

# Spatial and Structural Classification Analysis for Advanced Remotely Sensed Imagery

江田, 智尊

<https://hdl.handle.net/2324/1959077>

---

出版情報 : Kyushu University, 2018, 博士 (機能数理学) , 課程博士  
バージョン :  
権利関係 :



# **Spatial and Structural Classification Analysis for Advanced Remotely Sensed Imagery**

Satoru Koda

Graduate School of Mathematics  
Kyushu University

2018



## Acknowledgments

*I extend my profound thanks particularly to my supervisor, Dr. Ryuei Nishii, for his considerable and continuous support of my Ph.D. studies. He has raised not only my mathematical and statistical abilities but also my practical skills through a wide variety of joint research that he organized with companies and other research institutes. I also thank my supervisor in Italy, Dr. Farid Melgani, for accepting me for overseas study and giving me interesting topics on remote sensing data analysis. His feedback consistently brought our research to a more advanced stage.*

*I greatly appreciate the members of my Ph.D. committee, Dr. Hirokazu Anai, Dr. Yoshihiko Maesono, and Dr. Hiroki Masuda, for their reviews and valuable comments on this dissertation.*

*I am very grateful to my research collaborators: Dr. Keiichi Mochida, Dr. Yoshihiko Onda, and Dr. Hidetoshi Matsui. They have provided me with a fascinating research subject, analytical outcomes, and comments on my works from the perspectives of genome science and statistics.*

*I must also express my gratitude to all past and present laboratory members with whom I have worked in Japan and Italy. Research discussions, seminars, and ordinary conversations with them have fostered my motivation to do research. I also thank our secretary, Ms. Kyoko Ikebe, for her great dedication to our laboratory's management.*

*Last but not least, I cannot find words to fully acknowledge my dear family, and especially to my parents for giving birth to me in the first place. I could not have completed my Ph.D. studies without their tremendous support and sacrifices for me.*

Satoru Koda

September-2018



# Contents

|          |                                                     |           |
|----------|-----------------------------------------------------|-----------|
| <b>1</b> | <b>Introduction</b>                                 | <b>4</b>  |
| 1.1      | Remote Sensing Platforms and Applications . . . . . | 4         |
| 1.1.1    | Developments in Sensing Technology . . . . .        | 4         |
| 1.1.2    | Sensing Platforms . . . . .                         | 5         |
| 1.1.3    | Unmanned Aerial Vehicles . . . . .                  | 6         |
| 1.2      | Research Motivations and Challenges . . . . .       | 7         |
| 1.2.1    | Problem Formulation . . . . .                       | 8         |
| 1.2.2    | Multilabel Classification . . . . .                 | 10        |
| 1.2.3    | Output Structure . . . . .                          | 11        |
| 1.2.4    | Spatial Contiguity . . . . .                        | 11        |
| 1.2.5    | Hyperspectral Imaging . . . . .                     | 12        |
| 1.3      | Our Strategy . . . . .                              | 13        |
| 1.4      | Overview . . . . .                                  | 14        |
| <b>2</b> | <b>Problem Formulation and Tools</b>                | <b>16</b> |
| 2.1      | Coarse Description . . . . .                        | 16        |
| 2.2      | Feature Extraction via Neural Networks . . . . .    | 17        |
| 2.3      | Classification . . . . .                            | 23        |
| 2.3.1    | Support Vector Machine . . . . .                    | 23        |
| 2.3.2    | Binary Support Vector Machine . . . . .             | 24        |
| 2.3.3    | Multiclass Support Vector Machine . . . . .         | 26        |
| <b>3</b> | <b>Structural Learning</b>                          | <b>30</b> |

|          |                                                                    |           |
|----------|--------------------------------------------------------------------|-----------|
| 3.1      | Structural Support Vector Machine . . . . .                        | 30        |
| 3.1.1    | Structured Loss . . . . .                                          | 32        |
| 3.1.2    | Examples of Structural Learning . . . . .                          | 34        |
| 3.2      | Training of Structural Support Vector Machine . . . . .            | 36        |
| 3.2.1    | One-Slack Reformulation . . . . .                                  | 36        |
| 3.2.2    | Cutting-Plane Algorithm . . . . .                                  | 38        |
| <b>4</b> | <b>Spatial Support Vector Machine</b>                              | <b>41</b> |
| 4.1      | Spatial Embedding into Structural Support Vector Machine . . . . . | 41        |
| 4.2      | Numerical Experiments . . . . .                                    | 47        |
| 4.2.1    | Datasets . . . . .                                                 | 47        |
| 4.2.2    | Experimental Setup . . . . .                                       | 48        |
| 4.2.3    | Results . . . . .                                                  | 49        |
| 4.3      | Summary . . . . .                                                  | 55        |
| <b>5</b> | <b>Numerical Experiments on UAV Imagery</b>                        | <b>56</b> |
| 5.1      | Datasets . . . . .                                                 | 56        |
| 5.2      | Experimental Setup . . . . .                                       | 58        |
| 5.3      | Results . . . . .                                                  | 60        |
| 5.4      | Summary . . . . .                                                  | 67        |
| <b>6</b> | <b>Unsupervised Feature Extraction Using Autoencoders</b>          | <b>69</b> |
| 6.1      | Related Works . . . . .                                            | 70        |
| 6.2      | Strategy . . . . .                                                 | 72        |
| 6.2.1    | Autoencoder . . . . .                                              | 72        |
| 6.2.2    | Spatial Embedding into Autoencoder . . . . .                       | 74        |
| 6.3      | Numerical Experiments . . . . .                                    | 76        |
| 6.3.1    | Datasets . . . . .                                                 | 77        |
| 6.3.2    | Experimental Setups . . . . .                                      | 77        |
| 6.3.3    | Results . . . . .                                                  | 78        |
| 6.4      | Summary . . . . .                                                  | 85        |

|                                                                                                                                    |           |
|------------------------------------------------------------------------------------------------------------------------------------|-----------|
| <b>7 Conclusion</b>                                                                                                                | <b>86</b> |
| <b>Bibliography 1</b>                                                                                                              | <b>88</b> |
| <b>8 Other Topic: Diurnal Transcriptome and Gene Network Represented Through Sparse Modeling in <i>Brachypodium distachyon</i></b> | <b>97</b> |
| 8.1 Introduction . . . . .                                                                                                         | 97        |
| 8.1.1 Research Background . . . . .                                                                                                | 97        |
| 8.1.2 Gene Regulatory Network . . . . .                                                                                            | 98        |
| 8.1.3 Algorithms for Gene Network Inference . . . . .                                                                              | 99        |
| 8.1.4 Model Plant: <i>Brachypodium distachyon</i> . . . . .                                                                        | 100       |
| 8.1.5 Contribution . . . . .                                                                                                       | 100       |
| 8.2 Materials and Methods . . . . .                                                                                                | 101       |
| 8.2.1 Plant Materials . . . . .                                                                                                    | 101       |
| 8.2.2 Transcriptome Analysis . . . . .                                                                                             | 101       |
| 8.2.3 Functional Annotation . . . . .                                                                                              | 102       |
| 8.2.4 Gene Expression Analysis . . . . .                                                                                           | 102       |
| 8.2.5 Identification of Periodically Expressed Genes . . . . .                                                                     | 103       |
| 8.2.6 Discovery of Cis-Regulatory Motifs in Promoter Regions of Periodically Expressed Genes . . . . .                             | 103       |
| 8.2.7 GRN Estimation Based on ARX Models with Group SCAD . . . . .                                                                 | 104       |
| 8.3 Results and Discussion . . . . .                                                                                               | 104       |
| 8.3.1 Diurnal Transcriptome of <i>B. distachyon</i> Leaves . . . . .                                                               | 104       |
| 8.3.2 Identification of Periodic Genes in <i>B. distachyon</i> . . . . .                                                           | 106       |
| 8.3.3 GCN for Periodic Genes in <i>B. distachyon</i> . . . . .                                                                     | 108       |
| 8.3.4 GRN Inference Based on Group SCAD Analysis . . . . .                                                                         | 110       |
| 8.4 Summary . . . . .                                                                                                              | 113       |
| Appendix 1 . . . . .                                                                                                               | 114       |
| Appendix 2 . . . . .                                                                                                               | 115       |
| Bibliography 2 . . . . .                                                                                                           | 115       |

# Chapter 1

## Introduction

The introductory chapter of this dissertation begins by describing general information about remote sensing, such as image acquisition platforms, sensor technologies, and developed applications of remotely sensed data analysis. We then review emerging challenges raised by recent technological advancements in remote sensing, especially by unmanned aerial vehicles (UAVs). We examine these challenging problems in order and describe how to cope with them through the rest of the dissertation.

### 1.1 Remote Sensing Platforms and Applications

#### 1.1.1 Developments in Sensing Technology

*Remote sensing* means measuring information through the reflected energy from target objects under monitoring, without direct contact. In the context of geoscience, the target is typically the Earth's surface. There are two types of sensors for collecting surface information: active and passive sensors. Their difference depends on whether a sensor has its own energy source for observation. For instance, active sensors emit signals using their own energy, which travel through the atmosphere, and then measure the radiation reflected from the surface. Common active sensor technologies used in remote sensing include Radars and light detection and ranging (LiDAR), which are based on radio waves and lasers, respectively. These sensors are mounted on satellites or aircraft to collect information such as the distance, range, velocity, angle, and elevation of target areas under surveillance. In contrast, passive sensors do not have their own energy source but sim-

ply observe the latent energy that target objects reflect (e.g., the sun's radiation or electromagnetic waves). Photographic cameras and multispectral scanners are examples of passive sensors (see, e.g., Patino and Duque (2013)).

Remote sensing technology has been highly developed in recent decades according to technological progress in sensors. Satellite-based platforms with so-called first-generation sensors such as Landsat's multispectral scanner (MSS) were launched beginning in 1972, with a spatial resolution of 80 m per pixel. Data acquired by such sensors were practically used to create forest fire hazard maps (Chuvieco and Congalton, 1989) and land use maps (Wang, 1990; Benediktsson et al., 1990), for instance. The applications were rather limited, however, due to the sparse resolution. After second-generation sensors with a data resolution of 10-30 m, third-generation sensors on platforms such as IKONOS were developed beginning in 1999. These sensors first supplied us with very-high resolution (VHR) imagery of land surfaces, providing multispectral imagery with 4-m resolution. Then, in the early 21st century, last-generation sensors on platforms such as GeoEye, QuickBird, and WorldView were developed, with spatial resolution of 1-4 m in multispectral imagery. Such advancements in remote sensing image acquisition technologies and the resulting availability of VHR imagery have stimulated development of a wide variety of applications in various fields. For instance, Ke et al. (2010) used QuickBird multispectral imagery and LiDAR data for forest species classification, with the results showing that VHR imagery can be used to identify not only forest areas but also individual forest species. Salehi et al. (2012) used optical satellite imagery acquired by the WorldView-2 satellite to detect moving vehicles, which showed the possibility of detecting such small objects through VHR imagery. Coltri et al. (2013) aimed to evaluate coffee biomass and carbon with GeoEye-1 satellite imagery. They contributed to developing a platform that enables measurement of biophysical properties through VHR imagery without resorting to destructive approaches, i.e., direct contact with objects.

### **1.1.2 Sensing Platforms**

Besides the development of image acquisition sensors, various sensing platforms have also contributed to the progress of remote sensing technology and provided imagery with different properties. Compared with satellite-based platforms, airborne platforms equipped with image acquisition sensors, such as helicopters and aircraft, can provide much more accurate, high-quality geometri-

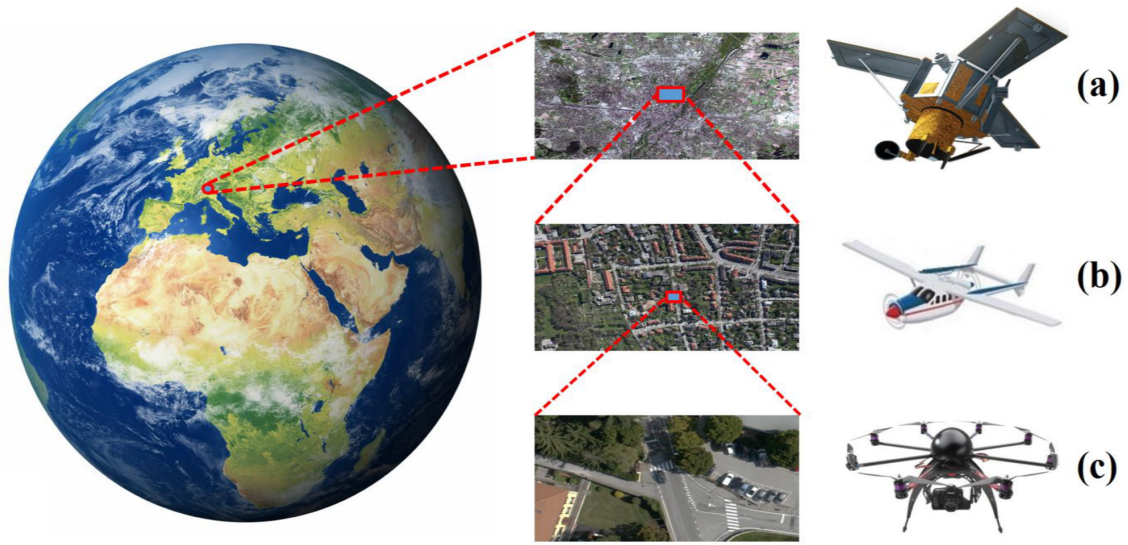


Figure 1.1: Examples of imagery covering the same area, with different spatial resolutions and acquired by different platforms: satellite (a), aircraft (b), and unmanned aerial vehicle (c).

cal observations. Figure 1.1 shows examples of images acquired by different platforms (Zeggada, 2018). As shown in Fig.1.1(a), satellite-based platforms can capture objects across a very wide area. Although the coverage of airborne platforms, as shown in Fig.1.1(b), is smaller than that of satellite platforms, their images are more suitable for specific applications requiring precise observation. For instance, Paris and Bruzzone (2015) presented a tree height estimation system based on combining low-density LiDAR data and high-resolution optical imagery. Another application was put forth by Wang et al. (2017), who proposed automatic extraction systems for power lines with airborne LiDAR data.

Despite these benefits, however, airborne platforms are not cost-efficient and require pilots. These shortcomings make them unsuitable to applications involving routine maintenance and emergency response. Therefore, given the demand for platforms more mobile than aircraft, UAVs, commonly known as drones, have strongly attracted attention in recent years.

### 1.1.3 Unmanned Aerial Vehicles

UAVs were originally developed in the early 1900s for military purposes, such as spying on dangerous battlefields and surveying areas inaccessible to humans. Since 2012, they have been used for commercial purposes. Their advantages can be characterized as enhancing data collectability,

customizability, portability, and cost efficiency. They can deploy appropriate sensors depending on mission requirements, fly above regions that satellites or aircraft cannot cover, and cope with their limited coverage by flying at any time, as needed. The imagery provided by UAVs is much more accurate than VHR imagery, i.e., it provides extremely-high resolution (EHR), with a spatial resolution of a few centimeters. In Fig.1.1(c), for example, the spatial resolution is 2 cm per pixel. Because of their excellent characteristics, UAVs have led to the development of highly practical applications such as high-accuracy land use mapping, traffic monitoring, disaster victim detection, and precision farming.

For instance, Ke et al. (2017) addressed the problem of processing real-time traffic data extracted from video captured onboard UAVs for traffic monitoring applications, where the camera platforms move with the UAVs. The proposed system quickly estimates traffic parameters such as the traffic direction, number of vehicles, and average stream speed. Aguilar et al. (2017) presented another work related to traffic systems and UAV-based camera applications, namely pedestrian detection. Li et al. (2016) developed a system of monitoring “UAV traffic,” with which they proposed an effective strategy to detect and track multiple UAVs with a single moving camera mounted on one UAV to systemize collision avoidance systems. In the field of disaster and victim detection applications, Bejica et al. (2017) interestingly suggested a method of assisting avalanche search and victim detection with UAVs fitted with vision cameras. Sumalan et al. (2016) developed a method for detecting and evaluating flooded areas in natural disaster scenarios with color information from UAV imagery. As for precision agriculture and forest monitoring applications, a relatively large number of contributions have been reported in the literature. David and Ballado (2016), for instance, introduced an automatic estimation system using UAV imagery to map land use in a vegetable farm. Sarkar et al. (2016) successfully developed a UAV-based sensing methodology to detect citrus greening disease, which is a major threat to Florida citrus production.

## **1.2 Research Motivations and Challenges**

Along with the emerging developments in sensors and sensing platforms reviewed above, their applications have enabled remarkable progress in a wide variety of research fields. Despite their effectiveness, these new technologies have raised several challenges. Because of these challenges, traditional analytical tools developed on VHR imagery cannot provide reasonable outcomes for

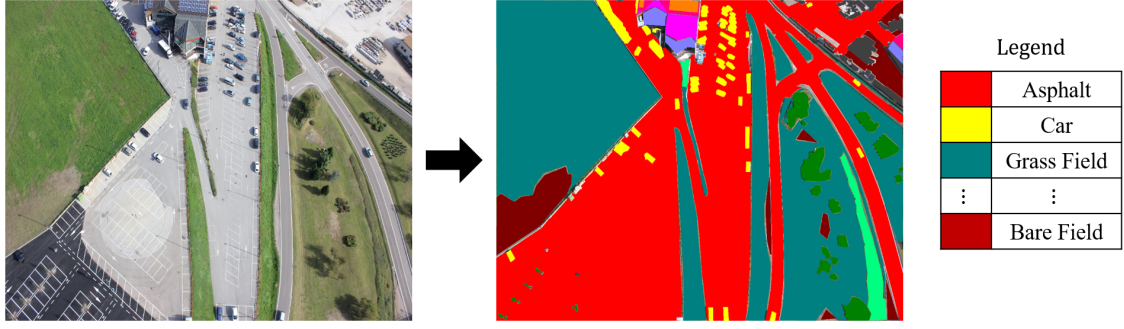


Figure 1.2: Example of land cover classification.

EHR imagery. The challenges have motivated us to develop new processing and analysis techniques to extract the full benefits of the abundant information contained in EHR imagery.

Hereafter, this dissertation mainly focuses on *land cover classification* analysis. The objective is to create land use maps on images. Figure 1.2 shows an example. Given an image, we assign class labels to certain regions defined by pixels, grids, or segments. Each class label represents what each region contains, e.g., asphalt, cars, grass fields, and so on. Applications in land cover classification have been constantly and widely developed, for purposes such as environmental change monitoring, urban design, and natural resource management.

*Unmixing*, a similar concept as land cover classification, also conveys land cover information. Classification just allocates each pixel into one of land cover categories, whereas unmixing outputs a constituent of categories in each pixel as a ratio. Unmixing is mainly applied to low-spatial resolution images in which even a single pixel covers a wide area of a land, see a review paper Bioucas-Dias et al. (2012).

### 1.2.1 Problem Formulation

Land cover classification is a well-studied topic in the remote sensing community, and many refined approaches have been proposed so far. For instance, various reports in the literature (Gao et al., 2016; Liu et al., 2016; Gu et al., 2017) adopted kernel-based classification approaches for the task. One of the most promising machine learning tools in the past several years, the deep neural network (NN), has also been applied to land cover classification. Makantanis et al. (2015), Chen et al. (2016), and Li et al. (2017) all used deep convolutional neural networks (CNNs). These works

focused, however, only on pixel-wise classification on VHR imagery. Since VHR imagery has meter-level resolution, objects such as vehicles, buildings, and trees are detected as single pixels, thus allowing a simple pixel-wise classification framework. In contrast, EHR imagery acquired by cameras mounted on UAVs has considerably more accurate information with centimeter-level pixels. In this case, a single pixel cannot convey sufficient contextual information to detect objects such as vehicles, buildings, and trees, because such objects cover hundreds of pixels in EHR imagery. Considering this difficulty, we must adjust problem formulations for land cover classification on such imagery.

An alternative strategy is so-called object-, region-, or segment-based classification approaches. These approaches first segment objects and regions in an image and then assign labels to the segmented regions. These approaches have actually been applied to UAV imagery, as reported in Zhang et al. (2017); Audebert et al. (2016, 2017), which all used CNNs. One major downside of segment-based approaches is the heavy training workload. For instance, a promising, well-known CNN-based segmentation approach, SegNet (Badrinarayanan et al., 2017), requires images to be completely labeled pixel by pixel for supervised learning. The workload to produce ground truth images is thus considerable. Unsupervised segmentation approaches provide an alternative, but the segmentation quality significantly affects the classification performance after segmentation.

