral patch size to 7, for example, we will get the grayscale images with size of $(15 * 7) \times (15 * 7)$ after SSP. Similarly, we interpolate 18 spectral bands into the Pavia University dataset to bring its 103 bands to 121. If the spectral patch size is 7, the image size will be $(11 * 7) \times (11 * 7)$. Table 5.6 illustrates the image size after SSP over different options of patch size for both datasets. After stacking spectral patches to generate the spectral quilts, the spectral quilts are input to the neural networks for classification.

5.3.2 Analysis of PCA whitening

The spectral bands are highly correlated in hyperspectral images. We use the Pearson correlation coefficient [18] to evaluate the correlation between the spectral bands. Figure 5.6 depicts the normalized distribution of pairwise Pearson correlation coefficient between each pair of spectral bands as well as the corresponding Gaussian kernel density estimate. We see that while the spectral bands of both Indian Pines and Pavia University datasets are highly correlated, the Pavia University dataset is less correlated than Indian Pines since it has a larger number of correlation coefficients close to zero.

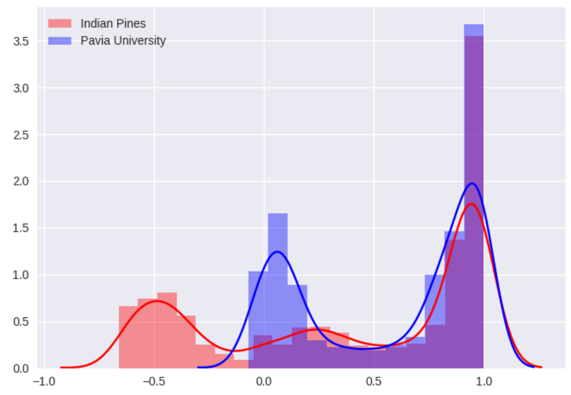
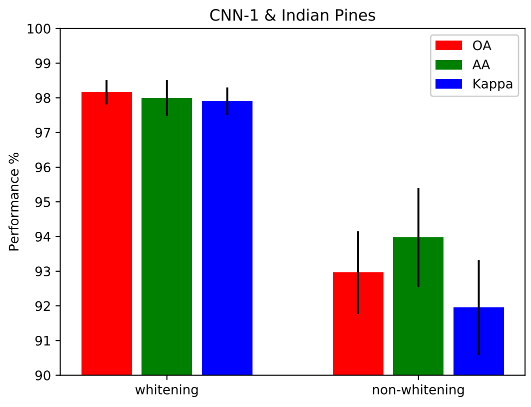
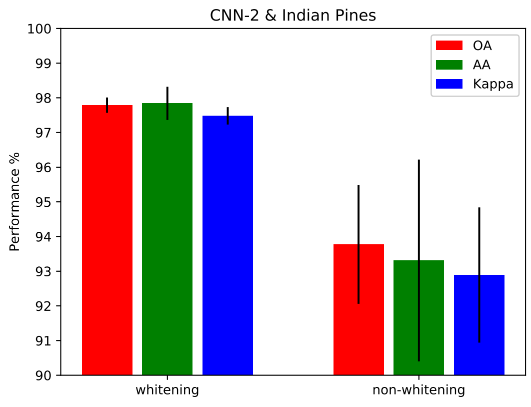


Figure 5.6: Normalized pairwise correlation distribution of Indian Pines and Pavia University scene.

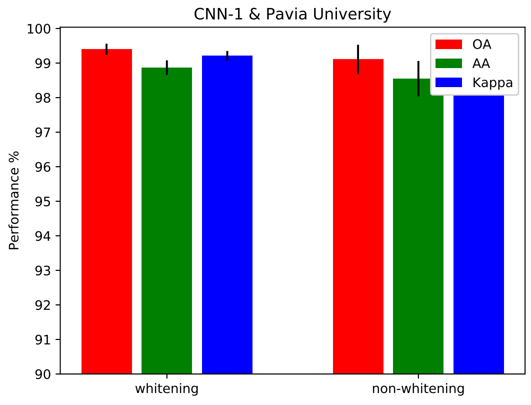
We compared the performance with PCA whitening and without PCA whitening for both datasets. The results, shown in Table 5.7 and Fig. 5.7, demonstrate that PCA whitening has a dramatic effect on the Indian Pines dataset but only a slight effect on Pavia University dataset. Fig. 5.7 (a) and (b) illustrate the performance for the Indian Pines dataset with CNN-1 and CNN-2 respectively. The height of the bars stands for the averaged classification accuracy and the vertical lines on the top stand for the standard deviation through the 10 Monto Carlo runs. We see that PCA whitening has a salient boost of performance for both networks in Indian pines dataset (Fig. 5.7 (a) and (b)). However, the enhancement of performance for Pavia University dataset is quite small (Fig. 5.7 (c) and (d)). We believe that is because Pavia University dataset is less correlated than Indian Pines as shown in Figure 5.6. Hence, PCA whitening is effective as a preprocessing step for hyperspectral image classification and the effectiveness



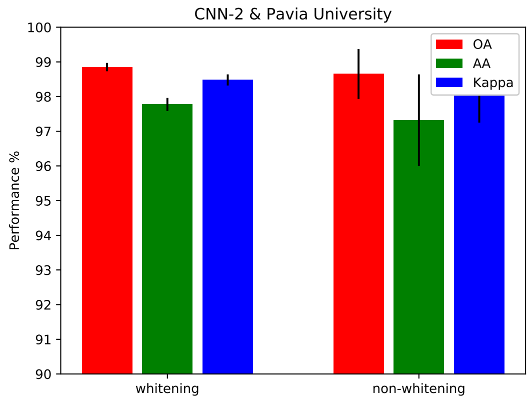
(a)



(b)



(c)



(d)

Figure 5.7: Performance with PCA whitening and without PCA whitening. (a) CNN-1 & Indian Pines, (b) CNN-2 & Indian Pines, (c) CNN-1 & Pavia University, (d) CNN-2 & Pavia University.

increases with increased prior correlation.

Table 5.7: Performance with PCA whitening and without PCA whitening. The patch size is set to 15. *Oui* stands for with PCA whitening and *Non* stands for without PCA whitening.

		Indian Pines		Pavia University	
		Oui	Non	Oui	Non
CNN-1	OA	98.16 ± 0.35	92.96 ± 1.19	99.40 ± 0.16	99.11 ± 0.42
	AA	97.99 ± 0.52	93.97 ± 1.43	98.87 ± 0.21	98.55 ± 0.51
	$\kappa \times 100$	97.90 ± 0.40	91.95 ± 1.37	99.21 ± 0.14	98.82 ± 0.56
CNN-2	OA	97.79 ± 0.22	93.77 ± 1.71	98.85 ± 0.12	98.65 ± 0.72
	AA	97.84 ± 0.48	93.31 ± 2.91	97.77 ± 0.19	97.32 ± 1.32
	$\kappa \times 100$	97.48 ± 0.25	92.89 ± 1.95	98.48 ± 0.16	98.21 ± 0.96

Table 5.8: Performance over different patch sizes of spectral cuboid.

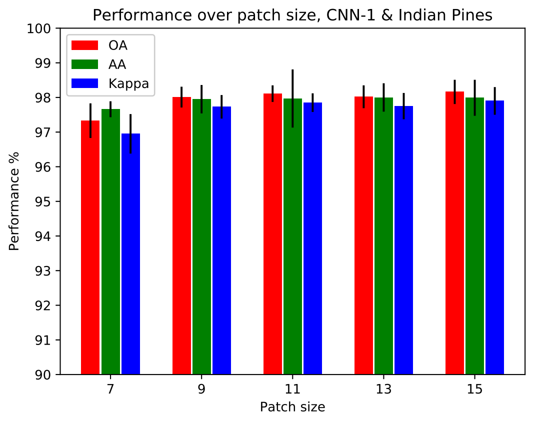
Patch size			7	9	11	13	15
Indian Pines	CNN-1	OA	97.33 ± 0.50	98.01 ± 0.30	98.11 ± 0.24	98.02 ± 0.33	98.16 ± 0.35
		AA	97.66 ± 0.23	97.95 ± 0.41	97.97 ± 0.84	98.00 ± 0.41	97.99 ± 0.52
		$\kappa \times 100$	96.95 ± 0.57	97.73 ± 0.34	97.85 ± 0.27	97.75 ± 0.38	97.90 ± 0.40
	CNN-2	OA	93.33 ± 0.41	96.43 ± 0.25	97.15 ± 0.44	97.71 ± 0.31	97.79 ± 0.22
		AA	94.26 ± 0.67	96.58 ± 0.63	97.25 ± 0.52	97.40 ± 0.44	97.84 ± 0.48
		$\kappa \times 100$	92.38 ± 0.47	95.93 ± 0.29	96.75 ± 0.51	97.38 ± 0.35	97.48 ± 0.25
Pavia University	CNN-1	OA	98.69 ± 0.21	99.04 ± 0.14	99.32 ± 0.13	99.38 ± 0.21	99.40 ± 0.16
		AA	97.93 ± 0.24	98.43 ± 0.24	98.72 ± 0.20	98.80 ± 0.30	98.87 ± 0.21
		$\kappa \times 100$	98.26 ± 0.28	98.72 ± 0.19	99.10 ± 0.17	99.18 ± 0.13	99.21 ± 0.14
	CNN-2	OA	96.90 ± 0.62	97.74 ± 0.26	98.15 ± 0.16	98.50 ± 0.19	98.85 ± 0.12
		AA	95.54 ± 0.65	96.43 ± 0.48	96.82 ± 0.28	97.26 ± 0.33	97.77 ± 0.19
		$\kappa \times 100$	95.89 ± 0.81	97.00 ± 0.34	97.55 ± 0.21	98.01 ± 0.25	98.48 ± 0.16

5.3.3 Analysis of patch size

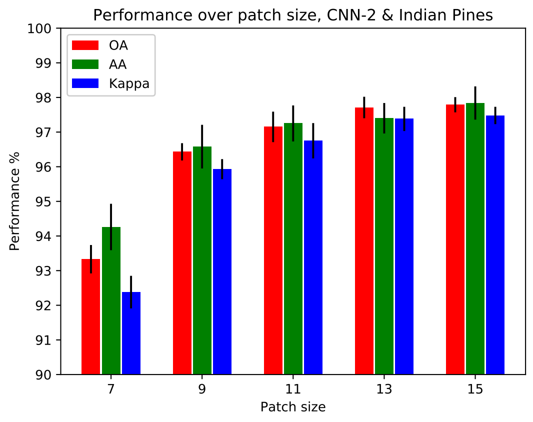
Both CNN-1 and CNN-2 require the patch size of input spectral cuboids to be specified before preprocessing. Therefore we conducted experiments to evaluate how the patch size affects the performance. As shown in Table 5.8 and Figure 5.8, the performance generally improves as the patch size increases. This is what we would expect because larger patches are better able to capture more complex features that cover a larger region of the images. Note that CNN-1 works well even with a small patch size. Thus, as the patch size increases, the improvement of performance of CNN-1 is quite limited. Whereas CNN-2's performance is more correlated with patch size. Its improvement with a larger patch size is remarkable.

5.3.4 Performance comparison

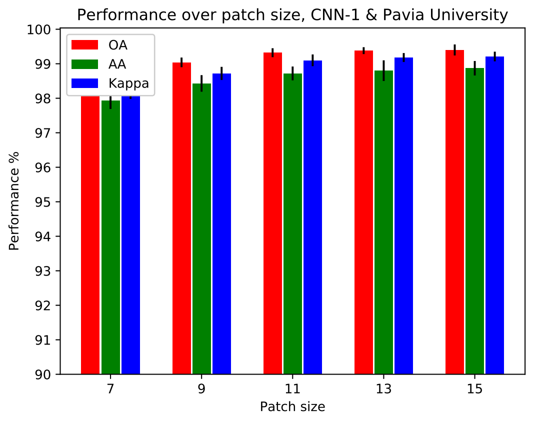
Though CNN-1 has only four convolutional layers, it has many more trainable parameters than CNN-2 (Table 5.9 lists the number of trainable parameters in CNN-1 and CNN-2 when patch size is set to 7) because of the two fully-connected layers each with 1024 nodes in CNN-1. Hence it is harder to train CNN-1. We add Dropout layers into CNN-1 to prevent the overfitting problem. After carefully fine-tuning the hyperparameters, CNN-1 can work better than CNN-2. But its training loss converges slower than the one of CNN-2 (Fig. 5.9) because of the large amount of trainable parameters and the fact that Dropout layers have added noise into the network.



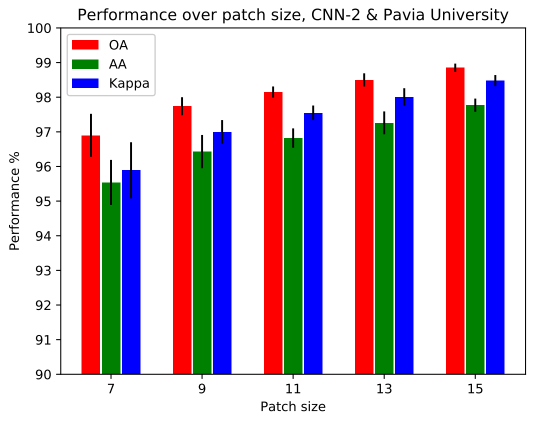
(a)



(b)



(c)

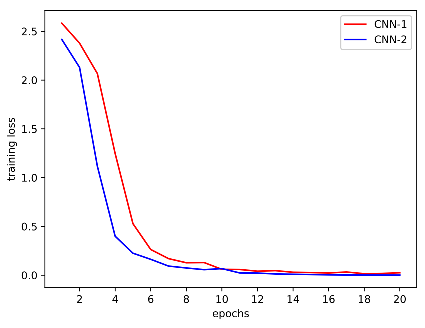


(d)

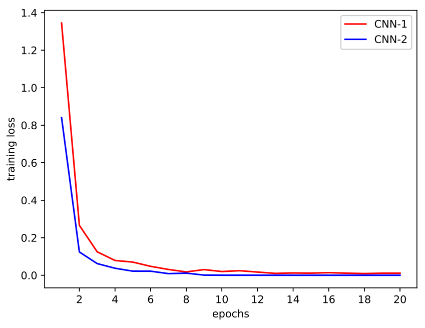
Figure 5.8: Performance over different patch sizes of spectral cuboid. (a) CNN-1 & Indian Pines, (b) CNN-2 & Indian Pines, (c) CNN-1 & Pavia University, (d) CNN-2 & Pavia University.

Table 5.9: The number of trainable parameters in CNN-1 and CNN-2 when patch size is set to 7.

	Indian Pines	Pavia University
CNN-1	70,644,368	34,854,537
CNN-2	476,816	227,977



(a)



(b)

Figure 5.9: Training rate over the epochs for two networks. (a) Indian Pines, (b) Pavia University. For both datasets, the training loss of CNN-2 converges faster than the one of CNN-1.

As shown in Table 5.8, CNN-1 works consistently better than CNN-2 over different patch sizes of spectral cuboids for both Indian pines and Pavia University datasets. CNN-1's performance is less dependent on the patch size, which is to say that it can perform reasonably well even with a small patch size. This is because CNN-1 has more parameters so that it is more expressive than CNN-2 and can extract more complicated features. The price is that techniques such as Dropout are required to prevent overfitting during training which adds more hyperparameters to tune and increases training time. While CNN-1 performs better, CNN-2 is much lighter and is faster to train. Fig. 5.8 indicates that the gap of performance between the two networks shrinks as the patch size increases (for example, the gap of the overall accuracy is decreasing from 4% to 0.3% in Indian Pines scene as shown in Table 5.8). Hence, CNN-2 is still competitive with a relatively large patch size of the spectral cuboids.

We have also compared the performance of the proposed methods with seven other state-of-the-art methods. They are EMAP [19], GCK [20], SMLR-SpATV [21], 3DDWT [22], Chen2016 [23], Lee2017 [8], Cao2017 [24]. Lee2017 is a fully convolutional network consisting mostly of 1×1 convolutional layers and as is claimed in [8] patch size 5 is the best for Lee2017's network architecture. Chen2016 is a shallow 2-D CNN which takes as input the cropped patches from the first principal spectral component. Cao2017 is also a CNN which takes as input the original spectral cuboids. For Cao2017, CNN-1 and CNN-2, the patch size of cropped spectral cuboids is set to 15. For Chen2016, the patch size is set to 27 as is fixed in [23]. For Lee2017, the patch size is set to 5 as is also fixed in [8]. As with previous experiments, 10% training samples are randomly selected for 10 Monto Carlo runs. All the other corresponding parameter setting is the same as stated in the references.

Table 5.10 indicates the performance of all the methods. Among these 9 methods, CNN-1 performs the best for both datasets (the performance gap between CNN-1 and the compared methods is around 0.4%~8% for Indian Pines and 0.2%~3% for Pavia University). CNN-2 is very close to SMLR-SpATV but remarkably outperforms EMAP, GCK and Lee2017. Although 3DDWT, Chen2016 and Cao2017 all slightly outperform CNN-2 in Pavia University scene (+0.2%~+0.4%), CNN-2 dramatically outperforms these three methods in Indian Pines scene (+1%~+3%). That indicates CNN-2 is relatively more robust when there are fewer training samples.

Table 5.10: Performance comparison with different methods - EMAP [19], GCK [20], SMLR-SpATV [21], 3DDWT [22], Chen2016 [23], Lee2017 [8], Cao2017 [24], CNN-1 and CNN-2.

		EMAP	GCK	SMLR-SpATV	3DDWT	Chen2016	Lee2017	Cao2017	CNN-1	CNN-2
Indian Pines	OA	92.78 ± 0.70	94.81 ± 0.60	97.80 ± 0.31	95.16 ± 0.79	94.51 ± 0.59	90.63 ± 1.40	96.79 ± 0.51	98.16 ± 0.35	97.79 ± 0.22
	AA	93.73 ± 1.02	95.41 ± 0.96	97.23 ± 0.62	93.59 ± 1.57	93.81 ± 1.19	90.01 ± 2.00	95.05 ± 0.70	97.99 ± 0.52	97.84 ± 0.48
	$\kappa \times 100$	92.35 ± 0.72	94.94 ± 0.80	97.49 ± 0.36	94.47 ± 0.90	93.73 ± 0.67	89.30 ± 1.61	96.34 ± 0.58	97.90 ± 0.40	97.48 ± 0.25
Pavia University	OA	97.94 ± 0.34	98.32 ± 0.25	98.73 ± 0.14	99.05 ± 0.11	99.23 ± 0.17	96.26 ± 0.72	99.05 ± 0.15	99.40 ± 0.16	98.85 ± 0.12
	AA	97.04 ± 0.45	97.46 ± 0.38	97.59 ± 0.28	98.07 ± 0.23	98.37 ± 0.41	94.15 ± 0.72	98.33 ± 0.28	98.87 ± 0.21	97.77 ± 0.19
	$\kappa \times 100$	96.98 ± 0.32	97.69 ± 0.43	98.32 ± 0.19	98.75 ± 0.15	98.97 ± 0.22	95.04 ± 0.96	98.74 ± 0.20	99.21 ± 0.14	98.48 ± 0.16

5.4 Conclusion

This chapter proposes a simple but innovative framework to classify hyperspectral image with two shallow convolutional neural networks. First, *principal component analysis (PCA) whitening* is applied to decorrelate hundreds of spectral bands. Instead of selecting the principal components to reduce the spectral dimensionality, we retain all the spectral bands but compress the image cuboid into 1-channel spectral quilt by *stacking spectral patches*. In this way, not only is all the spectral information retained, but also the computational complexity of training a neural network is reduced compared with conventional networks which directly input the spectral cuboids. Moreover, the spectral quilts will contain some novel textural patterns which are effective at separating classes.

Two shallow convolutional neural networks (CNN-1 and CNN-2) are then applied to classify the spectral quilts. As shown in the experiments, CNN-1 works better than CNN-2 since it has a larger set of trainable parameters, making it more expressive and better able to extract more complicated features. The downside of CNN-1 is that the large parameter set increases the risk of overfitting and causes the requirement for Dropout to regularize the network. While CNN-1 has higher accuracy, CNN-2 is much lighter and is faster to train. As long as the patch size of spectral cuboids is large enough, CNN-2 is still a powerful competitor. The performance of the proposed methods is compared with seven other state-of-the-art methods. The experimental results show that CNN-1 can remarkably outperform all the other state-of-the-art methods and CNN-2 can outperform most of the methods and appears more robust.

The future work will include testing the methodology on more hyperspectral images to classify the land cover and study the geosciences of planetary surface. An extended application can be classifying multispectral images which have less spectral information but more detailed spatial contextual information.

Bibliography

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[2] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833. Springer, 2014.

[3] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[4] Konstantinos Makantasis, Konstantinos Karantzalos, Anastasios Doulamis, and Nikolaos Doulamis. Deep Supervised Learning for Hyperspectral Data Classification through Convolutional Neural Networks. *IGARSS 2015. 2015 IEEE International Geoscience and Remote Sensing Symposium. Proceedings*, pages 4959–4962, 2015.

[5] Wenzhi Zhao and Shihong Du. Learning multiscale and deep representations for classifying remotely sensed imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 113:155–165, 2016.

[6] Wenzhi Zhao and Shihong Du. Spectral Spatial Feature Extraction for Hyperspectral Image Classification : A Dimension Reduction and Deep Learning Approach. *IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING*, 54(8):4544–4554, 2016.

[7] Hyungtae Lee and Heesung Kwon. Contextual Deep CNN Based Hyperspectral Classification. *arXiv*, 1604.03519:2–4, 2016.

[8] Hyungtae Lee and Heesung Kwon. Going deeper with contextual cnn for hyperspectral image classification. *IEEE Transactions on Image Processing*, 26(10):4843–4855, 2017.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Patrik O. Hoyer Aapo Hyvrinen, Jarmo Hurri. *Natural Image Statistics*. Springer, 2009.

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

[12] Yuan Yuan, Jianzhe Lin, and Qi Wang. Hyperspectral image classification via multitask joint sparse representation and stepwise mrf optimization. *IEEE Transactions on Cybernetics*, 46(12):2966–2977, 2016.

[13] Yushi Chen, Xing Zhao, and Xiuping Jia. Spectral–spatial classification of hyperspectral data based on deep belief network. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(6):2381–2392, 2015.

[14] Qi Wang, Zhaotie Meng, and Xuelong Li. Locality adaptive discriminant analysis for spectral–spatial classification of hyperspectral images. *IEEE Geoscience and Remote Sensing Letters*, 14(11):2077–2081, 2017.