Another approach, called the tile-based approach, first subdivides images into a non-overlapping grid of tiles of equal size and then labels each tile. If we subdivide images at 2-cm resolution into tiles of size  $50 \times 50$ , each tile covers an area of  $1\text{m} \times 1\text{m}$ . Such tiles can give enough information for classification. The workload for ground truth production is less than that of segment-based approaches. The tile-wise classification scheme was first proposed for EHR image analysis by Moranduzzo et al. (2015), who succeeded in making visually understandable classification maps. A downside of the tile-wise classification approach, however, is that it recasts land cover classification tasks to a more complicated problem formulation, namely multilabel classification, because each tile can hold more than one class label. The situation thus completely differs from segment-based classification and pixel-wise classification applied to VHR imagery.

### 1.2.2 Multilabel Classification

Multilabel classification is known as one of the most challenging tasks in pattern recognition because, unlike multiclass cases in which each sample belongs to only one class, multilabel outputs can exhibit more than one label simultaneously. In its mathematical formulation, the output label of multilabel classification can be denoted as  $\mathbf{y} \in \{0, 1\}^c$ , where  $c$  is the number of classes. The  $k$ -th element  $y_k = 1$  if its corresponding instance belongs to the  $k$ -th class and 0 otherwise. The cardinality of the output space grows exponentially in the number of classes. This exponentially growing output space not only causes the so-called “curse of dimensionality” but also makes the structure of classification itself complex, as explained later in this subsection.

Multilabel classification approaches are roughly categorized into two main groups: those based on problem transformation, and those based on algorithm adaptation. The problem transformation approach is more intuitive than algorithm adaptation, with the underlying idea of decomposing a multilabel problem into a set of binary classification tasks. After decomposition, a group of binary classifiers is trained independently (i.e., one classifier per label), and their predictions are combined to yield the final output. Boutell et al. (2004) introduced a novel multilabel learning strategy called *cross-training* or *binary relevance* by using output space decomposition. Another problem transformation approach, called *label power set*, considers each label combination in a training set as an identifier of a new distinct class. For example, if we conduct multilabel classification with two classes in the label power set formulation, then the output labels  $\mathbf{y} \in \{(1, 0), (0, 1), (1, 1)\}$  are treated as distinct classes  $\{C_1, C_2, C_{1-2}\}$ , and binary classification is then applied to each of the class pairs. The main problem here is the limitation of samples containing all label combinations in the training set.

The other multilabeling scheme, algorithm adaptation, exploits direct extensions of binary or multiclass classification algorithms to tackle multilabel classification. For instance, Zhang (2009) developed the radial basis function neural network (RBFNN) for multilabeling based on a thresholding rule at the network output layer, instead of simply considering the arguments of maxima. They also extended  $k$ -nearest neighbor algorithms for multilabeling (Zhang and Zhou, 2007). Their predictions are determined by the nearest samples to a test instance and maximum a posteriori (MAP) estimation.

As a common problem in multilabel classification based on problem transformation or algo-

rithm adaptation, correlation among labels is often ignored. Label relationships, however, meaning the structure of the output space, play a critical role especially in multilabel classification to greatly enhance its performance.

### **1.2.3 Output Structure**

In the context of multilabel classification, output structure can also be understood as the possibility of label coexistence or co-absence. As an example of a potentially structured output space in land cover classification, consider multilabel classification with three classes: “Asphalt,” “Car,” and “Grass.” Natural intuition suggests that Car labels should coexist with Asphalt labels more frequently than with Grass labels, which can also be confirmed in Fig.1.2. Such inter-label relationships are of little concern in multiclass classification. In the case of multilabel classification, however, the output structure is a serious factor that cannot be disregarded. To better exploit correlation among labels in a multilabel setting, for instance, Alvares-Cherman et al., (2012) developed a multilabel classifier that incorporates label dependency into the binary relevance framework. Their experiments confirmed that incorporating label dependency can improve classification performance.

Another key contribution was the derivation by Tsochantaridis et al. (2005) of multilabeling based on problem adaptation. They developed a structural support vector machine (structural SVM) framework for structural learning. Whereas the original SVM was implemented for binary classification, a structural SVM can be applied even to multilabel classification. The adaptation of SVMs to structural learning is straightforward, because structural SVMs still follow almost the same principles as binary SVMs. Note that the applications of structural SVMs are not limited to classification: they can also handle pattern recognition challenges with a complicated output structure, such as speech recognition (Le et al., 2011), sequence alignment (Joachims et al., 2006), and object tracking (Hare et al., 2016). The experimental assessments of the tasks all demonstrated the effectiveness of incorporating output structure.

### **1.2.4 Spatial Contiguity**

The issue of spatial contiguity is a considerable factor for not only EHR image analysis but also general remote sensing image analysis. In the case of pixel- or tile-wise classification, it is natural

to assume that each pixel/tile should give predictions having more similarity with its neighborhood than with any other randomly picked pixels/tiles. As shown in Fig.1.2, the region with Asphalt labels covers a contiguous area. Spatial dependency among inputs absolutely exists in land cover classification. Applications of basic statistical or machine learning algorithms directly to remote sensing imagery cannot address this principle, because these algorithms often assume that their inputs (i.e., pixels or tiles in land cover classification) were independently sampled from a distribution. Taking spatial dependency or neighborhood information into consideration, however, would definitely be useful in land cover classification. There are two types of approaches to embed such spatial information.

The first type of approach is to create or extract spatial features containing spatial context before classification. The most intuitive such approach is patch-based training. In pixel-wise classification based on patches, we extract a certain size of patches as inputs, e.g., a  $3 \times 3$  patch. The center pixel contains the spectral information of the target pixel, while the eight surrounding pixels have the information from the neighboring pixels. In patch-based training, therefore, each input is given as a certain size of cuboid. By applying feature extractors such as CNNs to these cuboids, we can extract common features among neighbors, resulting similar predictions with adjacent pixels (Chen et al., 2016; Li et al., 2017; Shu et al., 2018).

The second type of approach directly forces smoothness of predictions during classification. For instance, to enhance smoothness among a target pixel and its neighbors, Nishii and Eguchi (2004) formulated a context classifier, called AdaBoost (Adaptive Boosting), based on posteriors in neighborhoods. Reports in the literature (Dunder et al., 2006; Flamary and Rakotomamonjy, 2013) also include SVM-based spatial regularization approaches for binary land cover classification, in which prediction discrepancies among neighbors are penalized. In each of these cases, experiments verified the efficiency of spatial embedding.

### **1.2.5 Hyperspectral Imaging**

The last challenge described in this dissertation concerns hyperspectral imaging, which is a passive sensing technology (see, e.g., Wu and Sun (2013)). Beyond the visible spectrum for humans, hyperspectral imagery (HSI) contains both visible and non-visible spectral information from 50-500 bands with a narrow spectral bandwidth. Furthermore, ultraspectral imaging can now capture

spectral information with an even narrower band resolution than that of HSI. The abundant spectral information available in HSI enables identification of seemingly similar objects that cannot be distinguished in visible or even multispectral images.

Unfortunately, the large number of bands in HSI still causes the so-called “ $n \ll p$  problem,” even though many analytical tools to cope with this problem have been developed in the statistics and machine learning fields. This is because, for remote sensing data, samples with ground truth are not readily available. Another aspect of this problem is that new platforms like UAVs can make the situation even more serious. For instance, what if we perform multilabel classification under the label power set framework? Labeled samples are required for every label pattern in the huge output space, but the workload to create ground truth labels is considerable. Therefore, the  $n \ll p$  problem is still a worrisome challenge in the remote sensing community, and we need to develop tools to meet this challenge.

One potential solution is feature extraction/selection/reduction, meaning that we represent inputs only with low-dimensional, discriminative features extracted/selected/transformed from the original input space. Unsupervised feature extraction is especially useful in meeting such requirements in remote sensing data analysis (see, e.g., Romero et al. (2016)).

### 1.3 Our Strategy

Given the aforementioned challenges in current remote sensing data analysis, this dissertation examines how to fix them.

As a first topic, we focus on land cover classification on EHR images and discuss efficient approaches for their analysis, which is related to the discussion in **sections 1.2.1-1.2.4**. The strategy is as follows. To create reasonable classification maps on such images, the tile-based approach proposed by Moranduzzo et al. (2015) is applied as the problem formulation. This means that multilabel classification is formulated as discussed in **section 1.2.2**. Furthermore, the output structure and spatial contiguity should be embedded into classification to achieve better outcomes, as discussed in **sections 1.2.3 and 1.2.4**, respectively. To meet all these requirements, a novel structural and spatial multilabel classifier was developed, based on the structural SVM formulation. Structural SVMs can naturally cope with multilabel classification problems containing structural output relations. The structural SVM was modified to also incorporate spatial dependency among tiles.

Concretely speaking, a term was added into the structural SVM objective function to penalize prediction discrepancies among neighbors in test images. The training process of the new SVM model is efficient because its optimization retains the basic characteristics of the traditional binary SVM, meaning that it can be formulated as a quadratic programming problem. Therefore, a useful multilabel classifier satisfying the properties of both output structure and spatial contiguity can be learned in a single training process. This approach overcomes the difficulties of land cover classification on EHR imagery.

As a second topic, we discuss feature learning in HSI, which relates to the discussion in **sections 1.2.4** and **1.2.5**. Following the common situation of remote sensing data analysis as presented in **section 1.2.5**, a novel unsupervised feature extraction approach was designed, based on an autoencoder (AE) neural network. The important point is that the AE-based feature extractor also takes spatial dependency into consideration. By modifying the loss function of a normal AE, we can extract contextual features from HSI and force neighboring inputs/pixels to have similar feature representations. The extracted features make further analysis more widely applicable, such as for classification, characterization, and object detection implemented on HSI.

In summary, the main contributions of this research are the following:

1. A structural and spatial multilabel classifier for land cover classification on EHR imagery.
2. An unsupervised spatial and spectral feature extraction approach for HSI analysis.

## 1.4 Overview

This rest of the dissertation is organized as follows.

**Chapter 2** outlines the overall strategy for EHR image analysis. First, we introduce the tile-based problem formulation called *coarse description* in detail. As the second step of this strategy, we review the deep-CNN-based feature representation approach known as GoogLeNet (Szegedy et al., 2015) and discuss its capability for feature extraction. Then, to lay the groundwork of SVM-based structural learning for multilabel classification, as described in the following chapter, we review the basics of traditional SVMs: binary and multiclass SVMs.

In **chapter 3**, we review the mathematical formulation of the structural SVM. We discuss its related theories and a training algorithm based on the *cutting-plane algorithm* (Thorsten et

al., 2009). Applications of structural SVMs developed in the remote sensing community are also described.

**Chapter 4** introduces a novel classification algorithm for EHR imagery. We describe how to naturally incorporate spatial dependency into the structural SVM formulation, and training algorithms are examined here, as well. The results of numerical experiments with both artificial and real-world datasets show the effectiveness of embedding spatial information into classification.

In **chapter 5**, both the new approach and conventional approaches are applied to multiple real-world EHR image datasets. We compare their performances and show the capability of the proposed methodology both quantitatively and qualitatively in this chapter.

In contrast, the focus of **chapter 6** is not EHR imagery but HSI. We discuss unsupervised feature extraction approaches developed on spectral imagery. After reviewing AE networks, we introduce a modified AE designed to extract spatial and spectral features from HSI. We discuss the results of numerical experiments on six real-world HSI datasets and assess the potential capability of the AE-based feature extractor for land cover classification tasks.

**Chapter 7** summarizes the work in this dissertation on remotely sensed data analysis.

In addition, **chapter 8** discusses another topic: an application of statistical analysis to plant genome science. The goal here was to elucidate a phenomenon in plants called *heterosis* at the molecular level, through a combination of statistical and botanical analysis. To achieve this objective, gene regulatory networks (GRNs) were estimated on time series genomic data from plants by using a statistical sparse analysis tool based on group smoothly clipped absolute deviation (SCAD). We report the results of both network estimation and biological observations derived from the statistical analysis.

## Chapter 2

# Problem Formulation and Tools

This chapter provides a detailed description of the analytical strategy used here. It consists of three steps: coarse description, feature extraction, and classification. We review these steps in **sections 2.1, 2.2, and 2.3**, respectively.

### 2.1 Coarse Description

Consider EHR images in which all pixels are characterized by visible RGB channels. As reviewed before, traditional schemes for land cover classification on VHR imagery cannot reveal intrinsic information contained in EHR images. As an alternative, Moranduzzo et al. (2015) proposed the concept of so-called “coarse” description. The underlying idea is to formulate a list of possible classes of objects in a subdivided area of an image, instead of assigning class labels to single pixels individually.

Figure 2.1 shows the workflow for this idea. In tile-wise classification, an image is first subdivided into tiles of a certain size. Each tile is represented by a set of pixels, i.e., a cuboid, and a corresponding binary descriptor label indicates the presence or absence of objects. The goal is to create land cover maps on unknown test images. Test images are subdivided into tiles of the same size as those of training images, and then predictions for each tile are given by binary descriptors, in the same manner as with training data. The final output of land cover maps consists of a set of maps for each class, as shown at the bottom of Fig.2.1.

A drawback of the coarse description is that it turns classification problems into multilabel classification. Another drawback is that we lose the precise spatial locations of objects. Reducing

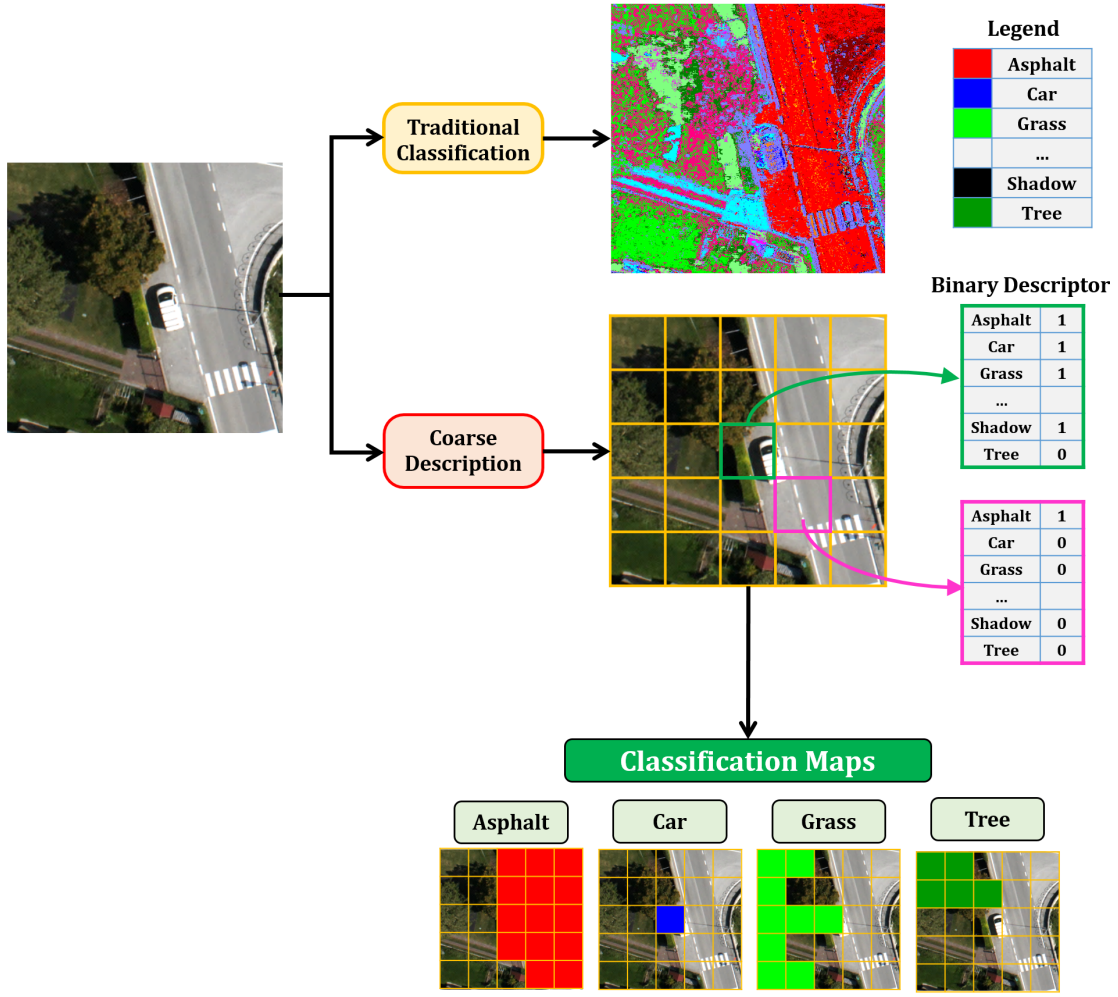


Figure 2.1: Workflow of tile-wise classification.

the tile size can mitigate the effect of the latter drawback. The tile size should be defined according to the spatial resolution of the imagery and the expected sizes of the target objects to recognize. Although appropriate optimization of the tile size might provide better recognition performance, we do not explore this aspect here because it is outside the scope of the principal objective.

## 2.2 Feature Extraction via Neural Networks

What we need for better classification performance is discriminative tile representation: we need to select, extract, and transform a set of features from original tile inputs. The extracted feature set should be distributed separately by class on a manifold in the feature space. Traditionally, computationally efficient, handcrafted features like the scale-invariant feature transform (SIFT) or

histogram of gradients (HOG), and their integration by applying the bag-of-words model, have been used to describe the texture and gradient of images. Current approaches instead focus on applying neural networks (NNs), particularly of the CNN sub-type, as feature extractors capable of obtaining deep, hierarchical features from images.

NNs are mathematically modeled information processing flows derived from biological systems. A neuron corresponds to a node in an NN. Each node receives information from its previous nodes, aggregates them, and sends the aggregate information to subsequent nodes via a nonlinear transformation. An NN thus aims to achieve pattern recognition by mathematically imitating the recognition processes of the human brain.

Let us denote an input of nodes as  $\mathbf{x} \in \mathbb{R}^d$ , and a network weight and bias as  $\mathbf{w}_j \in \mathbb{R}^d$  and  $b_j \in \mathbb{R}$  ( $j \in \{1, \dots, l\}$ ), respectively. The aggregate information received by a subsequent neuron  $z_j$  can be expressed as

$$z_j = a(\mathbf{w}_j^T \mathbf{x} + b_j), \quad (2.1)$$

where the nonlinear function  $a(\cdot)$  is called an *activation function*. A set of neurons,  $\{z_j\}_{j=1}^l$ , comprises a *layer*. By concatenating tens of layers, we can make the network express even complicated input-output relationships. In the case of classification, the network becomes a nonlinear classifier, whose output layer generates class labels. The network weights are trained so that the network yields reasonable outputs.

Since 2012, researchers have developed many excellent NN techniques, e.g., the rectified linear unit (ReLU) activation function, batch normalization, residual network architecture, and so forth (see e.g., Krizhevsky et al. (2012); Ioffe and Szegedy (2015); He et al. (2016)). These techniques have all relieved various difficulties with deep NN training, such as the vanishing gradient problem and covariate shift. Simultaneously, they have contributed toward accelerating network training. Image recognition competitions have particularly demonstrated the great capability of deep CNNs. Recognition performance has rapidly improved year by year since 2012. This ensures that deep CNNs can offer discriminative feature representation capable of making classification very accurate.

One drawback of deep CNNs is their training cost. A deep network architecture requires a sufficient number of training samples and amount of computational power, but such requirements are not always satisfied easily in the real world. Transfer learning, however, which involves reusing

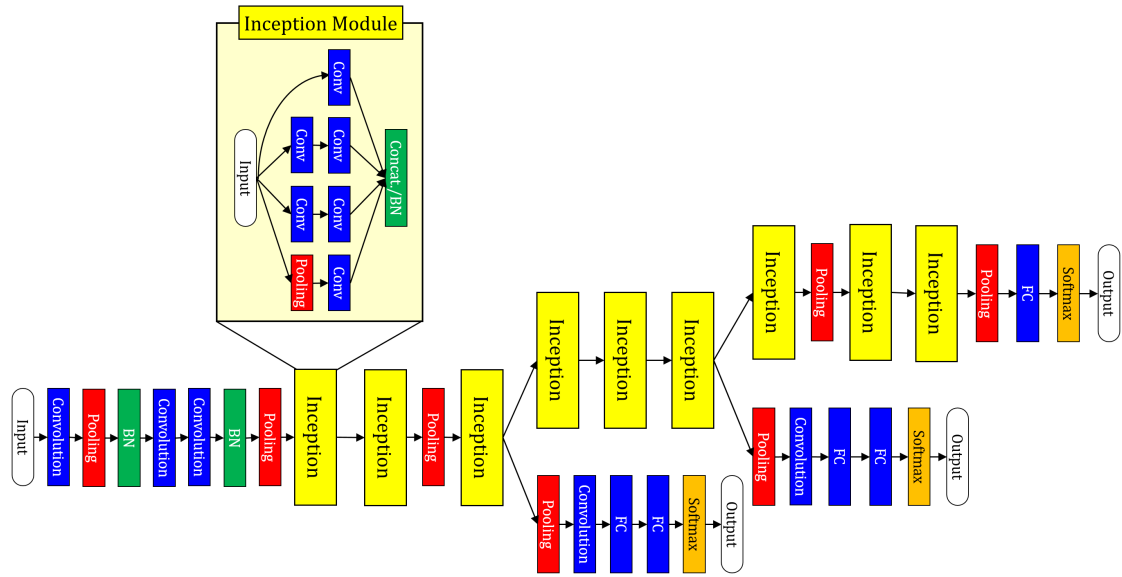


Figure 2.2: Network architecture of GoogLeNet.

and fine-tuning pre-trained networks for other tasks, can resolve this drawback. It also enables us to leverage the potential of promising networks.