[15] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[16] M Baumgardner, L Biehl, and D Landgrebe. 220 band aviris hyperspectral image data set: June 12, 1992 indian pine test site 3. *Purdue University Research Repository*, 2015.

[17] J Richard Landis and Gary G Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.

[18] Joseph Lee Rodgers and W Alan Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66, 1988.

[19] Mauro Dalla Mura, Jon Atli Benediktsson, Björn Waske, and Lorenzo Bruzzone. Extended profiles with morphological attribute filters for the analysis of hyperspectral data. *International Journal of Remote Sensing*, 31(22):5975–5991, 2010.

[20] Jun Li, Prashanth Reddy Marpu, Antonio Plaza, José M Bioucas-Dias, and Jon Atli Benediktsson. Generalized composite kernel framework for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 51(9):4816–4829, 2013.

[21] Le Sun, Zebin Wu, Jianjun Liu, Liang Xiao, and Zhihui Wei. Supervised spectral–spatial hyperspectral image classification with weighted markov random fields. *IEEE Transactions on Geoscience and Remote Sensing*, 53(3):1490–1503, 2015.

[22] Xiangyong Cao, Lin Xu, Deyu Meng, Qian Zhao, and Zongben Xu. Integration of 3-dimensional discrete wavelet transform and markov random field for hyperspectral image classification. *Neurocomputing*, 226:90–100, 2017.

[23] Yushi Chen, Hanlu Jiang, Chunyang Li, Xiuping Jia, and Pedram Ghamisi. Deep feature extraction and classification of hyperspectral images based on convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 54(10):6232–6251, 2016.

[24] Xiangyong Cao, Feng Zhou, Lin Xu, Deyu Meng, Zongben Xu, and John Paisley. Hyperspectral image segmentation with markov random fields and a convolutional neural network. *arXiv preprint arXiv:1705.00727*, 2017.

Chapter 6

Hyperspectral Image Classification with Combinational Fully Convolutional Network

Submitted for review by IEEE Transactions on Geoscience and Remote Sensing as Shu, L., McIsaac, K., & Osinski, G. R.

6.1 Introduction

Since neural networks provide the capability to automatically extract the spatial-spectral feature representations as well as the capability to flexibly generate higher level features with deeper layers, neural networks have achieved better performance on hyperspectral image classification than conventional methods. Last chapter has described a framework of applying convolutional neural network to hyperspectral image classification. This chapter will continue with another innovative network - combinational fully convolutional network.

Conventional convolutional neural networks consist of alternating convolutional layers and pooling layers and fully connected layers in the end [1]. In order for neural networks to jointly model spectral and spatial contextual information, we need to crop neighbor regions of a certain size as patch samples with a sliding-window and then conduct patchwise classification to compute an entire pipeline for each window of the input one at a time. Intuitively, areas around nearby pixels are duplicated. Larger cropped patch sizes lead to more duplication. The duplication of data will inevitably require more computing resources, i.e., computing time and memory. Usually, larger patches are better able to capture more spatial contextual information which will result in better performance. Therefore, there may need to be a tradeoff between performance and computing resources, especially for the real-time applications.

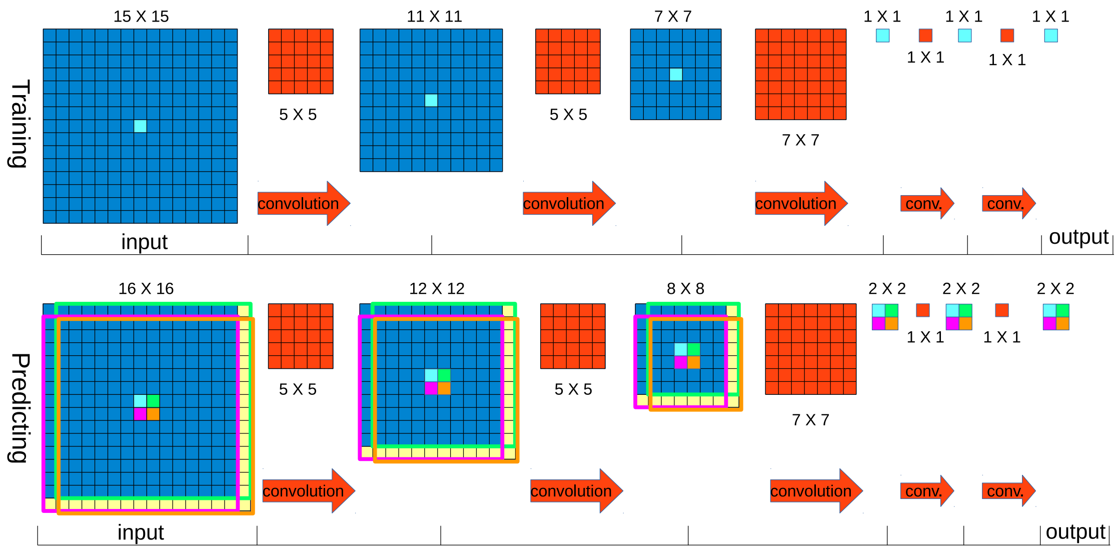


Figure 6.1: The mechanism of FCN and its efficiency for predicting a map. During training, a FCN produces only one single output (top) corresponding to the central pixel in the input patch. When applied at prediction time over a larger patch, it produces a spatial map, e.g. 2×2 map (bottom) corresponding to the central 2×2 pixels in the input patch. Because all layers are applied convolutionally, the extra computation required for the lager patch is limited to the yellow regions. This diagram omits the feature dimension for simplicity.

Convolutional layers take the inner product of the linear filter and the underlying receptive field followed by a nonlinear activation function at every local portion of the input. Clearly, nearby patches on hyperspectral images share a large amount of convolutional computations. Therefore, we propose a fully convolutional network (FCN) [2, 3] for hyperspectral image classification. Sermanet et al. [4] demonstrated the efficiency of a convolutional implementation of the sliding-window for object detection. Similarly, our FCN will also be inherently efficient when applied in a sliding fashion because the computation is highly amortized over the overlapping regions of spectral patches. Fig. 6.1 diagrams the efficiency of FCN for predicting a map. The FCN can take input of arbitrary size, thus we can directly input the whole hyperspectral image to predict the whole map with one single forward-propagation.

Since the early days of neural networks, researchers have arranged an ensemble of neural networks in a simple voting scheme to improve performance [5]. Networks with a collection of many paths can have ensemble-like behavior to some extent. For example, Veit et al. [6] showed that using identity mapping as the skip connections allows the residual network [7] to behave like ensembles of relatively shallow networks. One critical trait of ensembles is that their performance depends smoothly on the number of members. In particular, the performance improvement from additional ensemble members gets smaller with increasing ensemble size. Since ensembles of networks usually show robustness and high performance, our aim is to design a network which is a collection of many shallow paths and shows ensemble behavior while taking advantage of the inherent computational efficiency of fully convolutional networks.

Fully convolutional computation has been exploited for sliding window object detection [4], semantic segmentation [3] and image restoration [8] and has been proved to be efficient and effective in such scenarios. To our knowledge, Lee et al. [9] is the only application of a fully convolutional network on hyperspectral image classification. However, their proposed network can only input patches with limited spatial size (e.g., 5×5). Such patches are too small to incorporate much spatial contextual information and the network's stacked 1×1 convolutions are less able to extract deep features than larger convolution kernels.

In this chapter,we propose a combinational fully convolutional network (CFCN) for hyperspectral image classification. The FCN for typical semantic segmentation contains subsampling and upsampling layers, which make it difficult for the network to capture detailed structures of objects. Hyperspectral image classification places more value on pixelwise accuracy. Thus, our proposed network consists of only convolutional layers. Training is still patchwise, but prediction is performed on the entire image at one time. The detailed characteristics of the CFCN are summarized here:

a. Utilizes a 1×1 convolutional layer to learn overcomplete spectral feature maps which can enhance the performance;

b. Transforms fully connected layers into two 1×1 convolutional layers to enable convolutional implementation of sliding window;

c. The collection of paths demonstrates ensemble behavior to guarantee robust and high performance;

d. No subsampling and upsampling layers, which allows predictions for consecutive pixels and capturing more detailed spatial structures;

e. Takes input of arbitrary size, so given input of the whole hyperspectral image, it can directly predict the whole map.

The rest of the chapter is organized as follows. Section 6.2 presents the proposed combinational fully convolutional network. Section 6.3 describes the experimental results. Section 6.4 presents some concluding remarks.

6.2 Method

In order to take advantage of FCN's computational efficiency, we propose a fully convolutional network for hyperspectral image classification and its architecture is shown in Fig. 6.2. It consists of three 1×1 convolutional layers at the two ends and a set of conjunct 3×3 and/or 5×5 convolutional layers in the middle. The FCN for typical semantic segmentation contains subsampling and upsampling layers, which make it difficult for the network to capture detailed structures of objects. Since hyperspectral image classification places more value on pixelwise accuracy, our proposed network consists of only convolutional layers. For simplicity, Fig. 6.2 shows the architecture of the proposed network for input patch size 15, but the size is definitely not constrained to 15. It is straightforward to adjust network architecture according to the patch size as we will show later.

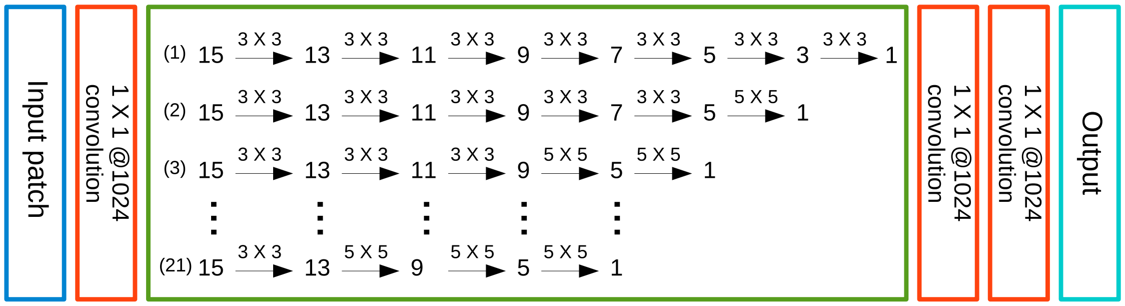

Input patch
1 X 1 @1024 convolution
(1) 15 3 X 3 13 3 X 3 11 3 X 3 9 3 X 3 7 3 X 3 5 3 X 3 3 3 X 3 1
(2) 15 3 X 3 13 3 X 3 11 3 X 3 9 3 X 3 7 3 X 3 5 5 X 5 1
(3) 15 3 X 3 13 3 X 3 11 3 X 3 9 5 X 5 5 5 X 5 1
(21) 15 3 X 3 13 5 X 5 9 5 X 5 5 5 X 5 1
1 X 1 @1024 convolution
1 X 1 @1024 convolution
Output

Figure 6.2: The architecture of the proposed family of FCNs. It consists of three 1×1 convolutional layers at the two ends (inside the red rectangles) and a set of conjunct 3×3 and/or 5×5 convolutional layers in the middle (inside the green rectangle). The first 1×1 convolutional layer is used to learn overcomplete feature maps, which in some sense increases the spectral bands of hyperspectral images and the last two work in the same way as fully connected layers do. Suppose we crop the 15×15 neighbor regions as the training samples for the central pixels. There will be 21 different architectures with sets of conjunct 3×3 and/or 5×5 convolutional layers (with valid padding) to shrink the patch size from 15 down to 1.

6.2.1 1×1 convolutional layers

As shown in Fig. 6.2, there are three 1×1 convolutional layers. The first one, coming right after the input layer, is used to learn overcomplete spectral feature maps, which will increase the number of spectral bands of hyperspectral images in some sense. As we will see in experiments, this 1×1 convolutional layer plays a significant role on the overall network.

The last two 1×1 convolutional layers work in the same way as the fully connected layers in conventional convolutional neural networks. They are the classifier of the network and have exactly the same weights as two fully connected layers do. By using 1×1 convolutional layers instead of fully connected layers, we can input an image of any size into the network at prediction time. Thereafter, one forward-propagation of the whole hyperspectral image can get all the computations for each neighbor patch.

6.2.2 Combinational fully convolutional network

Suppose we crop the 15×15 neighbor regions as the training samples for the central pixels. As shown in Fig. 6.2, we can apply a set of conjunct 3×3 and/or 5×5 convolutional layers (with valid padding) to shrink the patch size down to 1. The resulting outputs from convolutional layers are called feature maps. We only consider 3 and 5 as the convolutional kernel size because smaller kernel sizes are better able to capture fine details of the image. Depending on how to combine the convolutional layers, there will be 21 different network architectures. For example, we can use seven consecutive 3×3 convolutions to form the deepest network, or one 3×3 and two 5×5 convolutions to make the shallowest one. Note that there are no pooling layers here because otherwise the network would not be able to make predictions for consecutive pixels. This is another difference from conventional convolutional neural networks other than the 1×1 convolutional layers.

Training all the 21 possible networks and making model selection is tedious and exhausting. Instead, we have proposed the combinational fully convolutional network (CFCN) which can be viewed as a collection of many paths as shown in Fig. 6.3. By adding the skip-connections with 5×5 convoluional layers onto the main path, CFCN contains all the 21 possible paths listed in Fig. 6.2. Because only two consecutive 3×3 convolutions can end up with the same feature map size as one 5×5 convolution does, the proposed CFCN turns out to be the unique combination of all the 21 paths. In the experiment below, we will show that the CFCN has an ensemble-like behavior which guarantees robustness and high performance.

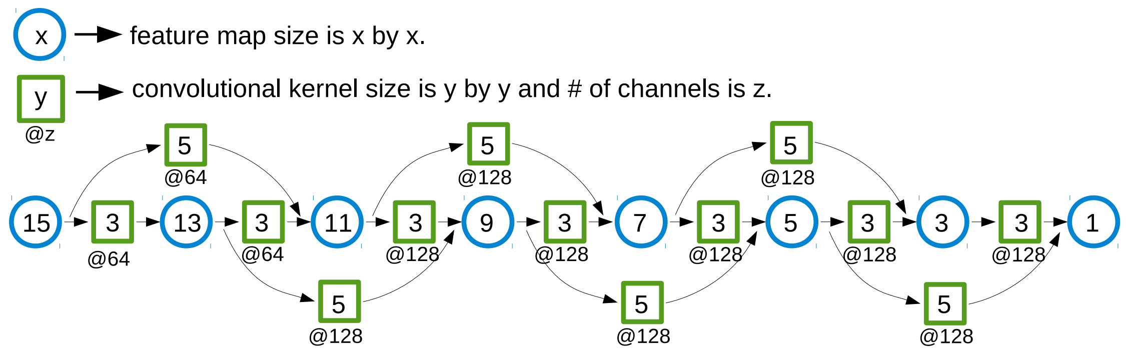

x
feature map size is x by x.
y
@z
convolutional kernel size is y by y and # of channels is z.
15
3
@64
13
3
@64
5
@64
11
3
@128
5
@128
9
3
@128
5
@128
7
3
@128
5
@128
5
3
@128
5
@128
3
3
@128
5
@128
1

Figure 6.3: The combinational fully convolutional network (CFCN) is the unique combination of all the 21 networks listed in Figure 6.2.

6.3 Experiments and results

6.3.1 Hyperspectral datasets

We carry out experiments on three hyperspectral datasets: Indian Pines, Pavia University and Salinas scene. These datasets are widely used to test the algorithms for hyperspectral image classification.

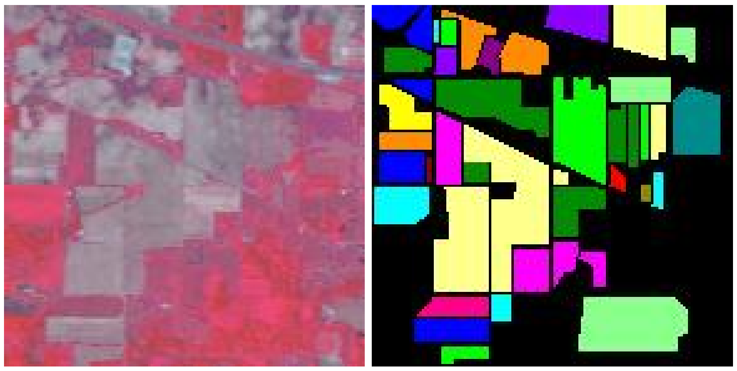
6.3.1.1 Indian pines

The hyperspectral data of Indian pines was acquired by Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) to support soils research in 1992 over Purdue University Agronomy farm northwest of West Lafayette and the surrounding area in Indiana, USA [10]. The area is a mixed agricultural/forest area, where some of the crops present, corn, soybeans, were in early stages of growth.

Figure 6.4a shows a false-color composition of the AVIRIS Indian Pines scene. This dataset consists of 145×145 pixels covering a 2 mile by 2 mile area at 20 m spatial resolution and 224 spectral bands covering the spectral range from 400 nm to 2.5 μm. Several spectral bands were removed from the dataset due to noise and water absorption phenomena, leaving a total of 200 bands to be used in the experiments. The ground-truth of the Indian Pines scene has been designated into sixteen classes (see Figure 6.4b). Table 6.1 shows the specific classes and the respective number of samples.

6.3.1.2 Pavia University

The hyperspectral data of Pavia University was acquired by Reflective Optics System Imaging Spectrometer (ROSIS) during a flight campaign over Pavia, northern Italy. Figure 6.5 illustrates the true-color image of Pavia University. It contains 610×340 pixels and the geometric resolution is 1.3 m. There are 103 spectral bands covering the spectral range from 430 to 860



(a) (b)

Figure 6.4: Indian Pines dataset. (a) False-color composition of bands 50, 27 and 17. (b) Ground-truth map. The specific classes denoted by different colors refer to Table 6.1.

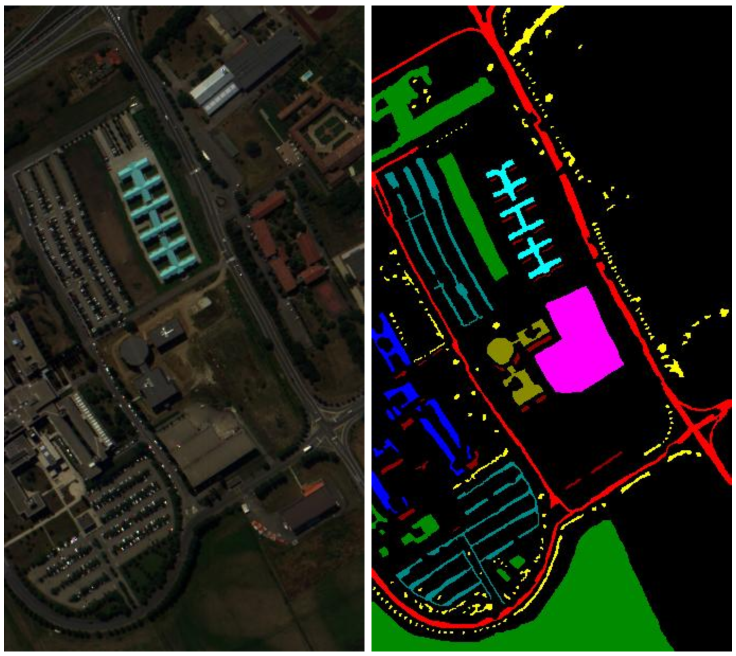
Table 6.1: Classes for the Indian Pines scene and their respective number of samples.

#	**Class**	**Num. of samples**
1	Alfalfa	46
2	Corn-notill	1428
3	Corn-mintill	830
4	Corn	237
5	Grass-pasture	483
6	Grass-trees	730
7	Grass-pasture-mowed	28
8	Hay-windrowed	478
9	Oats	20
10	Soybean-notill	972
11	Soybean-mintill	2455
12	Soybean-clean	593
13	Wheat	205
14	Woods	1265
15	Buildings-Grass-Trees-Drives	386
16	Stone-Steel-Towers	93
	Total	10249

nm. The ground-truth map differentiates 9 classes as shown in Table 6.2.

6.3.1.3 Salinas scene

Salinas scene was also collected by the 224-band AVIRIS sensor. It is over Salinas Valley, California, and the spatial resolution is relatively high (3.7-meter per pixel). The image contains



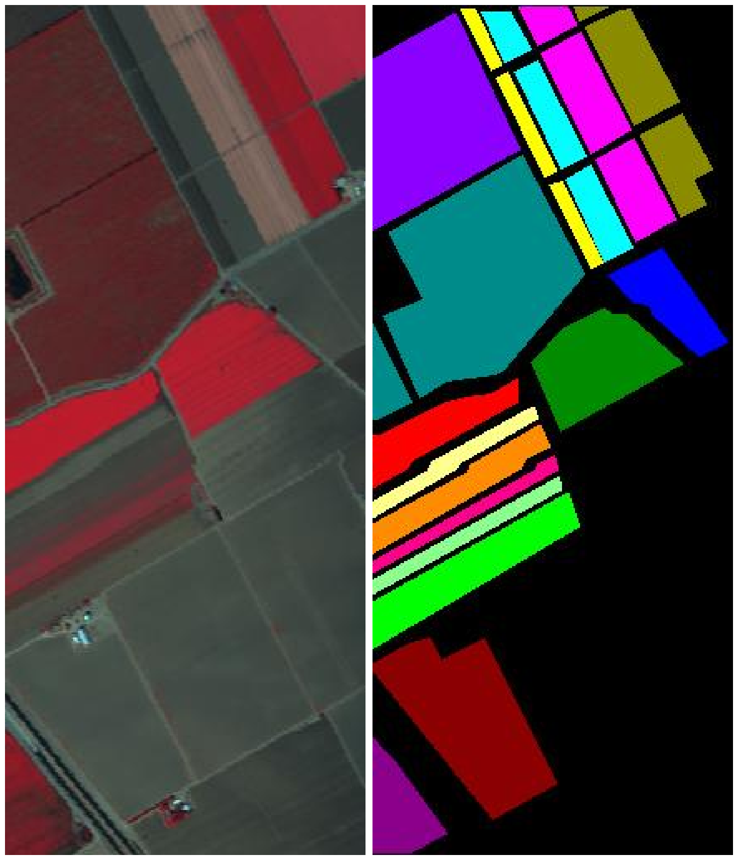
(a) (b)

Figure 6.5: Pavia University scene. (a) True-color composition of bands 53, 31 and 8. (b) Ground-truth map. The specific classes denoted by different colors refer to Table 6.2.

Table 6.2: Classes for the Pavia University scene and their respective number of samples.