As for the strategy used here, we apply a pre-trained NN called GoogLeNet as a tile feature extractor (Szegedy et al., 2015). This network was originally trained on the ImageNet ILSVRC2014 dataset, which contains over 1.2 million images in 1000 categories, demonstrating its promising capability for image recognition. Zeggada and Melgani (2016) previously reported the potential for feature representation on UAV imagery through transfer learning with GoogLeNet. GoogLeNet contains a 27-layer-deep network composed of convolution, pooling, and batch normalization (BN) layers, a unique component called an inception module, and then a fully connected (FC) layer with the softmax activation function. Figure 2.2 shows the entire network architecture.

### Convolution Layer

The convolution layer is the main building block of GoogLeNet. An important property of images is spatial correlation: nearby pixels can contain locally discriminative features. Also, such local features might be useful in other regions of an image. Even if an object (e.g., a car) is shifted to another region of an image, for instance, the features describing the object are preserved. Convolution layers take the responsibility of learning such invariant local features.

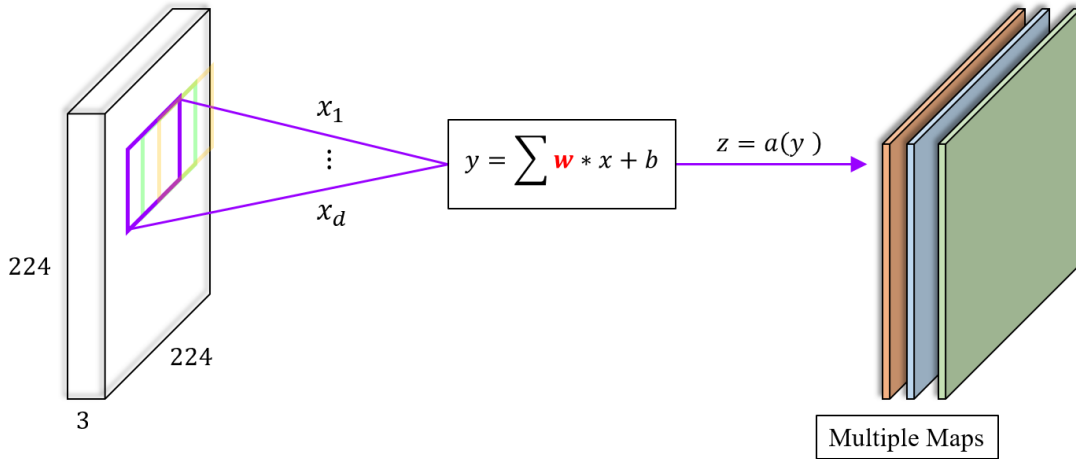


Figure 2.3: Graphical outline of a convolution layer.

The convolution layer has filters, which are learnable weights given as voxels of the size  $height \times width \times depth$ . The layer calculates a linear combination of the filter and an image subregion of the same size as the filter. Then, linear combinations with other image subregions are calculated by sliding the filter while sharing weights. This process results in a new map in which local regions are activated by the corresponding filters. Figure 2.3 illustrates these processes. By using multiple filters, we can activate different-shaped configurations from the inputs, resulting in multiple maps with different properties. In addition, by stacking convolution layers, we can extract features in a hierarchical manner. In such a stacked network, the convolution layers early in the network generally learn global features to discriminate superclasses or upper taxonomies. Later convolution layers then extract class-specific features to separate superclasses into subclasses. In this way the network offers a discriminative feature set and even forms an accurate classifier. The outputs of the convolution layers in GoogLeNet are activated by the ReLU function with the output threshold at zero.

### Pooling Layer

The pooling layer performs downsampling of inputs from the previous layers in spatial dimensions. Similar to the convolution layer, the pooling layer depends on sliding width and filter size parameters that control the spatial size of its outputs. For instance, when a filter of size  $2 \times 2$  is applied to inputs in a non-overlapping, sliding manner, the size of its outputs is halved. The pool-

ing layers play a role in desensitizing the response to small shifts of objects, i.e., in controlling overfitting. Simultaneously, they reduce the numbers of calculation operations and parameters, reducing the cost of network training.

The most commonly used pooling operation in CNNs is max pooling, which only propagates the maximum response of receptive fields to the following layers. The other common pooling operation (e.g., in residual networks) is average pooling, which propagates the average response of receptive fields.

### **Batch Normalization**

Ioffe and Szegedy (2015) proposed the technique called batch normalization (BN) in 2015 after the first version of GoogLeNet had been released. Most currently developed deep NNs include BN layers. The impact of the BN layer is significant: it both accelerates network training and increases the generalization performance of deep NNs. Although the technique called *dropout*, which is a sort of ensemble learning, was known as an efficient way of avoiding overfitting, the BN layer plays a similar role to a dropout layer while also speeding up training.

The underlying idea of BN is to reduce the influence of *internal covariate shift*. The term concerns the change of distributions of nodes inside networks. If the distribution of internal nodes changes step by step during training, network parameters must be adjusted at every step according to these unstable changes. This problem becomes especially serious for deep networks, because even small changes are amplified during propagation through their deeply stacked layers. Just as normalization of inputs can accelerate training in ordinary pattern recognition, normalization of internal nodes can induce faster training of deep networks.

Let us denote a mini-batch of size  $m$  as  $\mathcal{B} = \{x_i\}_{i=1}^m$ , which is a set of inputs at any internal layer. We assume here that the inputs are scalars, because we normalize them at each activation.

The BN layer then produces outputs  $z_i = \text{BN}_{\gamma, \beta}(x_i)$  as follows.

$$\begin{aligned}\mu_{\mathcal{B}} &\leftarrow \frac{1}{m} \sum_{i=1}^m x_i \\ \sigma_{\mathcal{B}}^2 &\leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 \\ \hat{x}_i &\leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \\ z_i &\leftarrow \text{BN}_{\gamma, \beta}(x_i) = \gamma \hat{x}_i + \beta.\end{aligned}$$

Here,  $\gamma$  and  $\beta$  are learnable parameters to enable identity mapping. The BN operation is normally applied after convolution.

In the test phase, the mean and variance values accumulated during training are used instead of those of mini-batches.

### Inception Module and Auxiliary Classifiers

The outstanding recognition capability of GoogLeNet derives from its unique component, the *inception module*, shown in Fig.2.2. It consists of convolution layers with various filter sizes, namely  $1 \times 1$ ,  $3 \times 3$ , and  $5 \times 5$ , and a pooling layer. The convolution layers with different filter sizes can capture various sizes of local input configurations. The  $1 \times 1$  convolution layer before the  $3 \times 3$  and  $5 \times 5$  convolution layers plays the roles of reducing the number of weights and accelerating network training. The original GoogLeNet architecture contains nine inception modules.

The other components of GoogLeNet are called *auxiliary classifiers*, which are output layers inserted midway in the network. These not only evaluate classification performance with respect to features at lower stages but also contribute to combatting the vanishing gradient problem by adding the gradient calculated at the intermediate layers to the one propagated from the output layer.

We can use the pre-trained GoogLeNet as a feature extractor by letting each tile propagate through the network up to a fully connected layer, which generates a feature vector of length 1024. The size of the receptive field of GoogLeNet is  $224 \times 224 \times 3$ . When the tile size does not match the receptive field, we resize the inputs appropriately through interpolation.

## 2.3 Classification

Once we represent tiles with features extracted from GoogLeNet, we pass them to a classifier. Among many classification algorithms based on statistics and machine learning, this section reviews SVMs, including both binary and multiclass SVMs.

### 2.3.1 Support Vector Machine

The SVM is known as one of the most promising machine learning tools for pattern recognition, including classification analysis (Cortes and Vapnik, 1995; Crammer and Singer, 2001). It was originally implemented for binary classification, in which an SVM model learns the boundary hyperplane separating two classes. An important aspect of the SVM is the so-called *kernel trick*, whose details will be described below. By applying the kernel trick, we can perform nonlinear classification through computationally friendly training. This promising property has made the SVM a widely used pattern recognition tool even in remote sensing data analysis (Melgani and Bruzzone, 2004; Mountrakis et al., 2011).

In general, the goal of an SVM (not restricted to a binary SVM) is to find a discriminant function  $g : \mathcal{X} \rightarrow \mathcal{Y}$  in a given hypothesis space  $\mathcal{H}_0$ , where  $\mathcal{X}$  and  $\mathcal{Y}$  denote the input and output spaces, respectively. Typically, each element of  $\mathcal{Y}$  denotes a class label. The discriminant function  $g$  is parameterized with a weight vector  $\mathbf{w}$  as  $g(\mathbf{x}) = g(\mathbf{x}; \mathbf{w})$ . Suppose that input-output pairs  $(\mathbf{x}, \mathbf{y})$  are sampled from a distribution  $P(\mathcal{X}, \mathcal{Y})$ . The goal is then to find the function  $g(\mathbf{x}, \hat{\mathbf{w}})$  that minimizes the expected loss,

$$R(\mathbf{w}) = \int_{\mathcal{X} \times \mathcal{Y}} \Delta(\mathbf{y}, g(\mathbf{x})) dP(\mathbf{x}, \mathbf{y}), \quad (2.2)$$

where the loss function  $\Delta : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}_{\geq 0}$  quantifies the loss associated with  $\mathbf{y}$  and its predictions. The loss  $\Delta(\mathbf{y}, \bar{\mathbf{y}}) = 0$  when  $\mathbf{y} = \bar{\mathbf{y}}$ . Because  $P$  is unknown, we learn the optimal function  $g \in \mathcal{H}_0$  in a supervised learning context. Given a training set  $D = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$  drawn in an independent and identically distributed (i.i.d.) manner from  $P(\mathcal{X}, \mathcal{Y})$ , we minimize the following empirical loss:

$$R_{emp}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, g(\mathbf{x}_i)). \quad (2.3)$$

The SVM selects the function  $g$  that minimizes the regularized empirical risk on the training set  $D$ . In other words, the training process minimizes the empirical loss  $R_{emp}(\mathbf{w})$  while maximizing the

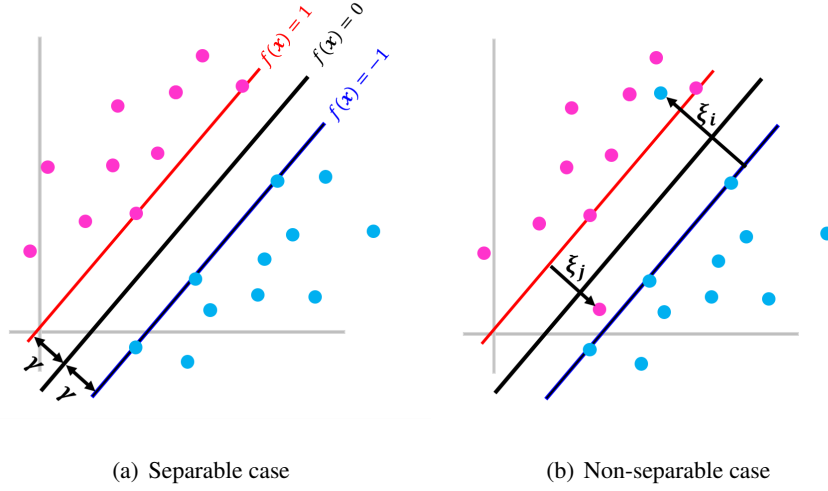


Figure 2.4: Illustrations of binary SVMs: the separable case (a) and non-separable case (b). The red and blue circles represent samples assigned to classes  $C_1$  and  $C_{-1}$ , respectively. The black line shows the decision boundary. The margin  $\gamma$  is defined as the perpendicular distance to the closest samples from the decision boundary, as shown in (a). If the classification is not separable, as shown in (b), then penalties  $\xi$  are imposed according to the margin violations.

*margin*, which is defined as the distance between the boundary hyperplane and the closest samples to it. This interpretation corresponds to the structural risk minimization (Vapnik, 1999). Maximizing the margin increases the generalization capability of a classifier. That is, the margin controls the functional complexity of the discriminant function as measured by the Vapnik-Chervonenkis (VC) dimension.

### 2.3.2 Binary Support Vector Machine

Let us first consider the binary classification case, in which  $\mathcal{X} = \mathbb{R}^d$  and each output  $y$  is a scalar in  $\mathcal{Y} = \{+1, -1\}$  representing a class label. Suppose that we aim to classify each sample  $\mathbf{x} \in \mathcal{X}$  into one of two classes,  $C_1$  and  $C_{-1}$ , with a function  $g : \mathcal{X} \rightarrow \mathcal{Y}$ . We define a function  $f : \mathcal{X} \rightarrow \mathbb{R}$  as

$$f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b, \quad (2.4)$$

where  $\mathbf{w} \in \mathbb{R}^p$  and  $b \in \mathbb{R}$  are the weights and a bias parameter, respectively, and  $\phi : \mathcal{X} \rightarrow \mathbb{R}^p$  is a feature mapping that maps an original  $d$ -dimensional input  $\mathbf{x}$  to a  $p$ -dimensional feature space. The function  $f(\mathbf{x})$  then gives a score for input  $\mathbf{x}$ . Samples are assigned to one class or the other

according to the sign of the score  $f(\mathbf{x})$ , i.e., the discriminant function is

$$g(\mathbf{x}) = \text{sign}(f(\mathbf{x})). \quad (2.5)$$

The contour given by  $f(\mathbf{x}) = 0$  represents the decision hyperplane.

As mentioned before, the optimal parameter is derived by minimizing the empirical risk and simultaneously maximizing the generalization capability of the trained classifiers. This capability is determined by the margin. Figure 2.4(a) shows an example of separable binary classification, in which the margin  $\gamma$  is defined as the perpendicular distance between the decision boundary  $f(\mathbf{x}) = 0$  and the closest samples. We say that a training set  $D$  is linearly separable if there exists a discriminant function that can separate the two classes with a decision boundary that is linear in the mapped space. This situation is mathematically described by the following condition:

$$\forall i \in \{1, \dots, n\} : y_i f(\mathbf{x}_i) \geq 1. \quad (2.6)$$

Separable classification requires maximizing the margin under the given conditions. Because the margin  $\gamma$  is given by  $\frac{1}{\|\mathbf{w}\|}$ , the margin maximization formulates the following primal optimization problem:

$$\begin{aligned} \min_{\mathbf{w}, b} \quad & \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{s.t.} \quad & \forall i : y_i f(\mathbf{x}_i) \geq 1. \end{aligned} \quad (2.7)$$

This is called a *hard-margin SVM*.

In contrast, the general situation, includes cases that are not linearly separable, as shown in Fig.2.4(b). To cope, we define slack variables  $\xi_i = \min\{0, 1 - y_i f(\mathbf{x}_i)\} \geq 0$  for each sample. Their role is to linearly impose penalties on boundary violations. The corresponding loss is called the *hinge loss*, which is the upper bound of the indicator loss function. We replace the optimization problem defined by Eq.(2.7) with the following primal optimization problem:

$$\begin{aligned} \min_{\mathbf{w}, b, \boldsymbol{\xi}} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & \forall i : y_i f(\mathbf{x}_i) \geq 1 - \xi_i \text{ and } \xi_i \geq 0, \end{aligned} \quad (2.8)$$

where  $\boldsymbol{\xi} = (\xi_1, \dots, \xi_n)$  and the constant  $C > 0$  controls the trade-off between the regularization and empirical error. This formulation is called a *soft-margin SVM*. It can be interpreted from the Bayesian perspective as the maximum a posteriori (MAP) estimation of the corresponding

probabilistic model with a Gaussian prior on  $\mathbf{w}$ . Therefore, we can readily understand that the inverse of the margin plays a role of regularization in Bayesian statistics (Sollich, 2002).

### Dual Formulation of SVM

To efficiently solve the primal problem of Eq.(2.8), we reformulate it with the Karush-Kuhn-Tucker (KKT) theorem (see, e.g., Vapnik (1999); Bishop (2006)). The Lagrangian of the dual problem is given by

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i - \sum_{i=1}^n a_i \{y_i f(\mathbf{x}_i) - 1 + \xi_i\} - \sum_{i=1}^n \mu_i \xi_i, \quad (2.9)$$

where  $\{a_i \geq 0\}$  and  $\{\mu_i \geq 0\}$  are Lagrange multipliers for each constraint. The saddle point of the Lagrangian is the minimum for the primal variables and the maximum for the dual variables.

We now optimize out the primal variables, resulting in the following dual problem:

$$\begin{aligned} \max_{\mathbf{a}} \quad & \tilde{L}(\mathbf{a}) = \sum_{i=1}^n a_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n a_i a_j y_i y_j \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) \\ \text{s.t. :} \quad & \sum_{i=1}^n a_i y_i = 0 \quad \text{and} \quad 0 \leq a_i \leq C, \quad \forall i \in \{1, \dots, n\}. \end{aligned} \quad (2.10)$$

By solving the quadratic programming problem of the dual, we can obtain the optimal dual variables and thus the primal variable.

It is important to note that the dual problem depends only on inner products of feature vectors. Thus, we can replace them with the kernel function  $K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ ; that is, we can avoid the time-consuming inner product calculation in high-dimensional feature space. This is the kernel trick. The score function can be derived as

$$\begin{aligned} f(\mathbf{x}) &= \mathbf{w}^T \phi(\mathbf{x}) + b \\ &= \sum_{i=1}^n a_i y_i K(\mathbf{x}_i, \mathbf{x}) + b, \end{aligned} \quad (2.11)$$

and the corresponding classifier becomes  $g(\mathbf{x}) = \text{sign}(f(\mathbf{x}))$ . Depending on the type of kernel function, the classifier draws a linear or nonlinear boundary hyperplane.

### 2.3.3 Multiclass Support Vector Machine

We now review multiclass SVMs to make the concepts of “multilabel” SVMs clearer in later chapters. Although the original SVM was designed to solve binary classification problems, extended

approaches for multiclass classification have been proposed in the literature. An intuitive approach is to decompose multiclass classification into a combination of binary classifications; examples include the one-versus-all SVM, one-versus-one SVM, and directed acyclic graph SVM (DAGSVM, Platt et al. (2000)). One proposed alternative to the decomposition approach is an approach using all-in-one machines, with which we can optimize multiclass classifiers by solving one quadratic programming problem. Such all-in-one approaches are further subdivided according to their formulation: Weston and Watkins (1999); Bredensteiner and Bennett (1999); Crammer and Singer (2001). Comparative studies from theoretical and empirical viewpoints have been reported in the literature (Hsu and Lin, 2002; Duan and Keerthi, 2005; Dogan et al., 2016). Class relationships are easier to incorporate into the formulation with all-in-one approaches than with decomposition approaches. Among the all-in-one approaches, the Crammer-Singer multiclass SVM (CS-SVM) is known to be efficient in its computational cost and memory use. Moreover, the CS-SVM is closely related to the structural SVM. Therefore, we now review multiclass classification based on the CS-SVM.

Suppose that we are given a training set  $D$ . The output space is given as  $\mathcal{Y} = \{1, \dots, c\}$ , where  $c$  denotes the number of classes. Every sample belongs to one of the  $c$  classes. The CS-SVM uses the classifier  $g : \mathcal{X} \rightarrow \mathcal{Y}$  given by

$$g(\mathbf{x}) = \arg \max_{k=1}^c \{ \mathbf{w}_k^T \phi(\mathbf{x}) \}. \quad (2.12)$$

The function  $f_k(\mathbf{x}) = \mathbf{w}_k^T \phi(\mathbf{x})$  defines the scores of input  $\mathbf{x}$  with respect to each class, and the prediction is determined as the argument with the highest score. This is a generalization of the binary SVM, in which the prediction is determined by  $\text{sign}(\mathbf{w}^T \phi(\mathbf{x}))$ . This can be interpreted in the multiclass view as  $\mathbf{w}_1 = \mathbf{w}$  and  $\mathbf{w}_2 = -\mathbf{w}$ .

Hereafter, let us define the weight  $\mathbf{w}$  as the series of weights associated with each class, i.e.,  $\mathbf{w} = (\mathbf{w}_1^T, \dots, \mathbf{w}_c^T)^T \in \mathbb{R}^{p \times c}$ .