#	**Class**	**Num. of samples**
1	Asphalt	6631
2	Meadows	18649
3	Gravel	2099
4	Trees	3064
5	Painted metal sheets	1345
6	Bare Soil	5029
7	Bitumen	1330
8	Self-Blocking Bricks	3682
9	Shadows	947
	Total	42776

512×217 pixels. As with the Indian Pines scene, 20 water absorption bands were discarded, leaving a total of 204 bands for experiments. The area covered includes vegetables, bare soils, and vineyard fields. The ground-truth contains 16 classes (see Table 6.3).



(a) (b)

Figure 6.6: Salinas scene dataset. (a) False-color composition of bands 50, 27 and 17. (b) Ground-truth map. The specific classes denoted by different colors refer to Table 6.3.

6.3.1.4 Data splitting

The labelled samples of Indian Pines are limited, so we randomly select 10% of the labelled samples per class for training (minimum 10 samples for the classes with extremely limited numbers of samples) and the rest of data is used for testing. For both Pavia University and Salinas scene, we randomly select 200 samples per class for training and the rest of data is used for testing. Table 6.4 indicates the amount of samples split for training and testing.

From Table 6.1, Table 6.2 and Table 6.3 we see that three datasets have unbalanced numbers of labelled samples per class. Therefore, the classification is challenging. The 3-fold cross-validation is conducted with training data to tune the hyperparameters of the network, such as learning rate, number of epochs etc. The classification performance is measured by overall ac-

Table 6.3: Classes for Salinas scene and their respective number of samples.

#	**Class**	**Num. of samples**
1	Brocoli_green_weeds_1	2009
2	Brocoli_green_weeds_2	3726
3	Fallow	1976
4	Fallow_rough_plow	1394
5	Fallow_smooth	2678
6	Stubble	3959
7	Celery	3579
8	Grapes_untrained	11271
9	Soil_vinyard_develop	6203
10	Corn_senesced_green_weeds	3278
11	Lettuce_romaine_4wk	1068
12	Lettuce_romaine_5wk	1927
13	Lettuce_romaine_6wk	916
14	Lettuce_romaine_7wk	1070
15	Vinyard_untrained	7268
16	Vinyard_vertical_trellis	1807
	Total	54129

Table 6.4: Data splitting for three datasets.

Class	Indian Pines		Salinas		Pavia Uni.	
	train	test	train	test	train	test
1	10	36	200	1809	200	6431
2	143	1285	200	3526	200	18449
3	83	747	200	1776	200	1899
4	24	213	200	1194	200	2864
5	49	434	200	2478	200	1145
6	73	657	200	3759	200	4829
7	10	18	200	3379	200	1130
8	48	430	200	11071	200	3482
9	10	10	200	6003	200	747
10	98	874	200	3078	-	-
11	246	2209	200	868	-	-
12	60	533	200	1727	-	-
13	21	184	200	716	-	-
14	127	1138	200	870	-	-
15	39	347	200	7068	-	-
16	10	83	200	1607	-	-

curacy (OA), average accuracy (AA) and Kappa coefficient (κ). Cohen's Kappa [11] is a metric that measures how much better the classifier is performing over the performance of a classifier that simply guesses at random according to the frequency of each class. It is considered a good metric for imbalanced and multi-class classification problems. A higher Kappa score indicates better performance.

The spectral bands are highly correlated in hyperspectral images. Thus, we applied PCA-whitening [12] to preprocess the spectral data. It can decorrelate the spectral bands and assure that each band has the same variance. This preprocessing step turns out to be effective. Neural networks are stochastic models. They make use of randomness by design, such as weights initialization and stochastic optimization. Thus, in order to properly evaluate the networks, we train the networks up to 10 times and use statistic mean and standard deviation to evaluate the performance.

6.3.2 Analysis of PCA-whitening

The spectral bands are highly correlated in hyperspectral images. We use the Pearson correlation coefficient [13] to evaluate the correlation between the spectral bands. Fig. 6.7 depicts the normalized distribution of pairwise Pearson correlation coefficient between each pair of spectral bands as well as the corresponding Gaussian kernel density estimate. We see that while the spectral bands of all three datasets are highly correlated, Pavia University is less correlated than the other two since it has a larger amount of correlation coefficients close to zero.

We compared the performance of the proposed network on three datasets with/without PCA-whitening. The results, shown in Table 6.5 and Fig. 6.8, demonstrate that PCA-whitening can enhance the performance, especially for Indian Pines and Salinas dataset. The enhancement of performance for Pavia University is relatively small because the spectral bands of Pavia University is less correlated than the other two.

6.3.3 The first 1×1 convolutional layer

The first 1×1 convolutional layer results in a feature map cube which has the same spatial size but different channel size with the input spectral cube. The number of kernels of the

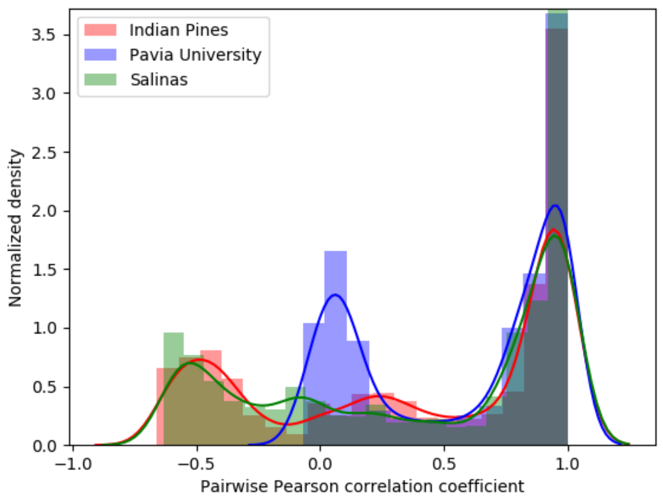
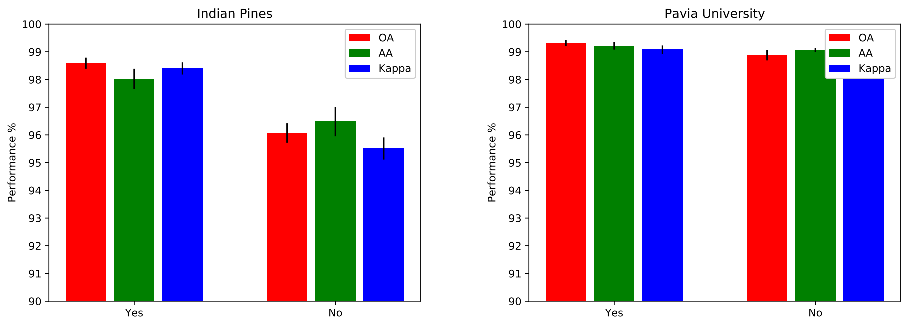


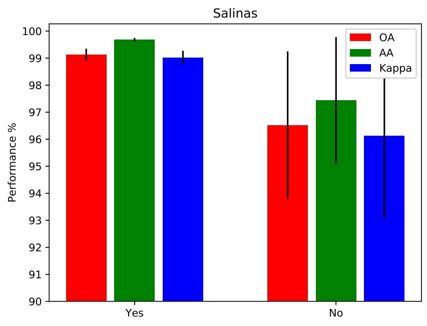
Figure 6.7: Normalized pairwise correlation distribution of three datasets.

Table 6.5: Performance with/without PCA-whitening. The patch size of spectral cubes is set to 15. *Yes* stands for with PCA-whitening and *No* stands for without PCA-whitening.

	Indian Pines		Pavia University		Salinas	
	Yes	No	Yes	No	Yes	No
OA	98.59 ± 0.20	96.07 ± 0.35	99.31 ± 0.11	98.88 ± 0.19	99.13 ± 0.22	96.52 ± 2.73
AA	98.02 ± 0.37	96.48 ± 0.53	99.22 ± 0.14	99.06 ± 0.07	99.68 ± 0.07	97.44 ± 2.34
$\kappa \times 100$	98.40 ± 0.22	95.51 ± 0.40	99.08 ± 0.15	98.50 ± 0.25	99.02 ± 0.25	96.12 ± 3.05



(a) (b)



(c)

Figure 6.8: Performance with/without PCA-whitening for (a) Indian Pines, (b) Pavia University and (c) Salinas. The bars and the vertical lines on the top stand for the average and the standard deviation of performance metrics over 10 runs, respectively. The detailed performance refers to Table 6.5.

convolutional layer equals the number of channels of the feature map cube. Each feature map in the cube is simply a linear combination of the original spectral bands together with non-linear activations. Thus, from the aspect of channel size of feature cube, we can say that the first 1×1 convolutional layer will either decrease or increase the spectral dimensionality. We have discovered that the first 1×1 convolutional layer with a large number of kernels (resulting in a large number of channels for the feature map cube) can enhance the performance of the network.

Table 6.6 shows the performance in Indian Pines dataset with different number of kernels in the first 1×1 convolutional layer, as well as the performance without the 1×1 convolutional layer. We see that the performance increases as the number of kernels increases. However, only when the number is big enough (e.g., more than 200) can the network outperform the one without the 1×1 convolutional layer. The hyperspectral image of Indian pines has 200 spectral bands. Thus, the kernel number larger than 200 leads to an overcomplete set of spectral feature maps. We believe these overcomplete feature maps can enhance the performance of the overall network.

Table 6.6: Performance in Indian pines over different number of kernels in the first 1×1 convolutional layer. *None* is the network without the 1×1 convolutional layer. *None+128* is the network without 1×1 convolutional layer but increasing the kernel sizes of the first three convolutional layers from 64 to 128. The input patch size is set to 17 for all the networks.

# of kernels	None	None+128	64	128	200	300	512	1024
OA	98.27 ± 0.25	98.28 ± 0.17	95.94 ± 0.42	97.51 ± 0.33	97.85 ± 0.30	98.53 ± 0.25	98.76 ± 0.12	98.87 ± 0.21
AA	97.62 ± 0.59	97.79 ± 0.41	94.57 ± 0.67	96.71 ± 0.68	96.95 ± 0.40	98.08 ± 0.57	98.29 ± 0.31	98.52 ± 0.40
$\kappa \times 100$	98.03 ± 0.29	98.04 ± 0.19	95.36 ± 0.48	97.16 ± 0.38	97.55 ± 0.35	98.33 ± 0.29	98.59 ± 0.14	98.71 ± 0.24

In order to prove that the enhanced performance with a larger kernel number is not simply due to the increased trainable parameters in the network, we also trained another network (*None+128* in Table 6.6) which has no such a 1×1 convolutional layer but enlarges the kernel number of the first three convolutional layers from 64 to 128. This network has $\sim$ 5.6 million trainable parameters in total which is more than the networks with 300 and 512 kernels in the 1×1 convolutional layer ($\sim$ 4.8 and $\sim$ 5.3 million respectively). However, the latter two networks can still outperform the former one. Moreover, *None+128* performs almost the same as *None* even with increased parameters. That is to say, the enhanced performance with a kernel number larger than the number of spectral bands is due to the overcompleteness of feature maps.

Adding such a 1×1 convolutional layer in front can not only enhance the performance of the proposed network, but also enhance the performance at least for Lee2017 [9] and Cao2017 [14], as we see in Table 6.7. Lee2017 is a fully convolutional network consisting mostly of 1×1 convolutional layers and the patch size 5 is the best for its architecture as stated in [9]. Cao2017 is a convolutional network which takes input as 17×17 patches. While most of previous hyperspectral image classification methods, both neural networks and non-neural-network methods, seek to compress the spectral cube with either band selection [15, 16] or dimensionality reduction [17, 18], our experiments provide a novel insight, i.e., enlarging the spectral dimensionality by learning overcomplete spectral feature maps can enhance the performance.

Table 6.7: Performance enhancement in Indian Pines by adding 1×1 convolutional layer in front for Lee2017 [9] and Cao2017 [14]. **-Plus* stands for the enhanced network. Patch size is 5 for Lee2017 and 17 for Cao2017.

		Lee2017	Lee2017-Plus	Cao2017	Cao2017-Plus
Indian Pines	OA	90.72 ± 1.37	91.26 ± 0.33	97.83 ± 0.18	98.17 ± 0.22
	AA	90.35 ± 1.89	91.16 ± 0.62	95.76 ± 0.30	97.44 ± 0.30
	$\kappa \times 100$	89.36 ± 1.58	89.97 ± 0.39	97.52 ± 0.19	97.91 ± 0.25
Pavia University	OA	91.73 ± 0.98	93.90 ± 0.40	98.81 ± 0.05	99.10 ± 0.07
	AA	93.52 ± 0.55	95.16 ± 0.25	98.50 ± 0.09	98.92 ± 0.10
	$\kappa \times 100$	89.05 ± 1.24	91.91 ± 0.51	98.38 ± 0.07	98.79 ± 0.09
Salinas	OA	93.00 ± 1.01	94.88 ± 0.39	98.09 ± 0.13	98.39 ± 0.22
	AA	96.96 ± 1.37	97.90 ± 0.12	99.33 ± 0.05	99.45 ± 0.07
	$\kappa \times 100$	92.17 ± 1.12	94.26 ± 0.43	97.86 ± 0.15	98.20 ± 0.24

6.3.4 Characteristics of CFCN

In order to analyze the characteristic of CFCN, we delete different skip-connections to test the prediction performance of a well-trained model. Deleting the skip-connections will delete some of the paths in the network. The indices of all 6 skip-connections are denoted in Fig. 6.9(a). Table 6.8 shows the number of paths deleted by the removal of each individual skip-connection. We first trained a CFCN model on the Indian Pines dataset and then deleted each individual skip-connection to predict the testing data. The corresponding results are shown in Table 6.8 and Fig. 6.9(b). As shown, deleting the first skip-connection will dramatically downgrade the prediction performance, while deleting the other skip-connections has a small impact on performance. This indicates that the first 5×5 convolutional layer is more important to network than the others and the paths through the other skip-connections do not strongly depend on each other although they are trained jointly.

Another experiment we conducted is to test the prediction performance after deleting various number of skip-connections among the later 5 ones (i.e., #2, #3, #4, #5 and #6). CFCN can be seen as a collection of many paths. This experiment is to see whether the collection of paths shows ensemble-like behavior. One critical trait of ensembles is that their performance depends smoothly on the number of members [6]. In particular, the performance improvement from additional ensemble members gets smaller with increasing ensemble size. If the collection of paths were to behave like an ensemble, we would expect the prediction performance of CFCN to smoothly correlate with the number of valid paths. This is indeed what we observe in Fig. 6.9(c): deleting increasing numbers of skip-connections from the later 5 ones downgrades the performance smoothly. This implies CFCN behaves like an ensemble to some extent.

The records in Fig. 6.9 and Table 6.8 are from experiments on Indian Pines dataset. Similar phenomena is also observed in both Pavia University and Salinas scene. To summarize, CFCN shows ensemble-like behavior to some extent in the sense that removing paths by deleting some skip-connections only has a modest and smooth impact on performance.

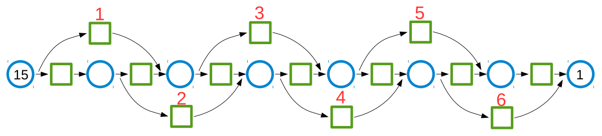
Table 6.8: Predicting performance on Indian Pines after deleting each individual skip-connection. *intact* stands for the full network.

index	1	2	3	4	5	6	intact
# of paths removed	8	5	6	6	5	8	0
OA	68.50	97.46	98.44	98.19	98.66	97.47	98.90
AA	45.61	95.16	97.88	96.94	97.83	93.18	98.33
$\kappa \times 100$	63.88	97.10	98.22	97.93	98.48	97.11	98.75

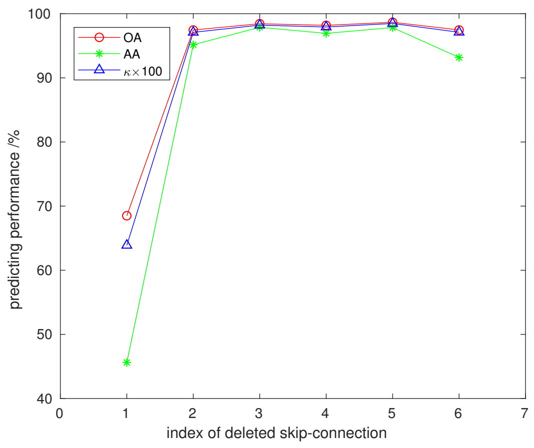
6.3.5 Patch size

In order to test out how the patch size affects the performance, we conduct experiments on various patch sizes (7, 9, 11, 13, 15, 17). Each different patch size requires a different architecture. Other than the architecture for patch size 15 (as illustrated in Fig. 6.3, we show the similar architectures for the other patch sizes in Fig. 6.10. For any architecture, the first three convolutional layers have 64 kernels and the remaining convolutional layers all have 128 kernels. Such architectures can flexibly adjust to different patch sizes.

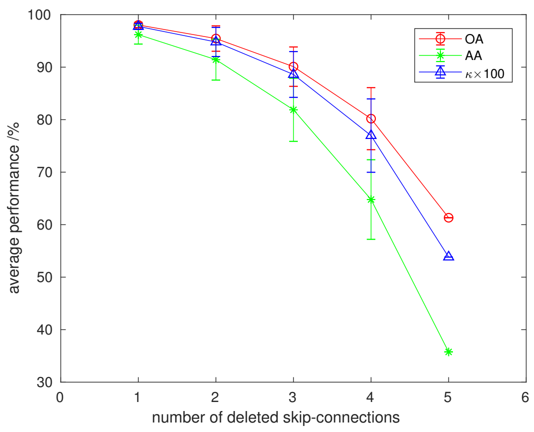
The performance of different patch size is shown in Table 6.9. As expected, there is a upward trend of performance as the patch size increases, since larger patches are better able to capture more complex features that cover a larger region of the images.



(a) index of skip-connections.



(b) performance after deleting each of skip-connections.



(c) performance after deleting various number of skip-connections among #2, #3, #4, #5 and #6.

Figure 6.9: Analyzing the characteristic of CFCN. (a) indicates the index of each skip-connection, (b) illustrates the predicting performance after deleting each individual skip-connection, (c) shows the performance after deleting various number of skip-connections among the later 5 ones.

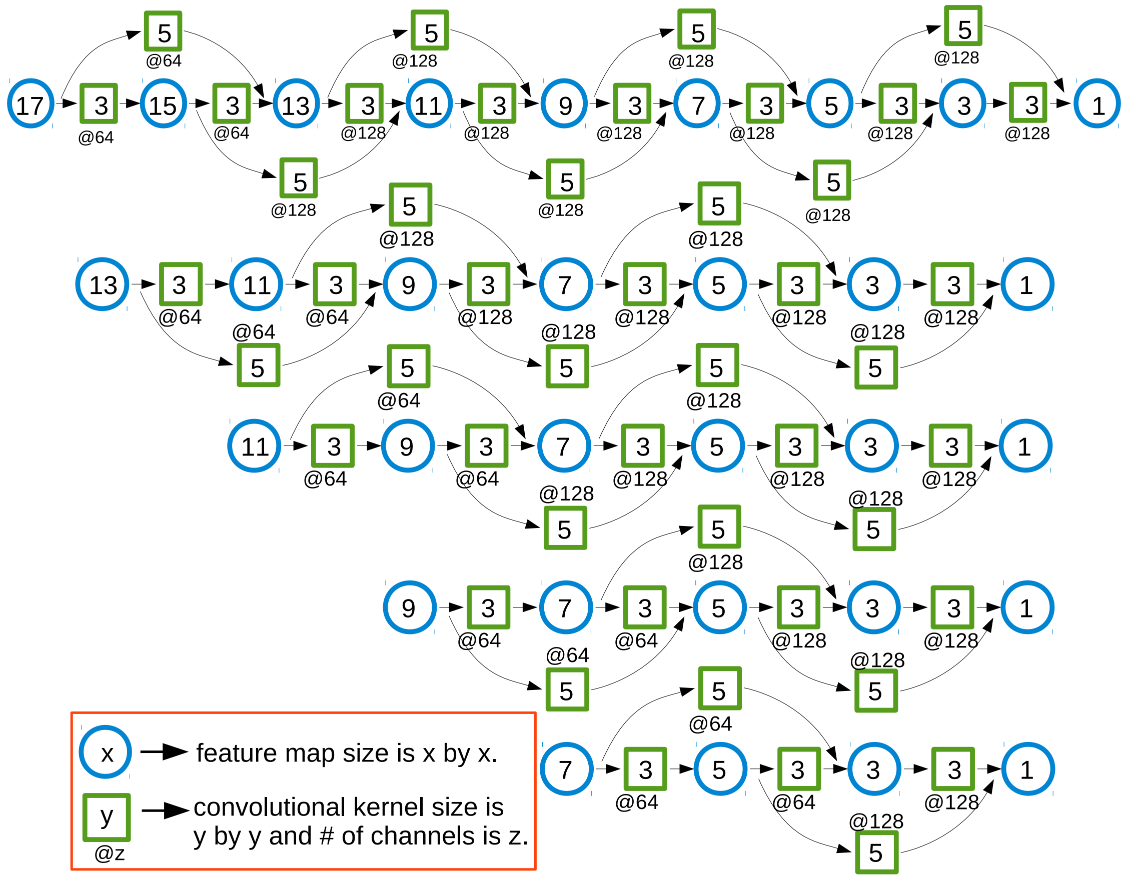

x
feature map size is x by x.
y
@z
convolutional kernel size is y by y and # of channels is z.

Figure 6.10: CFCN with different input patch sizes.

Table 6.9: Performance of CFCN over different patch sizes.

patch size		7	9	11	13	15	17
Indian Pines	OA	93.69 ± 0.50	96.77 ± 0.39	97.97 ± 0.21	98.51 ± 0.14	98.59 ± 0.20	98.87 ± 0.21
	AA	93.09 ± 0.66	96.36 ± 0.35	97.47 ± 0.27	98.14 ± 0.18	98.02 ± 0.37	98.52 ± 0.40
	$\kappa \times 100$	92.78 ± 0.58	96.31 ± 0.44	97.69 ± 0.24	98.31 ± 0.16	98.40 ± 0.22	98.71 ± 0.24
Pavia University	OA	97.46 ± 0.32	98.35 ± 0.20	98.80 ± 0.17	99.09 ± 0.19	99.31 ± 0.11	99.38 ± 0.08
	AA	97.24 ± 0.18	98.16 ± 0.08	98.63 ± 0.15	99.02 ± 0.13	99.22 ± 0.14	99.28 ± 0.12
	$\kappa \times 100$	96.59 ± 0.43	97.78 ± 0.27	98.39 ± 0.23	98.77 ± 0.26	99.08 ± 0.15	99.16 ± 0.11
Salinas	OA	94.77 ± 0.22	96.51 ± 0.49	97.92 ± 0.43	98.86 ± 0.24	99.13 ± 0.22	99.33 ± 0.17
	AA	98.01 ± 0.09	98.73 ± 0.16	99.23 ± 0.15	99.57 ± 0.10	99.68 ± 0.07	99.73 ± 0.06
	$\kappa \times 100$	94.15 ± 0.15	96.10 ± 0.55	97.67 ± 0.49	98.72 ± 0.27	99.02 ± 0.25	99.25 ± 0.20

6.3.6 Process at prediction time

With the well-trained CFCN, the whole process of predicting a hyperspectral map is straightforward. We can simply feed the whole original image into the network. With convolutional implementation of sliding window, the network can directly output a whole predicted map. Figure 6.11 illustrates the whole process of predicting.