The empirical loss based on the 0-1 loss (and the indicator function  $\mathbf{1}[\cdot]$ ),

$$R_{emp}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \mathbf{1}[y_i \neq g(\mathbf{x}_i)], \quad (2.13)$$

can be upper-bounded as

$$R_{emp}(\mathbf{w}) \leq \frac{1}{n} \sum_{i=1}^n \left[ \max_k \{ \mathbf{w}_k^T \phi(\mathbf{x}_i) + 1 - \delta_{y_i, k} \} - \mathbf{w}_{y_i}^T \phi(\mathbf{x}_i) \right], \quad (2.14)$$

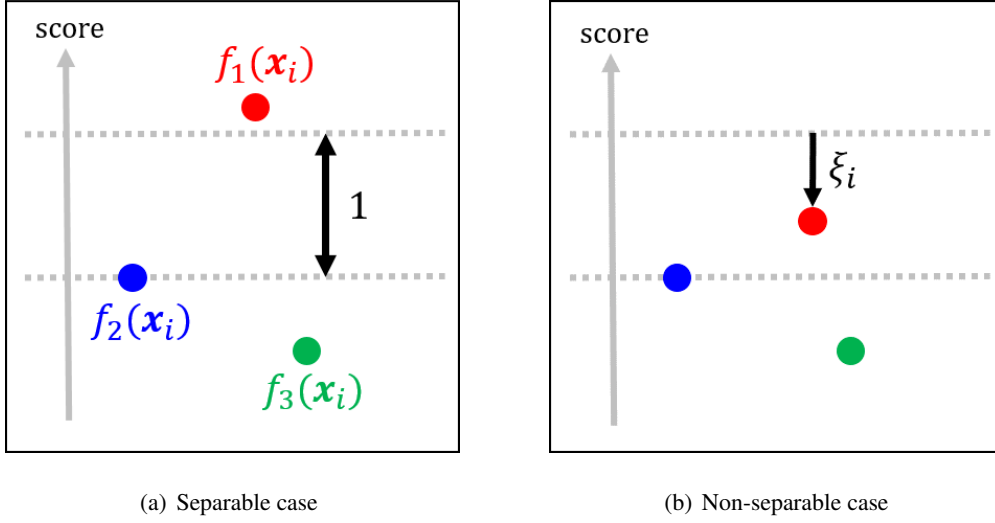


Figure 2.5: Illustrations of multiclass SVMs (Crammer and Singer, 2001): the separable case (a) and non-separable case (b). These examples assume three classes and that the true label of sample  $\mathbf{x}_i$  is 1. As shown in (a), complete classification requires that the score of the true class  $f_1(\mathbf{x}_i)$  must be greater by at least one than those of the other two classes. Otherwise, a positive penalty is imposed, as shown in (b).

where  $\delta_{y_i,k}$  is equal to 1 if  $y_i = k$  and 0 otherwise. Suppose that the above bound is equal to zero, that is,

$$\forall i : \max_k \{ \mathbf{w}_k^T \phi(\mathbf{x}_i) + 1 - \delta_{y_i,k} \} - \mathbf{w}_{y_i}^T \phi(\mathbf{x}_i) = 0. \quad (2.15)$$

This would mean that the score associated with the true class  $\mathbf{w}_{y_i}^T \phi(\mathbf{x}_i)$  must be greater by at least one than those of the other classes,  $\mathbf{w}_k^T \phi(\mathbf{x}_i)$  ( $k \in \mathcal{Y} \setminus \{y_i\}$ ). Figure 2.5(a) shows an illustration of this relationship with three classes. The score of the true class,  $f_1(\mathbf{x}_i) = \mathbf{w}_1^T \phi(\mathbf{x}_i)$  in this example, is greater by at least one than those of the other two classes,  $f_2(\mathbf{x}_i)$  and  $f_3(\mathbf{x}_i)$ . The weight  $\mathbf{w}$  satisfying Eq.(2.15) would also satisfy the following:

$$\forall i, k : \mathbf{w}_{y_i}^T \phi(\mathbf{x}_i) + \delta_{y_i,k} - \mathbf{w}_k^T \phi(\mathbf{x}_i) \geq 1. \quad (2.16)$$

To cope with non-separable classification, we add slack variables  $\xi_i \geq 0$  for every sample and modify Eq.(2.15) to

$$\forall i : \max_k \{ \mathbf{w}_k^T \phi(\mathbf{x}_i) + 1 - \delta_{y_i,k} \} - \mathbf{w}_{y_i}^T \phi(\mathbf{x}_i) = \xi_i. \quad (2.17)$$

The slack variable imposes a positive penalty on the margin violation, as shown in Fig.2.5(b). Also, the empirical loss is bounded as

$$R_{emp}(\mathbf{w}) \leq \frac{1}{n} \sum_{i=1}^n \xi_i. \quad (2.18)$$

The training process follows the same principle as the binary SVM: maximizing the margin and minimizing the empirical loss. The primal optimization problem of the multiclass SVM is given by

$$\begin{aligned} \min_{\mathbf{w}, \boldsymbol{\xi} \geq 0} \quad & \frac{1}{2} \sum_{k=1}^c \|\mathbf{w}_k\|^2 + C \sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & \forall i, k : \mathbf{w}_{y_i}^T \boldsymbol{\phi}(\mathbf{x}_i) + \delta_{y_i, k} - \mathbf{w}_k^T \boldsymbol{\phi}(\mathbf{x}_i) \geq 1 - \xi_i. \end{aligned} \tag{2.19}$$

As with the binary SVM, the dual problem of Eq.(2.19) can be derived by using the KKT theorem in a similar way. Kernelization is still possible, as well. As this can also be interpreted as an example of a structural SVM, we review that approach in the following chapter.

Note that the bias term is omitted in the CS-SVM for the sake of simplicity, but we can easily include it by coordinating the formulation (see, e.g., Crammer and Singer (2001)).

## Chapter 3

# Structural Learning

Modeling functional dependency in input and output spaces is challenging but often makes pattern recognition problems more practical (Tsochantaridis et al., 2005). Consider data contain with an output space of complicated structures such as trees, sequences, strings, graphs, or lattices. This situation is very common in real-world problems: taxonomic categorization (Navid et al., 2015), speech recognition (Le et al., 2011), sequence alignment (Joachims et al., 2006), object tracking (Hare et al., 2016), and so forth.

This chapter describes a major structural learning algorithm, the structural SVM. After reviewing its mathematical formulation and applications, we introduce an efficient training algorithm for structural SVMs, the cutting-plane algorithm.

### 3.1 Structural Support Vector Machine

As with a generalized SVM, the goal of a structural SVM is still to find a discriminant function  $g : \mathcal{X} \rightarrow \mathcal{Y}$  in a given hypothesis space  $\mathcal{H}_0$ . That is, we minimize the expected loss,

$$R(\mathbf{w}) = \int_{\mathcal{X} \times \mathcal{Y}} \Delta(\mathbf{y}, g(\mathbf{x})) dP(\mathbf{x}, \mathbf{y}), \quad (3.1)$$

or, practically, its empirical loss,

$$R_{emp}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, g(\mathbf{x}_i)). \quad (3.2)$$

In a structural SVM, the discriminant function is defined on input-output pairs as

$$g(\mathbf{x}) = \arg \max_{\mathbf{y} \in \mathcal{Y}} \{f(\mathbf{x}, \mathbf{y})\}. \quad (3.3)$$

The score function  $f : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$  is thus defined on the joint input-output space and evaluates scores for pairs. The function is parameterized as

$$f(\mathbf{x}, \mathbf{y}) = \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}, \mathbf{y}), \quad (3.4)$$

where  $\boldsymbol{\psi} : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{H}$  is called a joint feature map, which maps the input-output pairs jointly into a feature Hilbert space  $\mathcal{H}$ .

Unlike the SVMs described in the previous chapter, the score function of a structural SVM is defined on the input-output space. The score functions of these SVMs, however, are still related to each other. To describe their relation, consider the case of multiclass classification. Let us denote the output  $\mathbf{y}$  in the one-hot encoding style, i.e.,  $\mathbf{y} \in \{\mathbf{e}_1, \dots, \mathbf{e}_c\}$ , where  $\mathbf{e}_k$  is a unit vector whose  $k$ -th element is one. When we denote the weight as  $\mathbf{w} = (\mathbf{w}_1^T, \dots, \mathbf{w}_c^T)$  and define the joint feature vector as  $\boldsymbol{\psi}(\mathbf{x}, \mathbf{y}) = \mathbf{y} \otimes \phi(\mathbf{x})$ , the discriminant function becomes

$$g(\mathbf{x}) = \arg \max_{k=1}^c \{\mathbf{w}_k^T \phi(\mathbf{x})\}. \quad (3.5)$$

This functional form exactly corresponds to the CS-SVM discriminant function given by Eq.(2.12). In this way, any SVM formulation can be generalized with the flexible designs of the joint feature map.

Let us return to structural SVMs. The weight  $\mathbf{w}$  can be optimized in the same way as with basic SVMs. First, assume that the samples in a training set  $D = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$  are linearly separable. The condition of zero training error on the training set is expressed as

$$\forall i, \forall \bar{\mathbf{y}}_i \in \mathcal{Y} : \quad \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}_i, \bar{\mathbf{y}}_i) \geq 0. \quad (3.6)$$

Using the maximum-margin principle, we can derive a well-posed optimization problem as

$$\begin{aligned} \min_{\mathbf{w}} \quad & \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{s.t.} \quad & \forall i, \forall \bar{\mathbf{y}}_i \in \mathcal{Y} : \quad \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}_i, \bar{\mathbf{y}}_i) \geq 1. \end{aligned} \quad (3.7)$$

As with the binary and multiclass SVMs, the formulation can be extended to non-separable cases with slack variables  $\{\xi_i \geq 0\}$  as

$$\begin{aligned} \min_{\mathbf{w}, \boldsymbol{\xi} \geq 0} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & \forall i, \forall \bar{\mathbf{y}}_i \in \mathcal{Y} : \quad \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}_i, \bar{\mathbf{y}}_i) \geq 1 - \xi_i. \end{aligned} \quad (3.8)$$

The role of the constraints here can be interpreted similarly as with normal SVMs: for every sample  $i \in \{1, \dots, n\}$ , each constraint requires that the score of its true output  $\mathbf{y}_i \in \mathcal{Y}$ ,  $\mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}_i, \mathbf{y}_i)$ , must be at least one greater than those of all the other outputs  $\bar{\mathbf{y}}_i \in \mathcal{Y} \setminus \{\mathbf{y}_i\}$ . Otherwise, positive penalties are imposed.

The optimization problem formulated in Eq.(3.8) is called an  $n$ -slack formulation simply because  $n$  slack variables are defined. A different formulation will be presented later.

### 3.1.1 Structured Loss

Here we discuss how to incorporate the output structure into the modeling. The normal, non-structural SVM uses a uniform margin loss for all output combinations. By contrast, the structural SVM defines a structured loss  $\Delta : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}_{\geq 0}$  that penalizes the inconsistency between  $\forall \mathbf{y}$  and  $\forall \bar{\mathbf{y}} \in \mathcal{Y}$  individually according to the output pairs. Using this loss, we “rescale” the constraints of the optimization problem given by Eq.(3.8). Two types of rescaling approaches have been proposed.

#### Slack Rescaling

The first approach, called the *slack rescaling* approach, is formulated as

$$\begin{aligned} \min_{\mathbf{w}, \boldsymbol{\xi} \geq 0} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & \forall i, \forall \bar{\mathbf{y}}_i \in \mathcal{Y} : \quad \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}_i, \bar{\mathbf{y}}_i) \geq 1 - \frac{\xi_i}{\Delta(\mathbf{y}_i, \bar{\mathbf{y}}_i)}. \end{aligned} \quad (3.9)$$

#### Margin Rescaling

The second approach, called *margin rescaling*, is formulated as

$$\begin{aligned} \min_{\mathbf{w}, \boldsymbol{\xi} \geq 0} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & \forall i, \forall \bar{\mathbf{y}}_i \in \mathcal{Y} : \quad \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}_i, \bar{\mathbf{y}}_i) \geq \Delta(\mathbf{y}_i, \bar{\mathbf{y}}_i) - \xi_i. \end{aligned} \quad (3.10)$$

#### Discussion

The structured loss creates a sort of hierarchy in the output space for structural learning. Regardless of the type of formulation, the weight is optimized so that the trained classifier can avoid

misclassification with heavy loss. This context is explained in the next section through concrete examples. The difference between the two formulations can be summarized as follows: in margin rescaling, the hinge position is adaptable while the slope is fixed, and vice versa in slack rescaling.

Such formulations with different penalties have been exploited previously for, e.g., imbalance data classification analysis (Xu and Chan, 2003; Sun et al., 2009) and classification including outliers (Yang et al., 2005; Choi et al., 2016). For cases in which false negatives should be penalized more heavily than false positives, such as medical diagnosis, the adaptive penalty-based approach is also effective (Nath and Bhattacharyya, 2007; Wu et al., 2008). The structural SVM can be viewed as a generalization of these formulations.

Hereafter, we use the margin rescaling formulation. The discussions in the following section, however, still apply in slack rescaling with minor modification.

Because the sum of the slack variables is the upper bound on the empirical loss in binary and multiclass SVMs, the same idea is defined here as a proposition.

**Proposition 1.** *In the margin rescaling formulation given by Eq.(3.10), denote the optimal slack variables as  $\{\xi_i^*(\mathbf{w})\}_{i=1}^n$ . Then,  $\frac{1}{n} \sum_{i=1}^n \xi_i^*$  is an upper bound on the empirical risk  $R_{emp}(\mathbf{w})$  given by Eq.(3.2).*

*Proof.* Note that  $\xi_i^* = \max \left[ 0, \max_{\bar{\mathbf{y}}_i \in \mathcal{Y}} \{ \Delta(\mathbf{y}_i, \bar{\mathbf{y}}_i) - (\mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}, \mathbf{y}_i) - \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}, \bar{\mathbf{y}}_i)) \} \right]$ .

Case 1: If  $g(\mathbf{x}_i) = \hat{\mathbf{y}}_i = \mathbf{y}_i$ , then  $\xi_i^* \geq 0 = \Delta(\mathbf{y}_i, g(\mathbf{x}_i))$ .

Case 2: If  $g(\mathbf{x}_i) = \hat{\mathbf{y}}_i \neq \mathbf{y}_i$ , then there exist  $\bar{\mathbf{y}}_i$  such that  $\mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}, \mathbf{y}_i) \leq \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}, \hat{\mathbf{y}}_i)$  and

$$\begin{aligned} \Delta(\mathbf{y}_i, g(\mathbf{x}_i)) = \Delta(\mathbf{y}_i, \hat{\mathbf{y}}_i) &\leq \Delta(\mathbf{y}_i, \hat{\mathbf{y}}_i) - (\mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}, \mathbf{y}_i) - \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}, \hat{\mathbf{y}}_i)) \\ &\leq \max_{\bar{\mathbf{y}}_i \in \mathcal{Y}} \{ \Delta(\mathbf{y}_i, \bar{\mathbf{y}}_i) - (\mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}, \mathbf{y}_i) - \mathbf{w}^T \boldsymbol{\psi}(\mathbf{x}, \bar{\mathbf{y}}_i)) \} = \xi_i^* \end{aligned}$$

Therefore,  $\Delta(\mathbf{y}_i, g(\mathbf{x}_i)) \leq \xi_i^*$ . This also holds for the averaged case, i.e.,

$$R_{emp}(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, g(\mathbf{x}_i)) \leq \frac{1}{n} \sum_{i=1}^n \xi_i^*.$$

□

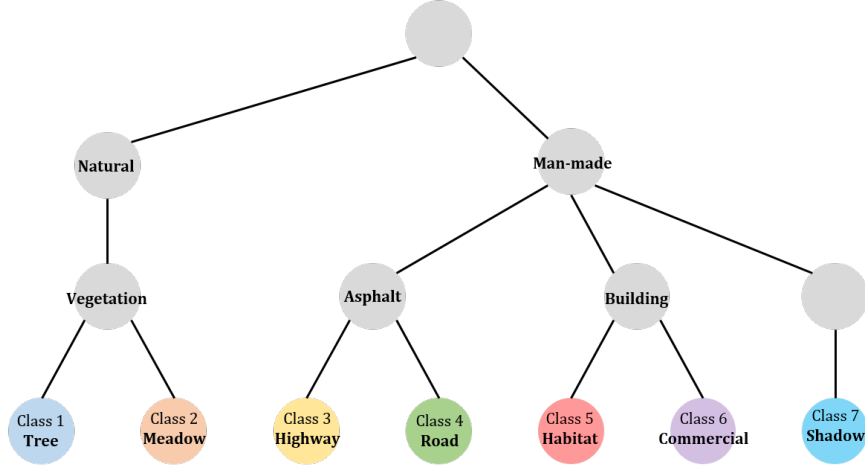


Figure 3.1: Taxonomic tree used in experiments by Tuia et al. (2011).

### 3.1.2 Examples of Structural Learning

Before describing the training strategy for structural SVMs, we show some examples of their application in remotely sensed data analysis.

#### Structured Multiclass Classification in Remote Sensing

The structural SVM was first applied to remote sensing data by Tuia et al. (2011), who performed pixel-wise seven-class classification on a VHR image, obtaining an output vector  $\mathbf{y} \in \{\mathbf{e}_1, \dots, \mathbf{e}_7\}$ . The following joint feature map,

$$\psi(\mathbf{x}, \mathbf{y}) = \mathbf{y} \otimes \phi(\mathbf{x}), \quad (3.11)$$

derives the discriminant function  $g(\mathbf{x}) = \arg \max_{k=1}^7 \{\mathbf{w}_k^T \phi(\mathbf{x})\}$ . The structured loss function  $\Delta$  was defined according to the taxonomic tree shown in Fig.3.1, in which the seven leaf nodes are the target classes. For all label pairs, the structured loss function was defined as the number of edges traversed in reaching the labels' common ancestor in the tree. For the Tree class ( $\mathbf{e}_1$ ) and Meadow class ( $\mathbf{e}_2$ ), for instance,  $\Delta(\mathbf{e}_1, \mathbf{e}_2) = 1$ , because one edge is traversed from each to reach their common ancestor, "Vegetation." For another pair, consisting of the Tree class ( $\mathbf{e}_1$ ) and Highway class ( $\mathbf{e}_3$ ), the structured loss  $\Delta(\mathbf{e}_1, \mathbf{e}_3) = 3$ . In this way, misclassification between classes belonging to different superclasses bears heavy penalties. The experiments by Tuia et al. (2011) actually showed the benefits of structural learning in terms of classification accuracy.

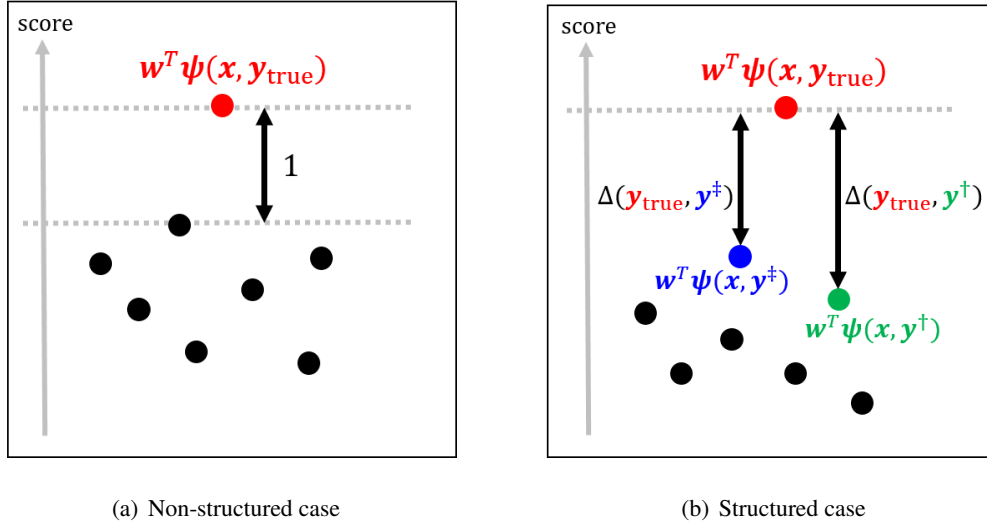


Figure 3.2: Illustrations of structural SVMs for multilabel classification. In the non-structured case (a), the score of the true label is greater than those of the other labels by a uniform margin. In the structured case (b), the true score must be greater by different margins depending on label pairs.

### Structured Multilabel Classification in Remote Sensing

There are still very few reports on coping with multilabel classification in the remote sensing literature. Koda et al. (2018) first tackled structured multilabel classification problems on remote sensing data. In the multilabel case, each element of the output vector  $\mathbf{y} \in \{0, 1\}^c$  expresses the presence or absence of class  $k \in \{1, \dots, c\}$ ; the  $k$ -th element of  $\mathbf{y}$  is 1 if the sample  $\mathbf{x}$  contains the class  $k$  label and 0 otherwise. The joint feature map was defined in the same manner as for the above multiclass classification, i.e.,  $\psi(\mathbf{x}, \mathbf{y}) = \mathbf{y} \otimes \phi(\mathbf{x})$ , which means that each prediction was determined according to the scores of every class and all their combinations.