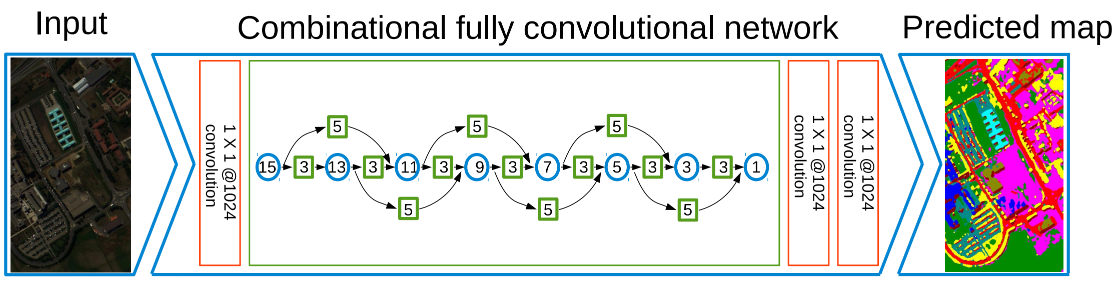


Figure 6.11: The whole process of predicting a map with combinational fully convolutional network.

6.3.7 Computational cost and performance comparison

We compared the proposed network with five other state-of-the-art networks. They are Chen2016 [19], Cao2017 [14], Cao2017-Plus, Paoletti2017 [20] and SSRN [21]. Chen2016 is a shallow 2-D CNN with input patches cropped from the first principal spectral component and its architecture is designed only for patch size 27. Cao2017-Plus is the enhanced version of Cao2017 with 1×1 convolutional layer added in front. Both Paoletti2017 and SSRN are 3-D convolutional networks. We did not add the 1×1 convolutional layer for these two networks because it would dramatically increase the computation and risk of overfitting. Except for Chen2016, all the other methods take input of 17×17 patches. For CFCN, we use the Adam optimizer and the corresponding hyperparameters after grid-search with 3-fold cross-validation are {learning_rate = 10^{-4}, batch_size = 16, epochs = 30 (Indian Pines & Salinas) or 20 (Pavia University), $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$}. The experimental settings for the other networks are the same as stated in the references. As before, we train every network 10 times and use statistic mean and standard deviation to evaluate the performance. For fair comparison, PCA-whitening is conducted for all the baseline methods and the data splitting is all the same as well.

6.3.7.1 Computational cost

One of the motivations of applying fully convolutional network for hyperspectral image classification is to take advantage of its computational efficiency. Our computing environment is Intel Core i7-3930K CPU @ 3.20GHz×12 and one NVIDIA GeForce GTX 960 with Ubuntu

16.04. The IDE is PyCharm-professional-2018.1 with Keras and Tensorflow backend. We have seen in Fig. 6.1 that, at prediction time the convolutional implementation of the sliding window can dramatically reduce duplicated computation. This has been proved in experiments. Table 6.10 indicates the consumption of computing resources (time and memory) for different methods. Clearly, with convolutional implementation of sliding window, the proposed network can remarkably reduce the memory need and predicting time. Among the comparable methods, CFCN can save roughly 10 $\sim$ 20 times prediction time and 50 $\sim$ 100 times data storage. This strongly supports the computational efficiency of the proposed fully convolutional network. The computational efficiency will have significant impact for a large hyperspectral image.

Table 6.10: The computing resource consumption (time and memory) for different methods. Memory cost is simply the size of data ready to input to network. The data type is npy_float16. *CFCN-Yes* stands for the proposed network with convolutional implementation of sliding window at predicting time and *CFCN-No* stands for the proposed network without convolutional implementation of sliding window.

		Chen2016	Cao2017	Cao2017-Plus	Paoletti2017	SSRN	CFCN-No	CFCN-Yes
Indian Pines	Training time (s)	19.87	20.1	61.1	71.2	522.4	93.9	93.9
	Predicting time (s)	0.94	7.7	15.5	15.1	63.9	12.7	1.2
	Memory (megabytes)	14.9	1184.8	1184.8	1184.8	1184.8	1184.8	10.1
Pavia University	Training time (s)	28.7	28.12	99.9	224.8	556.6	154.3	154.3
	Predicting time (s)	3.8	21.13	50.8	71.9	143.7	48.6	5.91
	Memory (megabytes)	62.4	2546.6	2546.6	2546.6	2546.6	2546.6	45.1
Salinas	Training time (s)	42.1	66.4	192.6	221.8	1569.4	285.1	285.1
	Predicting time (s)	4.8	49.5	82.7	80.7	341.9	68.4	4.2
	Memory (megabytes)	78.9	6382.3	6382.3	6382.3	6382.3	6382.3	49.0

Fig. 6.12 illustrates the process of loss convergence during training in Indian Pines for all the methods. All the methods converge within 30 epochs. Though CFCN is not the fastest to converge over the epochs, its time cost for each epoch is relatively small. Thus the entire training time is not much compared to the other methods.

6.3.7.2 Performance comparison

Table 6.11 presents the performance for all the methods and Fig. 6.13 illustrates the classified maps. Paoletti2017 and SSRN are the most comparable baselines. Although both Paoletti2017 and SSRN slightly outperform CFCN in Salinas scene, CFCN performs better than all the baselines in both Indian Pines and Pavia University. Even if Chen2016 cropped 27×27 patches as training samples, which intuitively incorporates more complicated spatial structures,

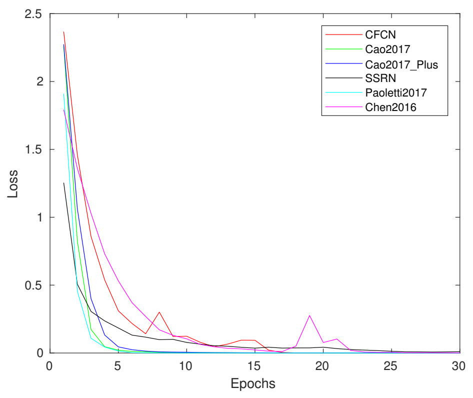


Figure 6.12: Loss convergence over epochs during training in Indian Pines.

it performs badly since it compresses the spectral dimension with principal component analysis which must have cast away some useful spectral information. Cao2017 is a light network which contains only two convolutional layers. It performs much better than Chen2016. However, two convolutional layers are too shallow to extract deep features as the proposed network does. Therefore the proposed network can outperform it.

Table 6.11: Performance comparison with the other state-of-the-art networks.

		Chen2016 (27×27)	Cao2017	Cao2017-Plus	Paoletti2017	SSRN	CFCN
Indian Pines	OA	95.38 ± 0.29	97.83 ± 0.18	98.17 ± 0.22	98.56 ± 0.26	98.68 ± 0.12	98.87 ± 0.21
	AA	94.28 ± 0.47	95.76 ± 0.30	97.44 ± 0.30	98.16 ± 0.37	97.97 ± 0.43	98.52 ± 0.40
	$\kappa \times 100$	94.73 ± 0.33	97.52 ± 0.19	97.91 ± 0.25	98.36 ± 0.29	98.61 ± 0.14	98.71 ± 0.24
Pavia University	OA	94.48 ± 2.86	98.81 ± 0.05	99.10 ± 0.07	99.33 ± 0.38	99.20 ± 0.96	99.38 ± 0.08
	AA	96.55 ± 1.92	98.50 ± 0.09	98.92 ± 0.10	99.21 ± 0.29	98.48 ± 1.04	99.28 ± 0.12
	$\kappa \times 100$	92.71 ± 3.61	98.38 ± 0.07	98.79 ± 0.09	99.10 ± 0.51	98.92 ± 0.89	99.16 ± 0.11
Salinas	OA	93.19 ± 1.48	98.09 ± 0.13	98.39 ± 0.22	99.53 ± 0.09	99.42 ± 0.06	99.33 ± 0.17
	AA	96.53 ± 1.02	99.33 ± 0.05	99.45 ± 0.07	99.84 ± 0.03	99.80 ± 0.04	99.73 ± 0.06
	$\kappa \times 100$	92.38 ± 1.66	97.86 ± 0.15	98.20 ± 0.24	99.37 ± 0.10	99.27 ± 0.07	99.25 ± 0.20

6.4 Conclusion

In this chapter, we propose a combinational fully convolutional network (CFCN) for hyperspectral image classification. The network can take advantage of the inherent computational efficiency of convolution at prediction time, i.e., the prediction is performed on the whole image at a time and the computation is highly amortized over the overlapping regions of patches. The FCN for typical semantic segmentation contains subsampling and upsampling layers, which make it difficult for the network to capture detailed structures of objects. Hyperspectral image

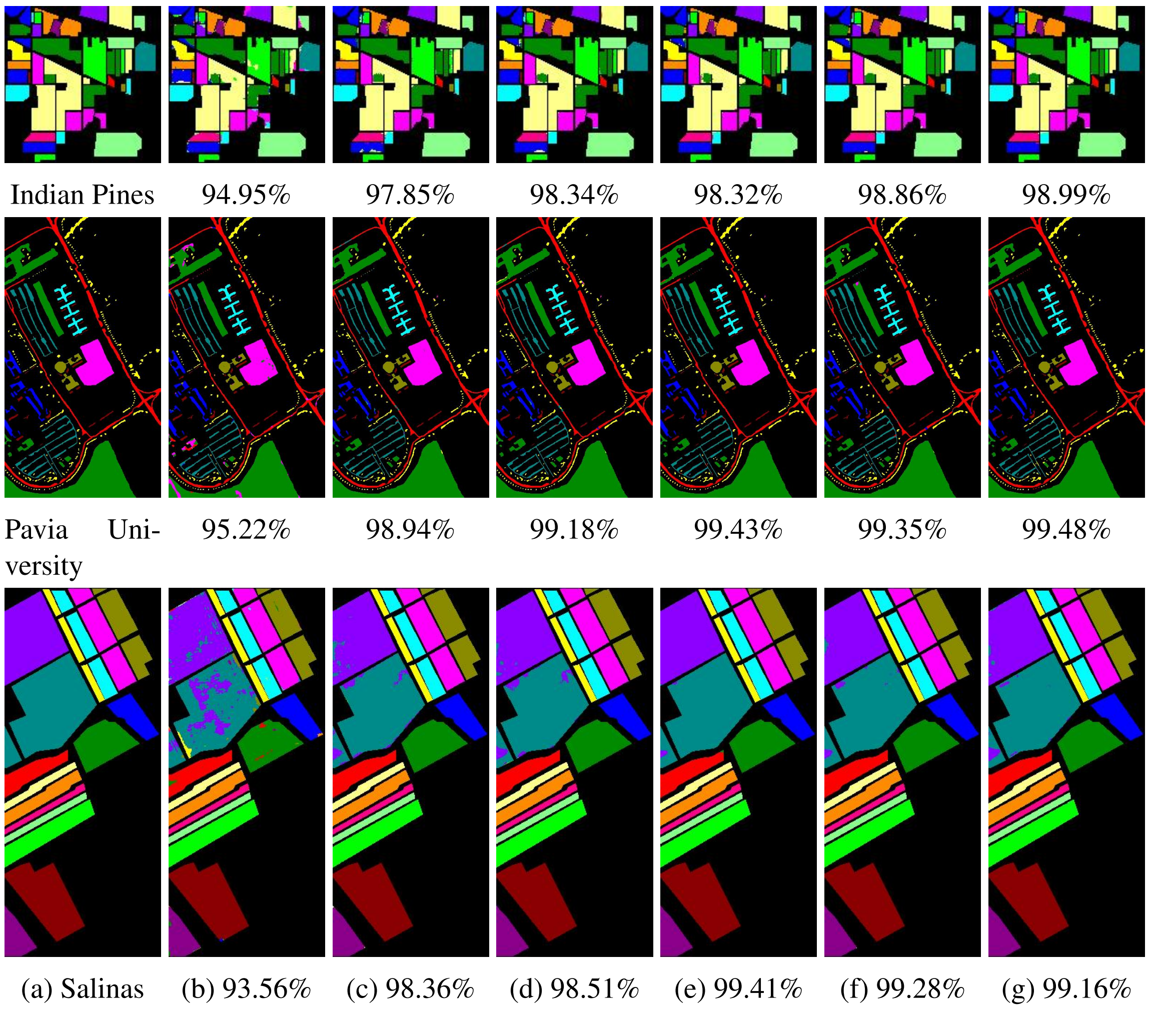


Figure 6.13: Classified maps with different methods. The first row is for Indian Pines, the second row is for Pavia University and the third row is for Salinas scene. The columns are (a) the ground-truth maps, (b) Chen2016, (c) Cao2017, (d) Cao2017-Plus, (e) Paoletti2017, (f) SSRN, and (g) CFCN.

classification places more value on pixelwise accuracy. Thus, our proposed network consists of only convolutional layers which can capture detailed spatial structures. In addition, the combinational network can be seen as a collection of many paths. Experimental results show ensemble-like behavior to some extent in the sense that removing paths from CFCN by deleting some skip-connections only has a modest and smooth impact on performance. Further experiments on three hyperspectral datasets indicate that CFCN is computationally efficient and can outperform other state-of-the-art methods.

Bibliography

[1] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[2] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[3] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651, 2017.

[4] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.

[5] Harris Drucker, Corinna Cortes, Lawrence D Jackel, Yann LeCun, and Vladimir Vapnik. Boosting and other ensemble methods. *Neural Computation*, 6(6):1289–1301, 1994.

[6] Andreas Veit, Michael J Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. In *Advances in Neural Information Processing Systems*, pages 550–558, 2016.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[8] David Eigen, Dilip Krishnan, and Rob Fergus. Restoring an image taken through a window covered with dirt or rain. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 633–640. IEEE, 2013.

[9] Hyungtae Lee and Heesung Kwon. Going deeper with contextual cnn for hyperspectral image classification. *IEEE Transactions on Image Processing*, 26(10):4843–4855, 2017.

[10] M Baumgardner, L Biehl, and D Landgrebe. 220 band aviris hyperspectral image data set: June 12, 1992 indian pine test site 3. *Purdue University Research Repository*, 2015.

[11] J Richard Landis and Gary G Koch. The measurement of observer agreement for categorical data. *Biometrics*, pages 159–174, 1977.

[12] Patrik O. Hoyer Aapo Hyvrinen, Jarmo Hurri. *Natural Image Statistics*. Springer, 2009.

[13] Joseph Lee Rodgers and W Alan Nicewander. Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66, 1988.

[14] Xiangyong Cao, Feng Zhou, Lin Xu, Deyu Meng, Zongben Xu, and John Paisley. Hyperspectral image segmentation with markov random fields and a convolutional neural network. *arXiv preprint arXiv:1705.00727*, 2017.

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

[16] Yuan Yuan, Jianzhe Lin, and Qi Wang. Hyperspectral image classification via multitask joint sparse representation and stepwise mrf optimization. *IEEE Transactions on Cybernetics*, 46(12):2966–2977, 2016.

[17] Yushi Chen, Xing Zhao, and Xiuping Jia. Spectral–spatial classification of hyperspectral data based on deep belief network. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(6):2381–2392, 2015.

[18] Wenzhi Zhao and Shihong Du. Spectral–spatial feature extraction for hyperspectral image classification: A dimension reduction and deep learning approach. *IEEE Transactions on Geoscience and Remote Sensing*, 54(8):4544–4554, 2016.

[19] Yushi Chen, Hanlu Jiang, Chunyang Li, Xiuping Jia, and Pedram Ghamisi. Deep feature extraction and classification of hyperspectral images based on convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 54(10):6232–6251, 2016.

[20] ME Paoletti, JM Haut, J Plaza, and A Plaza. A new deep convolutional neural network for fast hyperspectral image classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2017.

[21] Zilong Zhong, Jonathan Li, Zhiming Luo, and Michael Chapman. Spectral–spatial residual network for hyperspectral image classification: A 3-d deep learning framework. *IEEE Transactions on Geoscience and Remote Sensing*, 56(2):847–858, 2018.

Chapter 7

Conclusion

Planetary exploration continues to advance. By maximizing the scientific return and reducing the need for human involvement, robotic autonomy is increasingly playing an important role. In some cases, autonomous techniques will be the essential enabling tools for missions that cannot be achieved without them. This research work studies two problems in planetary exploration - rock image classification and hyperspectral image classification. Autonomous rock image classification can enhance the capability of robotic devices on surface mission for geological detection and autonomous hyerspectral image classification enables the detection on planetary surface without any physical contact and avoids labored human investigation.

For surface missions, geology will continue to be a core focus. Rocks can provide information as to the environment in which the rocks were created and the subsequent geological history, and unveil the origin and evolution of rocky planetary bodies throughout the Solar system. In the first part of this thesis work, we proposed an unsupervised feature learning method for representing the inhomogeneous rock texture and conducted rock image classification. For comparison, several sets of manual features were utilized as well. We found that different combinations of manual features affected classification substantially, whereas our proposed feature learning method performed well. While there is no guarantee that this feature learning method can absolutely outperform any manual feature configuration, it is easily implemented and more flexible than the manual features. We also explored the use of self-taught learning based on unsupervised feature learning for rock image classification. It can learn the feature representation directly from unlabelled images of mixed rock types, and then repeatedly apply the feature representation to different sub-classes of rocks. We suggest that the fundamental reason as to why this approach works is that rock images share some basic visual patterns or elements. Therefore, as long as these basic patterns can be learned from the whole mixed dataset, they can be well utilized for representing the new groups of images belonging to the sub-classes. The proposed feature learning method can also be applied to geological image archive (e.g., autonomous labelling) or image retrieval etc.

The rest of the thesis work was focused on hyperspectral image classification. We proposed a framework of modeling spectral information together with spatial contextual information to generate spatial-spectral features, two light convolutional neural networks with stacking spectral patches and a combinational fully convolutional network in Chapter 4, 5 and 6, respectively. In Chapter 4, two unsupervised learning methods - K-means and PCA - are utilized to learn the spatial feature bases in each retained spectral band after decorrelation. Then the spatial feature representations are extracted with these spatial feature bases. By concatenating the spatial feature representations in all/principal spectral bands, we get the spatial-spectral features. The spatial-spectral features are more flexible compared with the conventional feature methods and can cover more complicated spatial structures. We also found that decorrelating the spectral bands as the preprocessing step can dramatically enhance the performance. Retaining more Spectral-PCs results in better performance since even the small amount of variance in the last tens of spectral principal components can contribute to the classification and make the feature representations more robust. However, this comes with the cost of increased computation time. Thus, one can compromise between higher performance and faster computation in a specific situation.

Chapter 5 proposed an innovative framework to classify hyperspectral image with two light convolutional neural networks. First, PCA whitening is applied to decorrelate hundreds of spectral bands. Instead of selecting the principal components to reduce the spectral dimensionality, we retain all the spectral bands but compress the image cuboid into 1-channel spectral quilt by stacking spectral patches. In this way, not only is all the spectral information retained, but also the computational complexity of training a neural network is reduced compared with conventional methods. Moreover, the spectral quilts will contain some brand new textural patterns which could be useful for distinguishing classes. Two light convolutional neural networks are then applied to classify the spectral quilts. The performance of the proposed networks is compared with two conventional methods which model the HSI with hand-crafted features and the results show that the proposed methods can dramatically outperform the conventional methods.

Last but not least, Chapter 6 proposed a combinational fully convolutional network (CFCN) for hyperspectral image classification. The network can take advantage of the inherent computational efficiency of convolution at prediction time, i.e., the prediction is performed on the whole image at a time and the computation is highly amortized over the overlapping regions of patches. The fully convolutional network for typical semantic segmentation contains subsampling and upsampling layers, which make it difficult for the network to capture detailed structures of objects. Hyperspectral image classification places more value on pixelwise accuracy. Thus, our proposed network consists of only convolutional layers which can capture detailed spatial structures. In addition, the combinational network can be seen as a collection of many paths. Experimental results show ensemble-like behavior to some extent in the sense that removing paths from CFCN by deleting some skip-connections only has a modest and smooth impact on performance. Further experiments on three hyperspectral datasets indicate that CFCN is computationally efficient and can outperform the other state-of-art baselines.

Two problems in planetary exploration have been well addressed, for each of which several autonomous techniques have been proposed. Of course, there are still a lot of work to do in the future. For rock image classification, the feature learning method should be generalized and tested on a larger and more general rock image dataset. As such, techniques for hyperspectral image classification need to be improved and tested on more real spectral datasets. An extended application can be studied on multispectral images which have less spectral information but more detailed spatial information.

Curriculum Vitae

Name:	Lei Shu
Post-Secondary Education and Degrees:	Harbin Institute of Technology Harbin, China 2011 - 2013 M.Sc. Control Science and Technology
	Northwestern Polytechnical University Xi'an, China 2007 - 2011 B.Sc. Detection Guidance and Control Technology
Related Work Experience:	Research/Teaching Assistant The University of Western Ontario 2014 - 2018

Publications:

1. Shu, L., McIsaac, K., Osinski, G.R. Unsupervised feature learning for autonomous rock image classification. Computers & Geosciences, 106, pp.10-17, 2017.

2. Shu, L., McIsaac, K., and Osinski, G.R. Learning spatial-spectral features for hyperspectral image classification. IEEE Transactions on Geoscience and Remote Sensing, 2018.

3. Shu, L., McIsaac, K., and Osinski, G.R. Hyperspectral image classification with stacking spectral patches and convolutional neural networks. IEEE Transactions on Geoscience and Remote Sensing, 2018.

4. Shu, L., McIsaac, K., and Osinski, G.R. Hyperspectral image classification with combinational fully convolutional network. IEEE Transactions on Geoscience and Remote Sensing. Under review.

5. Shu, L., McIsaac, K., and Osinski, G.R. Automatic geological mapping with multi-spectral image classification. IEEE Journal of Selected Topics in Applied Earth Observation & Remote Sensing. Under review.

6. Shu, L., Osinski, G.R. and McIsaac, K. An automatic methodology for analyzing sorting level of rock particles. Computers & Geosciences. Accepted.