To clarify the interpretation of structured multilabel classification, consider three-class multilabeling as an example. For every sample  $i \in \{1, \dots, n\}$ , the constraint requires that the score of the true label must be greater than those of the other seven patterns ( $2^3 - 1$ , including the null label), including the margins individually defined on the seven patterns. Figure 3.2 illustrates such a relationship. When no structure is considered in the output space, as shown in Fig.3.2(a), the score of the true label is greater than those of the other labels by a fixed, uniform margin. As Fig.3.2(b) shows, however, that when we take the output structure into consideration, the margins depend on label pairs. In this way, a structural SVM can deal with multilabel classification even in cases tending to have complex structures in the output space.

## 3.2 Training of Structural Support Vector Machine

As the next step, we review the optimization of structural SVMs with the margin rescaling formulation given by Eq.(3.10). We can optimize them in the same way as normal SVMs, that is, by solving the dual problem. In practice, however, this is almost impossible because the number of constraints, which corresponds to the number of dual variables, is  $n|\mathcal{Y}|$ , where the cardinality  $|\mathcal{Y}|$  is typically extremely large or even infinite. Tsochantaridis et al. (2005) proposed a solver when they first proposed the structural SVM, but it still had a polynomial runtime. After that, Thorsten et al. (2009) proposed an efficient optimization algorithm. It involves two important factors, introduced in this section: the 1-slack reformulation and the cutting-plane algorithm.

### 3.2.1 One-Slack Reformulation

The first step of the new algorithm by Thorsten et al. (2009) is reformulation of the primal problem as follows.

$$\begin{aligned} \min_{\mathbf{w}, \xi \geq 0} \quad & \frac{1}{2} \|\mathbf{w}\|^2 + C\xi \\ \text{s.t.} \quad & \forall(\bar{\mathbf{y}}_1, \dots, \bar{\mathbf{y}}_n) \in \mathcal{Y}^n : \quad \frac{1}{n} \mathbf{w}^T \sum_{i=1}^n \left\{ \psi(\mathbf{x}_i, \mathbf{y}_i) - \psi(\mathbf{x}_i, \bar{\mathbf{y}}_i) \right\} \geq \frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, \bar{\mathbf{y}}_i) - \xi. \end{aligned} \quad (3.12)$$

This new formulation is called the *1-slack formulation*. Compared with the original, *n-slack* formulation, this formulation can be viewed as summing the constraints for all samples. As a result, the optimization problem contains  $|\mathcal{Y}|^n$  constraints. Although this number of constraints may be much greater than in the original formulation, the important point is that the slack variable  $\xi$  is shared across all samples. The benefit of this will be apparent when we describe the training process later.

The following theorem declares the equivalence of the two formulations.

**Theorem 1.** *Any solution of the 1-slack formulation,  $\mathbf{w}^*$ , is also a solution of the corresponding *n-slack* formulation, and vice versa, with  $\xi^* = \frac{1}{n} \sum_{i=1}^n \xi_i^*$ .*

*Proof.* For a given  $\mathbf{w}$ , each  $\xi_i$  in the *n-slack* formulation can be optimized individually as

$$\xi_i = \max_{\bar{\mathbf{y}}_i \in \mathcal{Y}} \left[ \Delta(\mathbf{y}_i, \bar{\mathbf{y}}_i) - \mathbf{w}^T \left\{ \psi(\mathbf{x}_i, \mathbf{y}_i) - \psi(\mathbf{x}_i, \bar{\mathbf{y}}_i) \right\} \right]. \quad (3.13)$$

As for the 1-slack formulation, the optimal  $\xi$  for a given  $\mathbf{w}$  is derived as

$$\xi = \max_{(\bar{\mathbf{y}}_1, \dots, \bar{\mathbf{y}}_n) \in \mathcal{Y}^n} \left[ \frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, \bar{\mathbf{y}}_i) - \frac{1}{n} \mathbf{w}^T \sum_{i=1}^n \left\{ \psi(\mathbf{x}, \mathbf{y}_i) - \psi(\mathbf{x}, \bar{\mathbf{y}}_i) \right\} \right]. \quad (3.14)$$

Because the function can be decomposed linearly in  $\bar{\mathbf{y}}_i$ , for any given  $\mathbf{w}$ , we can optimize each element of the summation independently as

$$\xi = \sum_{i=1}^n \max_{\bar{\mathbf{y}}_i \in \mathcal{Y}} \left[ \frac{1}{n} \Delta(\mathbf{y}_i, \bar{\mathbf{y}}_i) - \frac{1}{n} \mathbf{w}^T \left\{ \psi(\mathbf{x}, \mathbf{y}_i) - \psi(\mathbf{x}, \bar{\mathbf{y}}_i) \right\} \right] = \frac{1}{n} \sum_{i=1}^n \xi_i. \quad (3.15)$$

Therefore, for any given  $\mathbf{w}$ , the two formulations are equal. This is also true for their optima,  $\mathbf{w}^*$ ,  $\xi^*$ , and  $\{\xi_i^*\}_{i=1}^n$ .  $\square$

### Dual Problem

The 1-slack formulation gives us a more tractable dual problem than that of the  $n$ -slack formulation, while the equivalence of their solutions is ensured by the above theorem. As with a traditional SVM, we can use the KKT theorem to derive the dual problem.

The Lagrangian of the primal problem is

$$L(\mathbf{w}, \xi, \boldsymbol{\alpha}) = \frac{1}{2} \|\mathbf{w}\|^2 + C\xi - \sum_{\bar{\mathbf{y}} \in \mathcal{Y}^n} \alpha_{\bar{\mathbf{y}}} \left[ \frac{1}{n} \mathbf{w}^T \sum_{i=1}^n \left\{ \psi(\mathbf{x}_i, \mathbf{y}_i) - \psi(\mathbf{x}_i, \bar{\mathbf{y}}_i) \right\} - \frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, \bar{\mathbf{y}}_i) + \xi \right].$$

Differentiating the primal parameters, results in

$$\begin{aligned} \frac{\partial L}{\partial \mathbf{w}} = \mathbf{0} &\Rightarrow \mathbf{w} = \sum_{\bar{\mathbf{y}} \in \mathcal{Y}^n} \alpha_{\bar{\mathbf{y}}} \left[ \frac{1}{n} \sum_{i=1}^n \left\{ \psi(\mathbf{x}_i, \mathbf{y}_i) - \psi(\mathbf{x}_i, \bar{\mathbf{y}}_i) \right\} \right], \\ \frac{\partial L}{\partial \xi} = 0 &\Rightarrow \sum_{\bar{\mathbf{y}} \in \mathcal{Y}^n} \alpha_{\bar{\mathbf{y}}} = C. \end{aligned}$$

Substituting the above into the primal problem then generates the dual problem:

$$\max_{\boldsymbol{\alpha}} \quad \tilde{L}(\boldsymbol{\alpha}) = \sum_{\bar{\mathbf{y}} \in \mathcal{Y}^n} \alpha_{\bar{\mathbf{y}}} \left[ \frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, \bar{\mathbf{y}}_i) \right] - \frac{1}{2} \sum_{\bar{\mathbf{y}} \in \mathcal{Y}^n} \sum_{\bar{\mathbf{y}}' \in \mathcal{Y}^n} \alpha_{\bar{\mathbf{y}}} \alpha_{\bar{\mathbf{y}}'} H(\bar{\mathbf{y}}, \bar{\mathbf{y}}') \quad (3.16)$$

$$\text{s.t. :} \quad \sum_{\bar{\mathbf{y}} \in \mathcal{Y}^n} \alpha_{\bar{\mathbf{y}}} = C \quad \text{and} \quad \alpha_{\bar{\mathbf{y}}} \geq 0 \quad \text{for all } \bar{\mathbf{y}} \in \mathcal{Y}^n,$$

where

$$H(\bar{\mathbf{y}}, \bar{\mathbf{y}}') = \frac{1}{n^2} \left[ \sum_{i=1}^n \left\{ \psi(\mathbf{x}_i, \mathbf{y}_i) - \psi(\mathbf{x}_i, \bar{\mathbf{y}}_i) \right\} \right]^T \left[ \sum_{j=1}^n \left\{ \psi(\mathbf{x}_j, \mathbf{y}_j) - \psi(\mathbf{x}_j, \bar{\mathbf{y}}'_j) \right\} \right].$$

---

**Algorithm 1** Cutting-plane algorithm for the 1-slack formulation

---

```
1: Inputs:  $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n, (C, \varepsilon) \geq 0$ 
2:  $\mathcal{W} \leftarrow \emptyset$ 
3: repeat
4:    $(\mathbf{w}, \xi) \leftarrow \arg \min_{\mathbf{w}, \xi \geq 0} \frac{1}{2} \|\mathbf{w}\|^2 + C\xi$ 
      s.t.  $\forall (\bar{\mathbf{y}}_1, \dots, \bar{\mathbf{y}}_n) \in \mathcal{W} : \frac{1}{n} \mathbf{w}^T \sum_{i=1}^n \left\{ \psi(\mathbf{x}, \mathbf{y}_i) - \psi(\mathbf{x}, \bar{\mathbf{y}}_i) \right\} \geq \frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, \bar{\mathbf{y}}_i) - \xi$ 
5:   for  $i = 1, \dots, n$  do
6:      $\hat{\mathbf{y}}_i \leftarrow \arg \max_{\hat{\mathbf{y}} \in \mathcal{Y}} \left\{ \Delta(\mathbf{y}_i, \hat{\mathbf{y}}) + \mathbf{w}^T \psi(\mathbf{x}_i, \hat{\mathbf{y}}) \right\}$ 
7:   end for
8:    $\mathcal{W} \leftarrow \mathcal{W} \cup \{(\hat{\mathbf{y}}_1, \dots, \hat{\mathbf{y}}_n)\}$ 
9: until  $\frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, \hat{\mathbf{y}}_i) - \frac{1}{n} \mathbf{w}^T \sum_{i=1}^n \left\{ \psi(\mathbf{x}_i, \mathbf{y}_i) - \psi(\mathbf{x}_i, \hat{\mathbf{y}}_i) \right\} \leq \xi + \varepsilon$ 
10: return  $(\mathbf{w}, \xi)$ 
```

---

### 3.2.2 Cutting-Plane Algorithm

Because the Lagrangian  $\alpha_{\bar{\mathbf{y}}}$  in the dual problem given by Eq.(3.16) is associated with each constraint, the total number of dual variables is still of exponential order. To solve such a huge optimization, we consider relaxation programming of the dual problem. In other words, we seek a set of “dominant” constraints, out of the full set of constraints, such that the solution fulfilling the dominant constraints also fulfills almost all the constraints. By then solving the optimization problem subject only to the dominant constraints, we can obtain a near-optimal solution. **Algorithm 1**, called the *cutting-plane algorithm*, provides such a solution. It searches for the most violated constraint in each iteration and repeatedly solves the dual problem, generating a near-optimal solution after a certain step.

We call a set of current constraints under consideration a *working set*, denoted by  $\mathcal{W}$ . The algorithm starts with the null constraint  $\mathcal{W} = \emptyset$ ; that is, the first step does not involve any constraints. In each iteration, we obtain the solution with the current working set (line 4). Then, the most violated constraint is derived (lines 4-6) and added to the working set. We iterate this process until convergence.

The interpretation of the most violated constraint is as follows. In the primal problem, the

solution must satisfy the following constraint:

$$\xi \geq \frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, \hat{\mathbf{y}}_i) - \frac{1}{n} \mathbf{w}^T \sum_{i=1}^n \left\{ \psi(\mathbf{x}_i, \mathbf{y}_i) - \psi(\mathbf{x}_i, \hat{\mathbf{y}}_i) \right\}, \quad (3.17)$$

for all  $\hat{\mathbf{y}} = (\hat{\mathbf{y}}_1, \dots, \hat{\mathbf{y}}_n) \in \mathcal{Y}^n$ . Among these  $\hat{\mathbf{y}}$ , the label set that most seriously violates the constraint is given by the argument of the maximum of the right side:

$$\xi' = \max_{\hat{\mathbf{y}} \in \mathcal{Y}^n} \left[ \frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, \hat{\mathbf{y}}_i) - \frac{1}{n} \mathbf{w}^T \sum_{i=1}^n \left\{ \psi(\mathbf{x}_i, \mathbf{y}_i) - \psi(\mathbf{x}_i, \hat{\mathbf{y}}_i) \right\} \right]. \quad (3.18)$$

As the maximum operation is independent for each  $\hat{\mathbf{y}}_i$ , we obtain

$$\begin{aligned} \xi' &= \frac{1}{n} \sum_{i=1}^n \max_{\hat{\mathbf{y}}_i \in \mathcal{Y}} \left[ \Delta(\mathbf{y}_i, \hat{\mathbf{y}}_i) - \mathbf{w}^T \left\{ \psi(\mathbf{x}_i, \mathbf{y}_i) - \psi(\mathbf{x}_i, \hat{\mathbf{y}}_i) \right\} \right] \\ &= -\frac{1}{n} \sum_{i=1}^n \mathbf{w}^T \psi(\mathbf{x}_i, \mathbf{y}_i) + \frac{1}{n} \sum_{i=1}^n \max_{\hat{\mathbf{y}}_i \in \mathcal{Y}} \left\{ \Delta(\mathbf{y}_i, \hat{\mathbf{y}}_i) + \mathbf{w}^T \psi(\mathbf{x}_i, \hat{\mathbf{y}}_i) \right\}. \end{aligned}$$

Because the first term is constant, the most violated constraint for each sample is derived from the second term, which exactly corresponds to line 6 in **Algorithm 1**. Then, the most violated constraint is added to the working set. For instance, in multilabel classification with  $c$  classes, in which the output is given as a binary descriptor, each iteration step adds a constraint given as a matrix of size  $n \times c$ . Because each Lagrangian  $\alpha_{\hat{\mathbf{y}}}$  is associated with each constraint given as a matrix, the number of dual variables increases by only one in each iteration, whereas it increases by  $n$  in the  $n$ -slack formulation. Moreover, we only must store the sum of the structured loss values and the sum of the joint features. In contrast, the  $n$ -slack formulation requires storing them individually for each sample. This reduces the memory usage for storing constraints. The termination condition then judges the result with the newly added constraint, under the relaxation parameter  $\varepsilon > 0$ . Even though we cannot get a strictly optimal solution with this approach, in practice the parameters obtained by the cutting-plane algorithm can perform better in terms of generalization ability.

The number of iterations for the 1-slack formulation is derived as follows. Let us denote  $R^2 = \max_{i, \bar{\mathbf{y}}} \|\psi(\mathbf{x}_i, \mathbf{y}_i) - \psi(\mathbf{x}_i, \bar{\mathbf{y}})\|^2$  and  $\Delta = \max_{i, \bar{\mathbf{y}}} \Delta(\mathbf{y}_i, \bar{\mathbf{y}})$ . For any  $\varepsilon \in (0, 4R^2C]$ , the algorithm terminates after at most

$$\left\lceil \log_2 \left( \frac{\Delta}{4R^2C} \right) \right\rceil + \left\lceil \frac{16R^2C}{\varepsilon} \right\rceil \quad (3.19)$$

iterations, where  $\lceil \cdot \rceil$  is the integer ceiling function. The  $n$ -slack formulation, by contrast, requires at most

$$\max \left\{ \frac{2n\Delta}{\varepsilon}, \frac{8C\Delta R^2}{\varepsilon^2} \right\} \quad (3.20)$$

steps. The essential difference is that the number of required steps in the 1-slack formulation is independent of the number of samples  $n$ , unlike in the  $n$ -slack formulation.

Note that we can still benefit from the kernel trick even in a structural SVM by defining a kernel function for the inner product of the joint feature vectors, i.e.,  $K : (\mathcal{X} \times \mathcal{Y}) \times (\mathcal{X} \times \mathcal{Y}) \rightarrow \mathbb{R}$ . When we use the joint feature map defined previously for multiclass and multilabel classification,  $\psi(\mathbf{x}, \mathbf{y}) = \mathbf{y} \otimes \phi(\mathbf{x})$ , the kernel function is readily derived as

$$\begin{aligned} K(\mathbf{x}_i, \mathbf{y}_i, \mathbf{x}_j, \mathbf{y}_j) &= \psi(\mathbf{x}_i, \mathbf{y}_i)^T \psi(\mathbf{x}_j, \mathbf{y}_j) \\ &= \mathbf{y}_i^T \mathbf{y}_j \times \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j). \end{aligned}$$

Once we replace the inner product of the input space with a kernel function  $K' : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ , we no longer need to describe the feature map explicitly.

Note that one derivation of the cutting-plane algorithm was presented by Guzman-Rivera et al. (2013), who proposed adding several highly violated constraints in each iteration according to divergence. Their proposed approach can lead to significant computational savings, e.g., a 28% reduction in training time.

## Chapter 4

# Spatial Support Vector Machine

This chapter describes how to embed spatial information directly into the structural SVM formulation.

### 4.1 Spatial Embedding into Structural Support Vector Machine

This section focuses on multiclass or multilabel classification using structural SVMs and modeling of spatial dependency among pixels or tiles. We assume that the joint feature map is given as

$$\psi(\mathbf{x}, \mathbf{y}) = \mathbf{y} \otimes \phi(\mathbf{x}). \quad (4.1)$$

As in the two examples presented in the previous chapter, the output spaces for multiclass and multilabel classification are defined as  $\mathcal{Y} = \{\mathbf{e}_1, \dots, \mathbf{e}_c\}$  and  $\mathcal{Y} = \{0, 1\}^c$ , respectively.

In pixel- or tile-wise image classification, neighboring inputs often tend to share the same labels. To embed this property, we can add an extra term to penalize dissimilarity among neighbors into the structural SVM objective function. First, we focus on a central input  $i$  and its neighbors  $j \in N_i$ , where  $N_i$  is the set of the indexes of the neighborhood. Figure 4.1 shows two examples of neighborhood systems. In a first-order neighborhood system, the four closest inputs are included as the neighbors of  $i$ . In a second-order system, all eight inputs surrounding the center are neighbors. If the neighbors' true labels are known, then the sum of the score distances between the center and its neighbors,

$$\sum_{j \in N_i} \left\{ \mathbf{w}^T \psi(\mathbf{x}_i, \mathbf{y}_i) - \mathbf{w}^T \psi(\mathbf{x}_j, \mathbf{y}_j) \right\}^2, \quad (4.2)$$

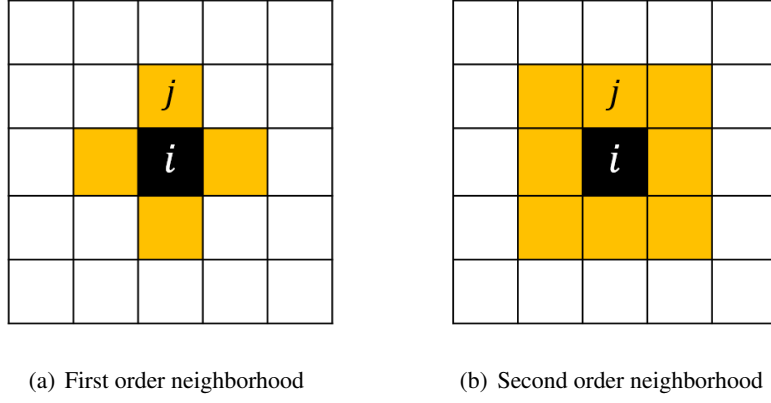


Figure 4.1: Two different definitions of neighborhood systems: a first-order neighborhood (a) and a second-order neighborhood (b).

should be minimized in training to improve smoothness on an image. A similar formulation was proposed for binary SVM classification (Dunder et al., 2006; Flamary and Rakotomamonjy, 2013). Here, however, we move to the structural SVM framework and aim to embed spatial contiguity into it. Note that this must be done on test (unknown) images, which do not have ground truth labels.

As long as we use the joint feature map given by Eq.(4.1), the weight  $\mathbf{w}$  consists of the series of weights associated with all classes, namely  $\mathbf{w} = (\mathbf{w}_1^T, \dots, \mathbf{w}_c^T)^T$ , for both multiclass and multilabel classification. The individual score of class  $k$  for an instance  $\mathbf{x}_i$ ,  $\mathbf{w}_k^T \phi(\mathbf{x}_i)$ , can be rewritten as  $\mathbf{w}^T \psi(\mathbf{x}_i, \mathbf{e}_k)$  by using the joint feature map and unit vectors. Furthermore, for each instance  $\mathbf{x}_i$ , the sum of the score distances among neighbors with respect to all classes has the following formula:

$$\sum_{k=1}^c \sum_{j \in N_i} \left\{ \mathbf{w}^T \psi(\mathbf{x}_i, \mathbf{e}_k) - \mathbf{w}^T \psi(\mathbf{x}_j, \mathbf{e}_k) \right\}^2. \quad (4.3)$$

Minimizing this quantity forces adjacent inputs to have similar scores not only on respective classes but also on every label pattern in the output space  $\mathcal{Y}$ , because they all are combinations of the respective class scores. Let us denote  $\psi_{i,k} = \psi(\mathbf{x}_i, \mathbf{e}_k)$ . Suppose that we have  $N(> n)$  samples, including both training and test data. We also define a symmetric adjacency matrix  $B$  of size  $N \times N$  such that  $B_{ij} = 1$  if inputs  $i$  and  $j$  are neighbors and  $B_{ij} = 0$  otherwise. The total

score distance among adjacent inputs on an entire image can then be written as

$$\begin{aligned}
& \sum_{k=1}^c \sum_{i,j=1}^N B_{ij} \left( \mathbf{w}^T \boldsymbol{\psi}_{i,k} - \mathbf{w}^T \boldsymbol{\psi}_{j,k} \right)^2 \\
&= \mathbf{w}^T \sum_{k=1}^c \left\{ \sum_{i,j=1}^N B_{ij} \left( \boldsymbol{\psi}_{i,k} - \boldsymbol{\psi}_{j,k} \right) \left( \boldsymbol{\psi}_{i,k} - \boldsymbol{\psi}_{j,k} \right)^T \right\} \mathbf{w} \\
&= \mathbf{w}^T \sum_{k=1}^c \left\{ \sum_{i,j=1}^N B_{ij} \boldsymbol{\psi}_{i,k} \boldsymbol{\psi}_{i,k}^T + \sum_{i,j=1}^N B_{ij} \boldsymbol{\psi}_{j,k} \boldsymbol{\psi}_{j,k}^T - 2 \sum_{i,j=1}^N B_{ij} \boldsymbol{\psi}_{i,k} \boldsymbol{\psi}_{j,k}^T \right\} \mathbf{w}.
\end{aligned}$$

Let us define  $d_i = \sum_{j=1}^N B_{ij}$  and the corresponding diagonal matrix  $D = \text{diag}(d_1, \dots, d_N)$ . Then, we rewrite the above equation as

$$\begin{aligned}
& \mathbf{w}^T \sum_{k=1}^c \left\{ 2 \sum_{i=1}^N d_i \boldsymbol{\psi}_{i,k} \boldsymbol{\psi}_{i,k}^T - 2 \sum_{i,j=1}^N B_{ij} \boldsymbol{\psi}_{i,k} \boldsymbol{\psi}_{j,k}^T \right\} \mathbf{w} \\
&= 2 \mathbf{w}^T \left\{ \sum_{k=1}^c \boldsymbol{\Psi}_k^T (D - B) \boldsymbol{\Psi}_k \right\} \mathbf{w},
\end{aligned}$$

where  $\boldsymbol{\Psi}_k = (\boldsymbol{\psi}(x_1, e_k), \dots, \boldsymbol{\psi}(x_N, e_k))^T$  is the joint feature matrix corresponding to class  $k$ .

We define the matrix  $\Sigma$  as

$$\Sigma = \sum_{k=1}^c \boldsymbol{\Psi}_k^T (D - B) \boldsymbol{\Psi}_k, \quad (4.4)$$

where the matrix  $L = D - B$  is a positive semi-definite matrix called a *Laplacian matrix*. Since the distance,  $\mathbf{w}^T \Sigma \mathbf{w}$ , should be minimized, we merge it with the structural SVM primal objective function, giving the following optimization problem.

#### Primal Problem: Spatial and Structural SVM for Land Cover Classification

$$\begin{aligned}
& \min_{\mathbf{w}, \boldsymbol{\xi} \geq 0} \quad \frac{1}{2} \mathbf{w}^T (I + \lambda \Sigma) \mathbf{w} + C \xi \\
& \text{s.t.} \quad \forall (\bar{\mathbf{y}}_1, \dots, \bar{\mathbf{y}}_n) \in \mathcal{Y}^n : \quad \frac{1}{n} \mathbf{w}^T \sum_{i=1}^n \left\{ \boldsymbol{\psi}(x_i, \mathbf{y}_i) - \boldsymbol{\psi}(x_i, \bar{\mathbf{y}}_i) \right\} \geq \sum_{i=1}^n \Delta(\mathbf{y}_i, \bar{\mathbf{y}}_i) - \xi.
\end{aligned} \quad (4.5)$$

The positive constant  $\lambda$  is a parameter to adjust smoothness. Minimization derives the optimal weight parameter to both separate classes for training samples and simultaneously enhance smoothness on an entire image. We have thus proposed an objective function to achieve spatial and structural land cover classification.

The dual problem is derived as follows.

---

**Algorithm 2** Cutting-plane algorithm for the 1-slack formulation with spatial information

---

1: Inputs:  $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n, \Sigma, (C, \lambda, \varepsilon) \geq 0$   
2:  $\mathcal{W} \leftarrow \emptyset$   
3: **repeat**  
4:    $(\mathbf{w}, \xi) \leftarrow \arg \min_{\mathbf{w}, \xi \geq 0} \frac{1}{2} \mathbf{w}^T (I + \lambda \Sigma) \mathbf{w} + C \xi$   
      s.t.  $\forall (\bar{\mathbf{y}}_1, \dots, \bar{\mathbf{y}}_n) \in \mathcal{W} : \frac{1}{n} \mathbf{w}^T \sum_{i=1}^n \left\{ \psi(\mathbf{x}_i, \mathbf{y}_i) - \psi(\mathbf{x}_i, \bar{\mathbf{y}}_i) \right\} \geq \frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, \bar{\mathbf{y}}_i) - \xi$   
5:   **for**  $i = 1, \dots, n$  **do**  
6:      $\hat{\mathbf{y}}_i \leftarrow \arg \max_{\bar{\mathbf{y}} \in \mathcal{Y}} \left\{ \Delta(\mathbf{y}_i, \bar{\mathbf{y}}) + \mathbf{w}^T \psi(\mathbf{x}_i, \bar{\mathbf{y}}) \right\}$   
7:   **end for**  
8:    $\mathcal{W} \leftarrow \mathcal{W} \cup \{(\hat{\mathbf{y}}_1, \dots, \hat{\mathbf{y}}_n)\}$   
9: **until**  $\frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, \hat{\mathbf{y}}_i) - \frac{1}{n} \mathbf{w}^T \sum_{i=1}^n \left\{ \psi(\mathbf{x}_i, \mathbf{y}_i) - \psi(\mathbf{x}_i, \hat{\mathbf{y}}_i) \right\} \leq \xi + \varepsilon$   
10: **return**  $(\mathbf{w}, \xi)$

---

**Dual Problem: Spatial and Structural SVM for Land Cover Classification**

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \tilde{L}(\boldsymbol{\alpha}) = \sum_{\bar{\mathbf{y}} \in \mathcal{Y}^n} \alpha_{\bar{\mathbf{y}}} \left[ \frac{1}{n} \sum_{i=1}^n \Delta(\mathbf{y}_i, \bar{\mathbf{y}}_i) \right] - \frac{1}{2} \sum_{\bar{\mathbf{y}} \in \mathcal{Y}^n} \sum_{\bar{\mathbf{y}}' \in \mathcal{Y}^n} \alpha_{\bar{\mathbf{y}}} \alpha_{\bar{\mathbf{y}}'} H(\mathbf{y}, \bar{\mathbf{y}}'), \quad (4.6) \\ \text{s.t. :} \quad & \sum_{\bar{\mathbf{y}} \in \mathcal{Y}^n} \alpha_{\bar{\mathbf{y}}} = C \quad \text{and} \quad \alpha_{\bar{\mathbf{y}}} \geq 0 \quad \text{for all } \bar{\mathbf{y}} \in \mathcal{Y}^n. \end{aligned}$$

where

$$\begin{aligned} H(\mathbf{y}, \bar{\mathbf{y}}') &= \frac{1}{n^2} \left[ \sum_{i=1}^n \left\{ \psi(\mathbf{x}_i, \mathbf{y}_i) - \psi(\mathbf{x}_i, \bar{\mathbf{y}}_i) \right\} \right]^T \times \\ &\quad (I + \lambda \Sigma)^{-1} \left[ \sum_{j=1}^n \left\{ \psi(\mathbf{x}_j, \mathbf{y}_j) - \psi(\mathbf{x}_j, \bar{\mathbf{y}}'_j) \right\} \right]. \end{aligned}$$

Because the matrix  $\lambda \Sigma$  is positive semi-definite, the dual problem can still be formulated as a well-posed quadratic programming problem. After solving it, we can derive the primal solution with Lagrange multipliers as

$$\mathbf{w} = (I + \lambda \Sigma)^{-1} \sum_{\bar{\mathbf{y}} \in \mathcal{Y}^n} \alpha_{\bar{\mathbf{y}}} \left[ \frac{1}{n} \sum_{i=1}^n \left\{ \psi(\mathbf{x}_i, \mathbf{y}_i) - \psi(\mathbf{x}_i, \bar{\mathbf{y}}_i) \right\} \right]. \quad (4.7)$$

**Algorithm 2** is a complete algorithm to optimize the weight parameter in the same way as the original cutting-plane algorithm. The only difference from the original algorithm is in step 4, which now includes the term  $\lambda \Sigma$  for spatial contiguity.

## Kernelization

Here, we show that kernelization of the dual problem is possible even in the spatial formulation. Note that the size of matrix  $\Sigma$  is  $(p \times c) \times (p \times c)$ , where  $c$  is the number of target classes and  $p$  is the dimension of the feature space based on the mapping  $\phi : \mathbb{R}^d \rightarrow \mathbb{R}^p$ . The matrix  $\Sigma$  is a block diagonal matrix with diagonal blocks  $\Sigma_k$  of size  $p \times p$  and given by

$$\Sigma_k = \Phi^T L \Phi, \quad (4.8)$$

where  $L = D - B$ , and  $\Phi = (\phi(x_1), \dots, \phi(x_N))^T$  is the feature matrix. Note that the diagonal blocks do not actually depend on  $k$ . Therefore, to obtain the inverse matrix in the primal solution,  $(I + \lambda \Sigma)^{-1}$ , requires only finding the inverse of the matrix  $(I_p + \lambda \Phi^T L \Phi)$ , where  $I_p$  denotes the  $p$ -dimensional identity matrix. Note that the Laplacian matrix  $L$  is not invertible: with a vector of ones,  $\mathbf{1}$ , we can derive the equation  $L\mathbf{1} = \mathbf{0}$ , meaning that at least one eigenvalue is zero.

In general, the adjacency matrix  $B$  can be viewed as a matrix representation of an undirected graph  $\mathcal{G} = (V, E)$ , where  $V$  and  $E$  are the sets of nodes and edges respectively. That is,  $B_{ij} = 1$  indicates that nodes  $i$  and  $j$  are connected on the corresponding graph  $\mathcal{G}$ . The rank of the graph's Laplacian matrix is given by  $N - r$ , where  $r$  is the number of “connected” components of  $G$  (see, e.g., Bapat (1996)). For instance, if a graph is fully connected, meaning that all pairs of nodes have a connected path between them on the graph, then the rank of the Laplacian matrix  $L$  is  $N - 1$ . In this situation,  $L$  has a series of nonnegative eigenvalues,  $0 = \lambda_1 < \lambda_2 \leq \dots \leq \lambda_N$ . Using the positive eigenvalues, we can decompose  $L$  as

$$L = V \Lambda V^T, \quad (4.9)$$

where  $\Lambda = \text{diag}(\lambda_2, \dots, \lambda_N)$ , and the columns of  $V$  are the eigenvectors of  $L$  whose corresponding eigenvalues are positive. Then, the required inverse matrix is computed with the *Woodbury matrix identity* as

$$\begin{aligned} (I_p + \lambda \Phi^T L \Phi)^{-1} &= (I_p + \lambda \Phi^T V \Lambda V^T \Phi)^{-1} \\ &= I_p - \lambda \Phi^T V (\Lambda^{-1} + \lambda V^T \Phi \Phi^T V)^{-1} V^T \Phi, \end{aligned}$$

which gives

$$(I + \lambda \Sigma)^{-1} = I - \lambda \sum_{k=1}^c \Psi_k^T V (\Lambda^{-1} + \lambda V^T \Phi \Phi^T V)^{-1} V^T \Psi_k. \quad (4.10)$$

Once we define kernels on the feature and joint feature spaces as

$$\begin{aligned} K'(\mathbf{x}_i, \mathbf{x}_j) &= \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j), \\ K(\mathbf{x}_i, \mathbf{y}_i, \mathbf{x}_j, \mathbf{y}_j) &= \psi(\mathbf{x}_i, \mathbf{y}_i)^T \psi(\mathbf{x}_j, \mathbf{y}_j) = \mathbf{y}_i^T \mathbf{y}_j \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j), \end{aligned}$$

respectively, we no longer need to explicitly represent the feature vectors, meaning that we can still make use of the kernel trick even in the spatial formulation. The computational cost, however, drastically increases.

Note that the discussion so far can be extended to weighted adjacency matrices, on whose corresponding graphs edges are weighted according to their importance, strength, or reliability.

## Related Works

Spatial SVM modeling was originally developed by Dunder et al. (2006) and Flamary and Rakotomamonjy (2013) for traditional binary SVMs. We extended their works to enable multiclass and multilabel classification even with output structure learning. This extension to multiclass classification can be viewed as the generalization of the Crammer-Singer multiclass SVM (Crammer and Singer, 2001).

Previously developed spatial classification techniques for remotely sensed data have mostly required multiple steps to generate their final outputs. For instance, Markov random field- (MRF-) based approaches have two major steps: pixel-wise classification and smoothing classification (Jackson and Landgrebe, 2002; Moser and Serpico, 2013). Approaches have been also developed by applying the two processes of spatial feature extraction and classification with normal classifiers (Shu et al., 2018). CNNs can integrate those two processes (Makantanis et al., 2015; Chen et al., 2016; Li et al., 2017). Compared with such approaches, the advantage of the proposal described here is that we can implement classifiers that simultaneously incorporate output structure and spatial contiguity in just one step, for both multiclass and multilabel classification. Moreover, this approach can easily be combined with spatial feature extraction approaches, which provide better classification.

## 4.2 Numerical Experiments

This section describes numerical experiments conducted with the spatial SVM approach. First, we assume that the structured loss is given by the step function:  $\Delta(\mathbf{y}, \bar{\mathbf{y}}) = 1$  if  $\mathbf{y} \neq \bar{\mathbf{y}}$  and 0 otherwise. That is, for the sake of evaluating the effectiveness of spatial modeling, we do not consider any output structure. Instead, the following chapter presents numerical assessments of multilabel classification with spatial and structured learning.

### 4.2.1 Datasets

#### Artificial Data

The first dataset was artificial data used in Nishii and Eguchi (2004), with the ground truth image shown in Fig.4.2(a). The dataset consists of three categories ( $c = 3$ ) corresponding to the colors in the image: purple, green, and yellow. The number of pixels is  $91 \times 91$ . The numbers of pixels for each of the three categories are 3330, 1371, and 3580, respectively. A four-dimensional spectral image was simulated for the experiments, making the image size  $91 \times 91 \times 4$  (height  $\times$  width  $\times$  depth). As an input  $\mathbf{x}_i$ , a  $5 \times 5$  patch was extracted with its center on pixel  $i$ , giving a feature dimension of  $d = 5 \times 5 \times 4 = 100$ . The spectral features were generated independently from multivariate Gaussian distributions with mean vectors  $\boldsymbol{\mu}_1 = (1, 0, 0, 0)^T$ ,  $\boldsymbol{\mu}_2 = (-1, 1, 0, 0)^T$ , and  $\boldsymbol{\mu}_3 = (-1, -1, 0, 0)^T$  for each category. A common covariance matrix  $\sigma^2 I$  was used for all the generators.

#### Real-World Data

In addition, the spatial SVM performance was evaluated on two real-world hyperspectral image datasets. Both can be downloaded from the web<sup>1</sup>.

1. **Pavia University:** The first hyperspectral image dataset, Pavia University, was acquired in 2002 by the ROSIS sensor during a flight campaign over the University of Pavia in northern Italy. The geometric resolution is 1.3 meters. The numbers of target classes and spectral bands are 9 and 103, respectively. For each pixel a  $3 \times 3$  patch was extracted as its features.

---

<sup>1</sup>[http://www.ehu.eus/ccwintco/index.php/Hyperspectral\\_Remote\\_Sensing\\_Scenes](http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes)

The number of pixels is  $610 \times 340 (= 207400)$ , among which 42776 pixels hold ground truth labels.

2. **Indian Pines:** The second hyperspectral image dataset, Indian Pines, was acquired in 1992 by the AVIRIS sensor over a test site in northwestern Indiana. The geometric resolution is 20 meters. The numbers of target classes and spectral bands are 16 and 200, respectively, and bands covering regions of water absorption were removed. As with the Pavia University dataset,  $3 \times 3$  patches were extracted. The number of pixels is  $145 \times 145 (= 21025)$ , among which 10249 pixels hold ground truth labels.

#### 4.2.2 Experimental Setup

The performance criteria were overall accuracy (OA) and average accuracy (AA). The OA is the ratio of correctly predicted samples. The AA is the average accuracy over all classes, which can reflect classification difficulties for small classes. Samples were divided into training, validation, and test sets by random sampling. The parameters  $C$  and  $\lambda$  were selected by line search, with  $C$  selected first according to the AAs of the validation sets and then  $\lambda$  selected as well. The parameters were line-searched on  $C, \lambda \in \{2^{-15}, 2^{-14}, \dots, 2^0\}$  for the artificial data, and on  $C \in \{2^{-5}, 2^{-4}, \dots, 2^{12}\}$  and  $\lambda \in \{2^{-12}, 2^{-11}, \dots, 2^5\}$  for the real-world data. A total of 10 runs were performed, and the final results were the averages of the OAs and AAs on the test sets. The proposed spatial SVM was compared with a normal multiclass SVM based on the CS-SVM, without considering structure here.

### 4.2.3 Results

#### Artificial Data

Table 4.1: Classification accuracy on artificial data

| $\sigma^2$ | Criterion | Normal SVM            | Spatial SVM                           | Gain    |
|------------|-----------|-----------------------|---------------------------------------|---------|
| 1          | OA        | 0.964 ( $\pm 0.003$ ) | <b>0.966 (<math>\pm 0.003</math>)</b> | + 0.2 % |
|            | AA        | 0.967 ( $\pm 0.003$ ) | <b>0.970 (<math>\pm 0.003</math>)</b> | + 0.3 % |
| 2          | OA        | 0.906 ( $\pm 0.007$ ) | <b>0.915 (<math>\pm 0.010</math>)</b> | + 0.9 % |
|            | AA        | 0.908 ( $\pm 0.011$ ) | <b>0.920 (<math>\pm 0.010</math>)</b> | + 1.2 % |
| 3          | OA        | 0.794 ( $\pm 0.013$ ) | <b>0.829 (<math>\pm 0.012</math>)</b> | + 3.5 % |
|            | AA        | 0.795 ( $\pm 0.014$ ) | <b>0.834 (<math>\pm 0.016</math>)</b> | + 3.9 % |
| 4          | OA        | 0.687 ( $\pm 0.019$ ) | <b>0.730 (<math>\pm 0.013</math>)</b> | + 4.3 % |
|            | AA        | 0.688 ( $\pm 0.021$ ) | <b>0.734 (<math>\pm 0.012</math>)</b> | + 4.6 % |

Experiments were performed first on the artificial data. For the training and validation sets, 200 and 100 samples, respectively, were randomly sampled from each of the three categories; the rest of the data was used for the test sets. Table 4.1 lists the performance across 10 runs. When  $\sigma^2 = 1$ , meaning that the input features themselves were still discriminative, the two classifiers had comparable performance: their differences were less than 0.3% on both criteria. As the variance increased, however, the more the spatial SVM’s performance improved over that of the normal SVM. Specific, the performance gains were 4.3% (OA) and 4.6% (AA) when  $\sigma^2 = 4$ . This indicates that spatial information makes up for the lack of discriminative information in features.

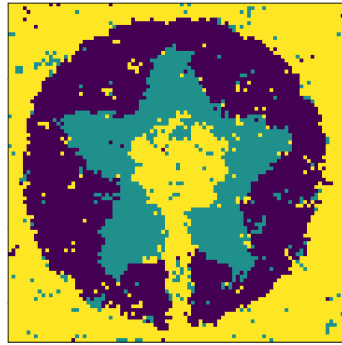
Figure 4.2 illustrates the qualitative performance of each approach. Compared with the classification maps created by the normal SVM (middle row), those created by the spatial SVM were clearer and smoother (bottom row). When  $\sigma^2 = 1$ , the map from the normal SVM looked reasonable (b), but misclassified pixels were scattered across the image. In contrast, as shown in Fig.4.2(e), the spatial SVM “flipped” such labels to correct ones. Misclassified pixels were distributed only around class boundaries, where it is harder, in general, to provide discriminative features. Enhancing smoothness sometimes wrongly affected the map, as shown in the pink circles in Figs.4.2(d) and (g): the spatial SVM filled the holes in islands composed of misclassified



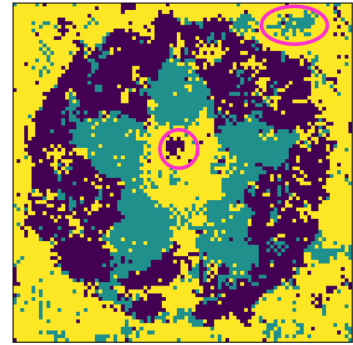
(a) Ground truth



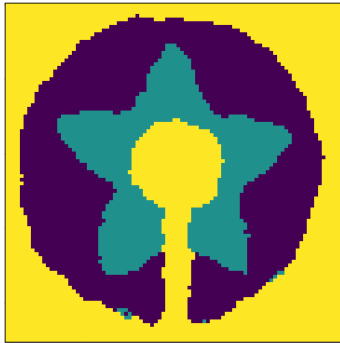
(b) Normal SVM,  $\sigma^2 = 1$



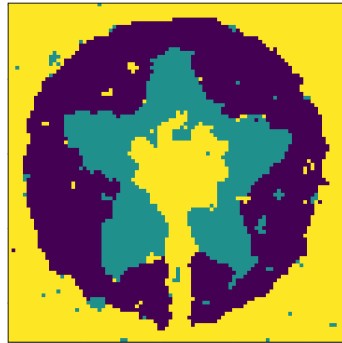
(c) Normal SVM,  $\sigma^2 = 2$



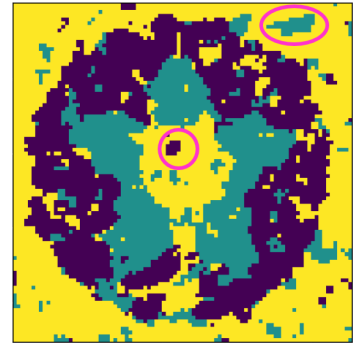
(d) Normal SVM,  $\sigma^2 = 3$



(e) Spatial SVM,  $\sigma^2 = 1$



(f) Spatial SVM,  $\sigma^2 = 2$



(g) Spatial SVM,  $\sigma^2 = 3$

Figure 4.2: Classification maps generated by normal and spatial SVMs on artificial data (Nishii and Eguchi, 2004) for different variances ( $\sigma^2 = 1, 2, 3$ ). The middle row (b-d) shows maps from normal SVMs, while the bottom row (e-g) shows maps from spatial SVMs. The pink-circled areas in (d) and (g) indicate locations where spatial contiguity wrongly affected the map.

pixels with incorrect labels. This implies that discriminative features, to some extent, are necessary to receive the full benefits of spatial embedding, indicating the necessity of developing better pre-processing feature extraction/reduction/transformation approaches. Overall, however, the proposed spatial SVM classifiers outperformed the conventional SVMs.

## Pavia University

Table 4.2: Classification accuracy on Pavia University data

| #  | Class        | Training | Validation | Test  | Normal SVM                            | Spatial SVM                           |
|----|--------------|----------|------------|-------|---------------------------------------|---------------------------------------|
| 1  | Asphalt      | 300      | 150        | 6181  | 0.919 ( $\pm$ 0.010)                  | <b>0.928 (<math>\pm</math> 0.007)</b> |
| 2  | Meadow       | 300      | 150        | 18199 | 0.862 ( $\pm$ 0.016)                  | <b>0.871 (<math>\pm</math> 0.015)</b> |
| 3  | Gravel       | 100      | 50         | 1949  | <b>0.879 (<math>\pm</math> 0.022)</b> | 0.867 ( $\pm$ 0.025)                  |
| 4  | Trees        | 100      | 50         | 2914  | 0.903 ( $\pm$ 0.023)                  | <b>0.917 (<math>\pm</math> 0.020)</b> |
| 5  | Metal sheets | 100      | 50         | 1195  | <b>0.999 (<math>\pm</math> 0.001)</b> | <b>0.999 (<math>\pm</math> 0.002)</b> |
| 6  | Bare soil    | 300      | 150        | 4579  | 0.827 ( $\pm$ 0.028)                  | <b>0.835 (<math>\pm</math> 0.024)</b> |
| 7  | Bitumen      | 100      | 50         | 1180  | 0.916 ( $\pm$ 0.014)                  | <b>0.928 (<math>\pm</math> 0.018)</b> |
| 8  | Bricks       | 100      | 50         | 3532  | 0.827 ( $\pm$ 0.031)                  | <b>0.828 (<math>\pm</math> 0.034)</b> |
| 9  | Shadows      | 100      | 50         | 797   | 0.991 ( $\pm$ 0.002)                  | <b>0.994 (<math>\pm</math> 0.004)</b> |
| OA |              | 1500     | 750        | 40526 | 0.876 ( $\pm$ 0.007)                  | <b>0.882 (<math>\pm</math> 0.005)</b> |
| AA |              |          |            |       | 0.903 ( $\pm$ 0.007)                  | <b>0.907 (<math>\pm</math> 0.005)</b> |

Table 4.2 lists the performance across 10 runs on the Pavia University data. Although the SVMs' performances were very competitive, the spatial SVM performed better. Spatial modeling achieved higher accuracy by 0.6% and 0.4% in terms of OA and AA, respectively, over the normal SVM. Only for the "Gravel" class did the normal SVM achieve higher accuracy than the spatial SVM did, but the use of spatial information generally improved the classification accuracy regardless of the class size.

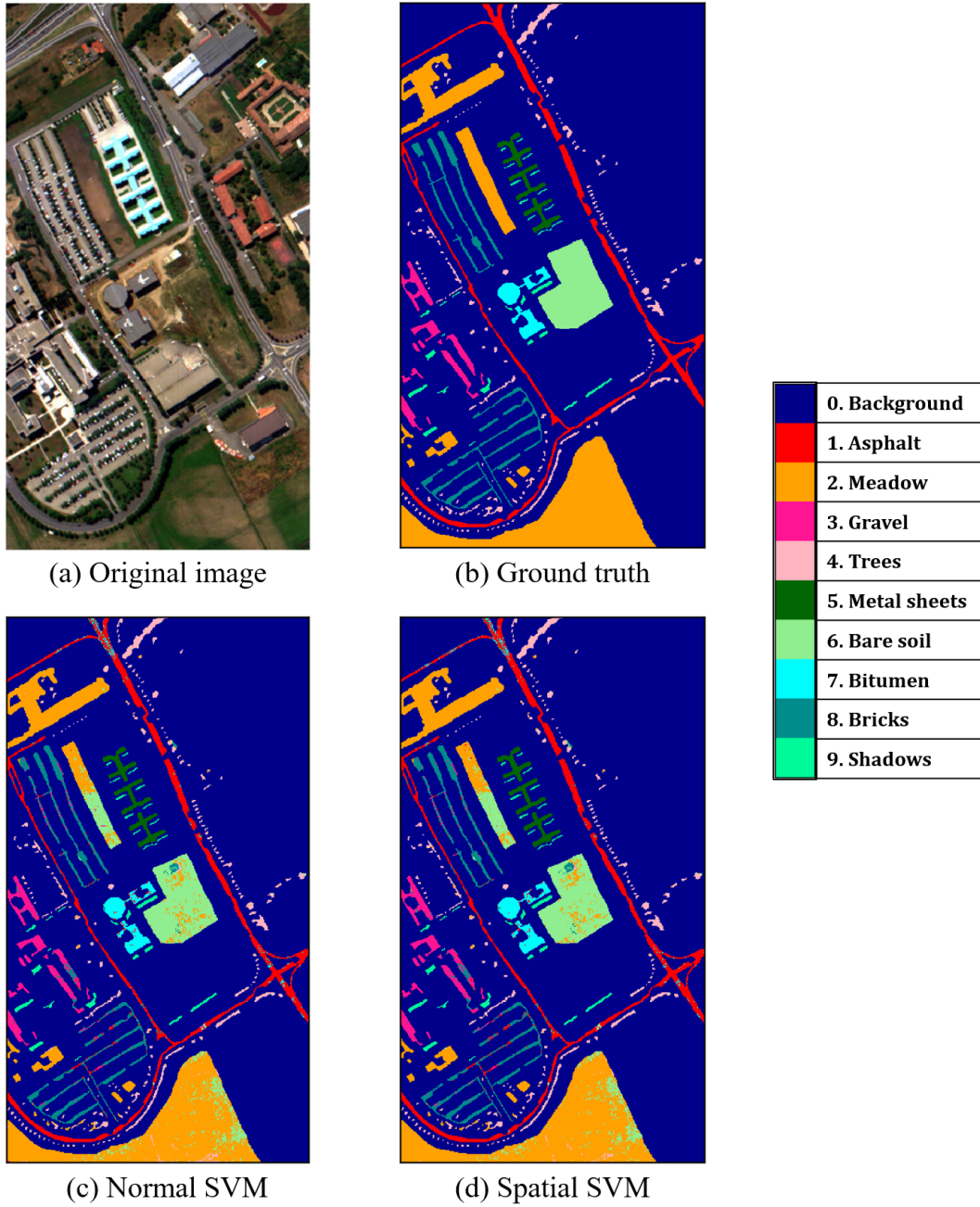


Figure 4.3: Original image (a) and classification maps for the Pavia University data: the ground truth (b), and predicted maps generated by the normal SVM (c) and spatial SVM (d).

Figure 4.3 shows the resulting classification maps. Qualitatively, the performance also looked competitive between the SVMs. The dataset has relatively discriminative features already, as the accuracy results implied, but combining spectral and spatial information could still make classification more accurate. Even on VHR images expressed with a few meters' resolution, spatial

information can still improve accuracy with correct tuning of the smoothness parameter.

## Indian Pines

Table 4.3: Classification accuracy on Indian Pines data

| #  | Class                       | Training | Validation | Test | Normal SVM                          | Spatial SVM                         |
|----|-----------------------------|----------|------------|------|-------------------------------------|-------------------------------------|
| 1  | Alfalfa                     | 10       | 5          | 31   | <b>0.674</b> ( $\pm$ <b>0.144</b> ) | 0.667 ( $\pm$ 0.155)                |
| 2  | Corn-notill                 | 30       | 15         | 1383 | 0.431 ( $\pm$ 0.048)                | <b>0.559</b> ( $\pm$ <b>0.098</b> ) |
| 3  | Corn-nimtill                | 30       | 15         | 785  | 0.501 ( $\pm$ 0.053)                | <b>0.524</b> ( $\pm$ <b>0.049</b> ) |
| 4  | Corn                        | 30       | 15         | 192  | 0.736 ( $\pm$ 0.062)                | <b>0.830</b> ( $\pm$ <b>0.073</b> ) |
| 5  | Grass-pasture               | 30       | 15         | 438  | 0.900 ( $\pm$ 0.021)                | <b>0.915</b> ( $\pm$ <b>0.025</b> ) |
| 6  | Grass-trees                 | 30       | 15         | 685  | 0.946 ( $\pm$ 0.010)                | <b>0.976</b> ( $\pm$ <b>0.011</b> ) |
| 7  | Grass-pasture-mowed         | 10       | 5          | 13   | 0.983 ( $\pm$ 0.032)                | <b>0.991</b> ( $\pm$ <b>0.024</b> ) |
| 8  | Hay-windrowed               | 30       | 15         | 433  | 0.977 ( $\pm$ 0.015)                | <b>0.992</b> ( $\pm$ <b>0.006</b> ) |
| 9  | Oats                        | 10       | 5          | 5    | <b>1.0</b> ( $\pm$ <b>0.0</b> )     | <b>1.0</b> ( $\pm$ <b>0.0</b> )     |
| 10 | Soybean-notill              | 30       | 15         | 927  | 0.593 ( $\pm$ 0.048)                | <b>0.677</b> ( $\pm$ <b>0.075</b> ) |
| 11 | Soybean-mintill             | 30       | 15         | 2410 | 0.520 ( $\pm$ 0.047)                | <b>0.546</b> ( $\pm$ <b>0.066</b> ) |
| 12 | Soybean-clean               | 30       | 15         | 548  | 0.513 ( $\pm$ 0.048)                | <b>0.707</b> ( $\pm$ <b>0.142</b> ) |
| 13 | Wheat                       | 30       | 15         | 160  | 0.997 ( $\pm$ 0.004)                | <b>1.0</b> ( $\pm$ <b>0.0</b> )     |
| 14 | Woods                       | 30       | 15         | 1220 | 0.858 ( $\pm$ 0.027)                | <b>0.907</b> ( $\pm$ <b>0.034</b> ) |
| 15 | Building-grass-trees-drives | 30       | 15         | 341  | 0.617 ( $\pm$ 0.023)                | <b>0.762</b> ( $\pm$ <b>0.091</b> ) |
| 16 | Stone-steel-towers          | 30       | 15         | 48   | 0.975 ( $\pm$ 0.024)                | <b>0.991</b> ( $\pm$ <b>0.014</b> ) |
| OA |                             | 420      | 210        | 9619 | 0.641 ( $\pm$ 0.014)                | <b>0.764</b> ( $\pm$ <b>0.046</b> ) |
| AA |                             |          |            |      | 0.764 ( $\pm$ 0.010)                | <b>0.822</b> ( $\pm$ <b>0.039</b> ) |

Table 4.3 lists the performance across 10 runs on the Indian Pines data. Unlike the results on the Pavia University data, the spatial SVM significantly improved the classification performance. Spatial modeling achieved higher accuracy by 12.3% and 5.8% in terms of OA and AA, respectively, over the normal SVM. As with classifying the high-variance artificial data, we could obtain great gains through spatial embedding when dealing with inaccurate classification.

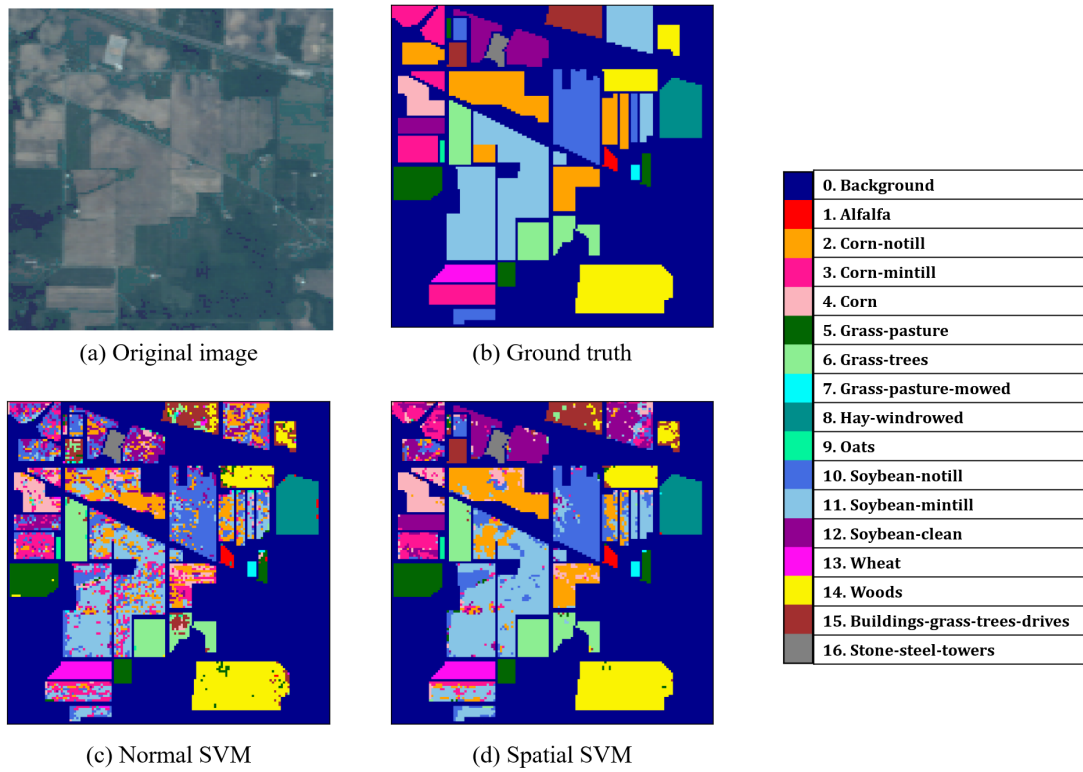


Figure 4.4: Original image (a) and classification maps for the Indian Pines data: the ground truth (b), and predicted maps generated by the normal SVM (c) and spatial SVM (d).

Figure 4.4 shows the resulting classification maps. From the qualitative viewpoint, the spatial SVM apparently improved the classification performance. For instance, consider the classification of class 14 (“Woods”), represented by the yellow islands on the right sides of the figures. For this class, the spatial SVM flipped incorrectly classified labels and created smoother predictions than those of the normal SVM.

We next examined the sensitivity of the spatial SVM performance on the smoothing parameter  $\lambda$ . Figure 4.5(a) shows the AAs for all pairs of  $C$  and  $\lambda$  on one run. The results confirmed that the AA continuously varied with the parameters. When  $\lambda$  was nearly equal to zero (i.e., the normal SVM model), the best AA was around 0.78. As  $\lambda$  increased, the AAs for the spatial SVM exceeded the baseline over a wide region of the parameter space, shown by the pink-red gradation in the figure. Especially when  $C$  was large, the smoothing parameter acted as a spatial regularizer. As a result, the classification accuracy was greatly enhanced. Figure 4.5(b) shows the AAs when  $C = 64$ , which was the parameter value selected for the normal SVM according to the validation

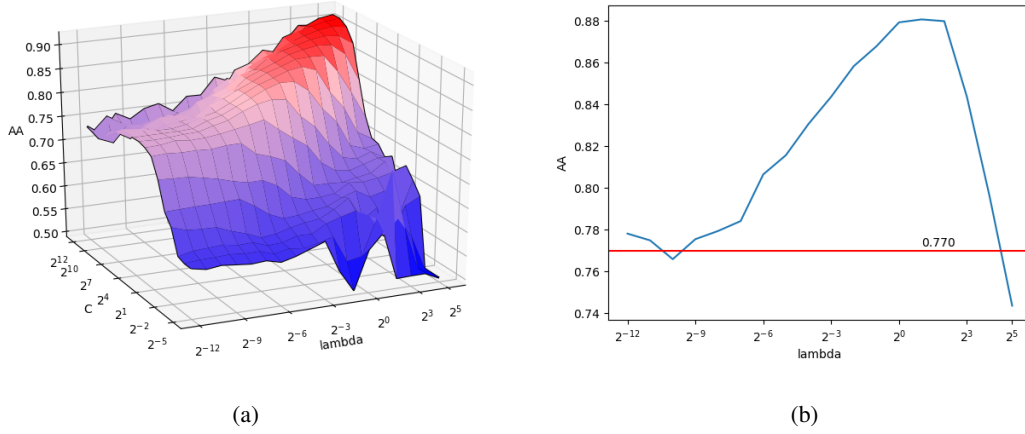


Figure 4.5: Parameter sensitivity for the average accuracy (AA) on the grid space defined by the parameters  $(C, \lambda)$  (a), and for different  $\lambda$  values when  $C$  was fixed (b). The red line represents the best AA generated by the normal SVM.

error. The red line shows the corresponding AA on the test set. Here, the results also confirmed that the spatial SVM could improve the accuracy with appropriately tuned smoothing parameters.

### 4.3 Summary

This chapter has described a novel spatial SVM formulation to generate smooth predictions, as an extension of the structural SVM, which can be applied to multiclass and multilabel classification of data containing complex output structures. To evaluate how much the spatial modeling could improve classification performance, numerical experiments were conducted on both artificial data and two real-world sets of hyperspectral image data. The results clearly demonstrated that the spatial SVM can efficiently improve classification performance.

## Chapter 5

# Numerical Experiments on UAV

## Imagery

This chapter describes numerical assessments of the proposed spatial and structural SVMs for multilabel land cover classification on EHR images acquired by UAVs and an airplane.

### 5.1 Datasets

#### **Dataset 1: Civezzano Dataset**

The first set of images is a UAV dataset acquired near the city of Civezzano (Trento, Italy) at different off-nadir angles on October 17, 2012 (Zeggada et al., 2017). The data acquisition was performed with a Canon EOS 550D still camera with an 18-megapixel CMOS APS-C sensor. The resulting images had three channels (RGB) and a spatial resolution of approximately 2 cm. Their size was  $5184 \times 3456$  pixels with 8-bit radiometric resolution. The dataset consists of ten images. All the images were subdivided into a non-overlapping grid of tiles with  $50 \times 50$  pixels each, as explained in Chapter 2. Three learning images were selected such that they each contained all 14 predefined object classes, as listed in Table 5.1. The learning set was then divided randomly into training and validation sets. The final sizes of the training, validation, and test sets were 5000, 16321, and 49749 tiles, respectively. Table 5.1 lists all the class occurrences in the data. Because of its large heterogeneity, the Roof category was divided into three different classes.

Table 5.1: Summary of class occurrences in dataset 1 (Civezzano)

| Class |                 | Number of tiles |            |       |
|-------|-----------------|-----------------|------------|-------|
| #     | Name            | Training        | Validation | Test  |
|       |                 | 5000            | 16324      | 49749 |
| 1     | Asphalt         | 1195            | 3957       | 13280 |
| 2     | Grass           | 2082            | 6906       | 18508 |
| 3     | Tree            | 886             | 2843       | 8790  |
| 4     | Vineyard        | 58              | 182        | 1228  |
| 5     | Low vegetation  | 374             | 1110       | 1354  |
| 6     | Car             | 215             | 610        | 1458  |
| 7     | Roof1           | 239             | 772        | 1728  |
| 8     | Roof2           | 218             | 657        | 1254  |
| 9     | Roof3           | 121             | 341        | 2833  |
| 10    | Solar panel     | 98              | 295        | 456   |
| 11    | Building facade | 217             | 706        | 1224  |
| 12    | Soil            | 205             | 755        | 1957  |
| 13    | Gravel          | 17              | 58         | 491   |
| 14    | Rocks           | 156             | 538        | 731   |

## Dataset 2: Munich Dataset

The second dataset was originally acquired for the purpose of vehicle detection over Munich, Germany (Liu and Mattyus, 2015), but we use it here for land use classification. The images were captured from an airplane by a Canon Eos 1Ds Mark III camera with a resolution of  $5616 \times 3744$  pixels and a 50-mm focal length. The images were taken at a height of 1000 meters above ground, with a spatial resolution of approximately 13 cm. The dataset consists of 20 images, of which 10 were used as test images. The rest of the images were divided into training and validation sets containing 5000 and 77880 tiles, respectively. Table 5.2 lists the numbers of class occurrences in every set.

Table 5.2: Summary of class occurrences in dataset 2 (Munich)

| Class |                                     | Tiles    |            |       |
|-------|-------------------------------------|----------|------------|-------|
| #     | Name                                | Training | Validation | Test  |
|       |                                     | 5000     | 77880      | 82880 |
| 1     | Slate roof                          | 296      | 4394       | 9836  |
| 2     | Clay roof                           | 285      | 4046       | 6069  |
| 3     | Asphalt                             | 1344     | 21839      | 31366 |
| 4     | Standing seam roof                  | 224      | 3378       | 2786  |
| 5     | Sports field                        | 13       | 234        | 383   |
| 6     | Green trees                         | 1487     | 24105      | 32983 |
| 7     | Bare trees                          | 685      | 11112      | 7466  |
| 8     | Grass                               | 1797     | 27697      | 25737 |
| 9     | Building site / operating machinery | 19       | 408        | 275   |
| 10    | Paved road                          | 162      | 2367       | 968   |
| 11    | Soil                                | 336      | 5107       | 3649  |
| 12    | Dark soil                           | 243      | 3571       | 1563  |
| 13    | Green roof                          | 130      | 2002       | 2594  |
| 14    | Flat concrete roof                  | 231      | 3955       | 4773  |
| 15    | Railway                             | 128      | 2020       | 299   |
| 16    | Cars                                | 306      | 4830       | 8709  |

## 5.2 Experimental Setup

### Performance Criteria

We assessed the classification accuracy with the sensitivity (Sens), specificity (Spec), and their average (AVG):

$$\text{Sens} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \quad \text{Spec} = \frac{\text{TN}}{\text{TN} + \text{FP}}, \quad \text{AVG} = \frac{\text{Spec} + \text{Sens}}{2}, \quad (5.1)$$

where TP, TN, FP, and FN stand for true positive, true negative, false positive, and false negative, respectively. The Hamming loss between the ground truth and predictions was also computed. The loss is given as the mean over all samples, so that it lies in  $[0, c]$ , where  $c$  is the number of classes. Moreover, McNemar's statistical test was used to assess the pairwise significance of differences

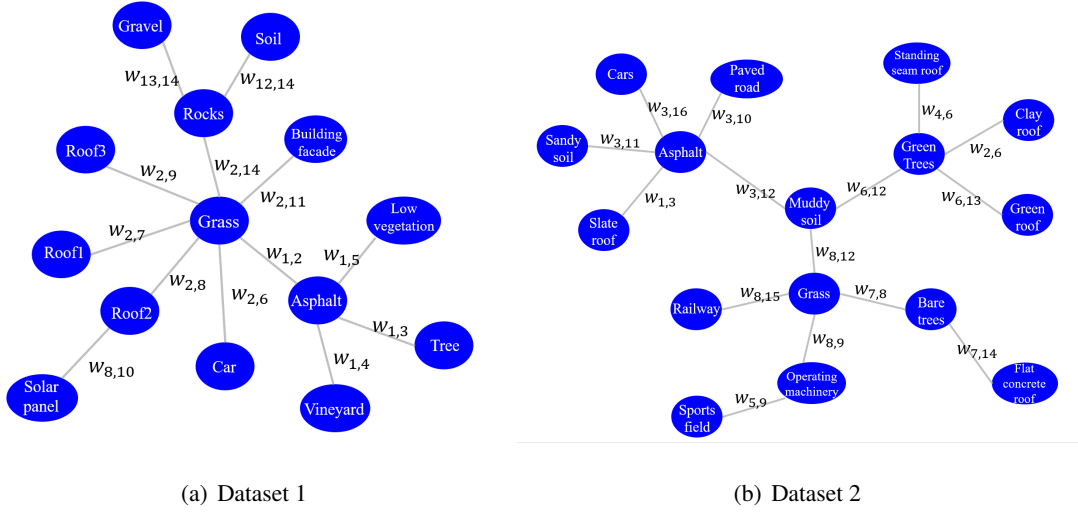


Figure 5.1: Output graph structure obtained for dataset 1 (a) and dataset 2 (b).

between two classifiers (Foody, 2004). This entailed the following calculation:

$$Z_{ij} = \frac{f_{ij} - f_{ji}}{\sqrt{f_{ij} + f_{ji}}}, \quad (5.2)$$

where  $f_{ij}$  indicates the number of samples correctly classified by the  $i$ -th classifier and wrongly classified by the  $j$ -th classifier. At the commonly used 5% significant level, the difference in accuracy of two classifiers is considered statistically significant if  $|Z_{ij}| > 1.96$ . The sign of  $Z_{ij}$  indicates the relative goodness of the two classifiers:  $Z_{ij} > 0$  means that the  $i$ -th classifier is more accurate than the  $j$ -th classifier, and vice versa.

We compared the proposed spatial and structural SVM (SSSVM) model with two closely related reference approaches: the standard SVM and the structural SVM (SSVM). The SSVM was implemented with the *pystruct* module in Python (Muller and Behnke, 2014). The standard SVM can be viewed as a special case of an SSVM simply by implementing it without including any structure in the module. In this way, we can solve the multiclass classification problem directly without dividing the output space, unlike in the so-called one-against-all classification approach.

### Structural Setup Based on Chow-Liu Algorithm

The structural models, namely the SSVM and SSSVM, require definition of the structural loss. One approach is to construct a taxonomic tree from prior knowledge about the target classes, like the one shown previously in Fig.3.1, and define the structural loss according to the tree. Here, however, we use another data-driven approach based on the Chow-Liu algorithm (Chow and Liu,

1968). This approach constructs a forest structure among the output variables, which expresses probabilistic relations among them based on pairwise mutual information. Figure 5.1 shows the actual output graphs obtained for datasets 1 (a) and 2 (b). The pairwise weights defined on the label pairs in the graphs (i.e., the edges) regulate the coexistence or co-absence of the labels. The weights were also optimized naturally in the training process by adding the corresponding categorical variables into the set of explanatory variables (see, e.g., Muller and Behnke (2014)). To embed the output-output relations, the test phase considered not only the unary potential of each class but also the pairwise potentials controlled by the pairwise weights.

### Parameter Setup

The regularization parameter  $C$  was tuned according to the validation errors in the set  $\{2^{-15}, \dots, 2^{-3}\}$ . For the SSSVM model,  $C$  was first tuned according to the validation errors of the SSVM model, and then the best smoothing parameter  $\lambda \in \{2^{-9}, \dots, 2^2\}$  was selected to minimize the validation error. In all the experiments we adopted a simple linear kernel, which does not require any additional parameters except for  $\lambda$ . The first-order neighborhood system was used.

## 5.3 Results

### Dataset 1: Civezzano Dataset

Table 5.3: Classification accuracies of the three classifiers on dataset 1

|         | SVM   | SSVM  | SSSVM        |
|---------|-------|-------|--------------|
| Sens    | 88.20 | 78.18 | 76.69        |
| Spec    | 73.04 | 89.47 | 91.57        |
| AVG     | 80.62 | 83.82 | <b>84.12</b> |
| Hamming | 3.597 | 1.607 | <b>1.355</b> |

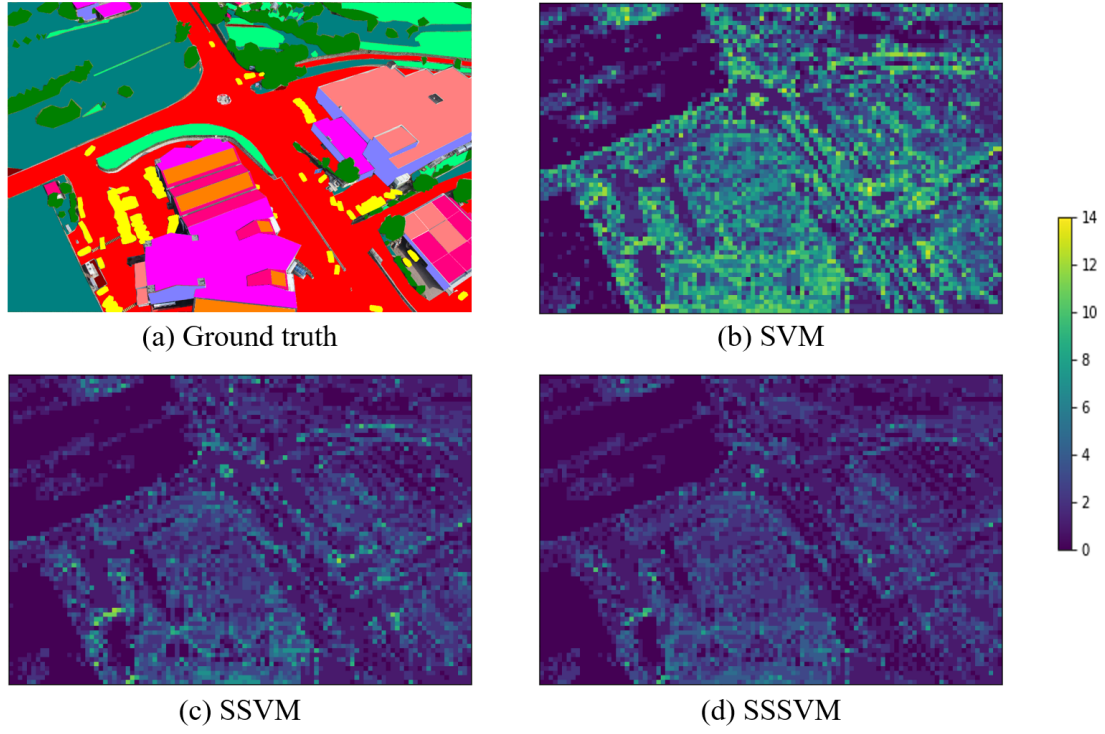


Figure 5.2: Ground truth for one test image in dataset 1 (a), and classification results obtained by the SVM (b), SSVM (c), and SSSVM (d). Each tile is colored according to the Hamming distance between the ground truth and prediction.

Table 5.4: Statistical comparison based on McNemar’s statistical test among the three models for dataset 1.

|       | SVM    | SSVM  |
|-------|--------|-------|
| SSVM  | -293.5 |       |
| SSSVM | -313.6 | -81.1 |

Table 5.3 lists the quantitative experimental results for dataset 1. The SSVM performed better than the standard SVM did, and both were outperformed by the proposed SSSVM model. The average accuracies of the SSVM and SSSVM were 83.82% and 84.12%, respectively. Although this gain of less than 1% might appear tiny, McNemar’s statistical test demonstrated the significance of this improvement, as listed in Table 5.4. Such improvement could also be confirmed by analyzing the classification maps, shown in Fig.5.2. Figure 5.2(a) shows the ground truth image of one of the test images. The tiles in Figs.5.2(b-d) are colored according to the Hamming distances between the

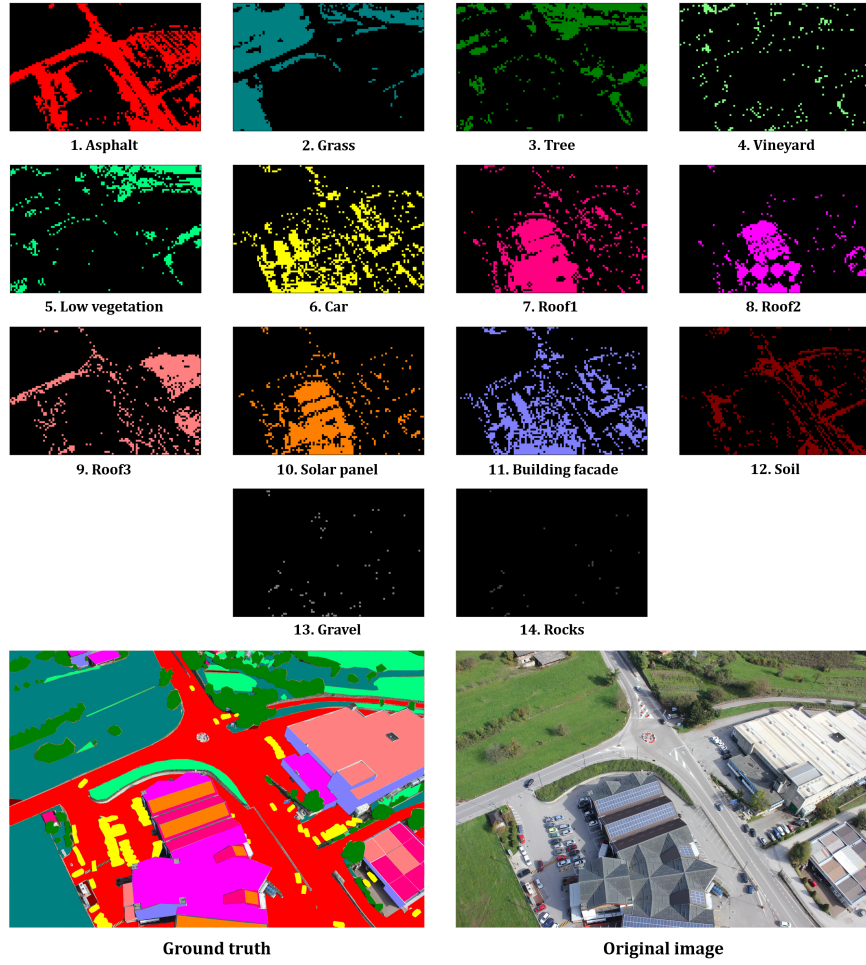


Figure 5.3: Multilabel classification maps obtained by the SSSVM classifier on one test image in dataset 1 (the same as in Fig.5.2), along with the corresponding ground truth and original image.

ground truth labels and the predicted labels obtained by the three classifiers. The map generated by the standard SVM, shown in Fig.5.2(b), is notably bright, indicating that many of the tiles were wrongly predicted. In contrast, the darkness of the SSSVM map in Fig.5.2(d) suggests that the classification accuracy of the proposed model was better than that of the SVM or even the SSVM.

Figure 5.3 shows the class-by-class classification maps generated by the SSSVM for the same test image. Note that multilabel classification generates as many classification maps as the number of classes. In general, the achieved results were good, even though, as expected, there were some confusion in discriminating between classes with similar visual appearance (e.g., “Grass” and “Trees”). In addition, some false positives emerged because of intrinsic class variability, which occurs especially when dealing with UAV imagery and a considerable number of

Table 5.5: Class-by-class accuracy achieved by the three models on dataset 1.

|       | SVM  |      |             | SSVM |      |             | SSSVM |      |             |
|-------|------|------|-------------|------|------|-------------|-------|------|-------------|
| Class | Sens | Spec | AVG         | Sens | Spec | AVG         | Sens  | Spec | AVG         |
| 1     | 86.6 | 85.4 | <b>86.0</b> | 76.2 | 89.0 | 82.6        | 73.3  | 89.8 | 81.5        |
| 2     | 87.4 | 89.2 | <b>88.3</b> | 83.3 | 92.0 | 87.7        | 82.6  | 92.3 | 87.5        |
| 3     | 89.2 | 78.5 | 83.9        | 85.4 | 82.6 | 84.0        | 88.5  | 81.8 | <b>85.2</b> |
| 4     | 82.2 | 63.7 | <b>72.9</b> | 44.7 | 91.2 | 67.9        | 25.4  | 96.2 | 60.8        |
| 5     | 86.5 | 65.0 | <b>75.7</b> | 67.4 | 82.5 | 74.9        | 65.9  | 83.3 | 74.6        |
| 6     | 94.0 | 79.6 | 86.8        | 87.7 | 89.2 | 88.4        | 90.5  | 90.0 | <b>90.3</b> |
| 7     | 83.9 | 80.0 | 81.9        | 77.4 | 90.2 | 83.8        | 76.9  | 92.0 | <b>84.5</b> |
| 8     | 88.2 | 74.9 | <b>81.5</b> | 66.1 | 95.9 | 81.0        | 60.4  | 97.0 | 78.7        |
| 9     | 96.3 | 71.9 | 84.1        | 79.6 | 87.5 | 83.6        | 80.7  | 91.7 | <b>86.2</b> |
| 10    | 99.1 | 74.6 | 86.9        | 97.1 | 89.3 | 93.2        | 96.1  | 93.0 | <b>94.5</b> |
| 11    | 87.5 | 71.2 | <b>79.4</b> | 73.3 | 84.0 | 78.7        | 71.2  | 87.4 | 79.3        |
| 12    | 96.0 | 61.5 | 78.8        | 82.5 | 83.8 | <b>83.2</b> | 78.1  | 86.1 | 82.1        |
| 13    | 73.7 | 65.3 | <b>69.5</b> | 18.1 | 95.5 | 56.8        | 4.3   | 99.1 | 51.7        |
| 14    | 94.0 | 70.9 | <b>82.4</b> | 40.2 | 98.7 | 69.5        | 22.3  | 99.6 | 60.9        |

classes simultaneously. Such confusion could be settled by fine-tuning the pre-trained CNN or customizing the sensors used for data acquisition. In overall terms, however, the classification maps exhibited a satisfactory description of the scene.

In greater detail, Table 5.5 compares the accuracies of the three classifiers for each class. Since the overall sensitivity of the SVM was high and its specificity was low, the SVM classifier tended to overestimate positive labels. This gave an advantage in classifying large classes such as class 1 (“Asphalt”) and class 2 (“Grass”), as listed in Table 5.5. Compared with the two reference models in terms of specificity, however, the SSSVM produced higher accuracies. For dominant classes (with large samples) such as classes 1, 2, and 3, the SSSVM’s average rates were comparable with those of the SSVM. For medium-size classes (i.e., those with intermediate sample sizes), on the other hand, the SSSVM achieved more accurate classification than the SSVM did. For small classes in the training set such as class 4 (“Vineyard”) and class 13 (“Gravel”), the SSVM classifier was a bit more accurate than the SSSVM classifier. This can be explained by the fact that

spatial contextual information tends to smooth out isolated labels or small structures. In general, however, the SSSVM model succeeded in improving both the average and the Hamming loss.

Regarding the output structure, according to the structure estimated by the Chow-Liu algorithm (Fig.5.1(a)), the “Asphalt” class has connections with the following four classes: “Grass,” “Tree,” “Vineyard,” and “Low vegetation.” This means that the classifier could increase or decrease penalties for co-occurrence of “Asphalt” with each of them. From the results, the penalty for co-occurrence of “Asphalt” and “Vineyard” was the biggest of the four. This is indeed reasonable because no samples in the training set shared both labels. The training process could thus distinguish unlikely events automatically.

Three additional aspects of the results deserve mention. First, many average criterion values for the SVM were greater than those for the SSVM and SSSVM. This was caused by the low cardinality of the dataset, meaning the number of labels held by each instance, on average. In fact, the cardinality of the dataset was 1.17. In this situation, even though the SVM overestimated false positives, specificity was not affected as much as sensitivity, because many true negatives still existed. This is why the SVM’s classification performance appears good at first glance. Even in such a situation, however, the structured models were superior to the standard SVM, and the Hamming distances supported their superiority, as well. Second, structured models work to reduce co-occurrence of incorrect labels in their predictions, by exploiting inter-label correlation. This suggests that the SSVM and SSSVM classifiers could learn that the cardinality of a dataset is low. The estimated weights of the graph edges confirmed this effect. The cardinalities of the predictions of the SSVM and SSSVM were only 2.27 and 1.98, respectively. Third, as a downside of the SSSVM as compared with the SSVM, spatial embedding sometimes worked in the wrong way at the boundaries of two objects, where it is typically harder to provide discriminative features. As a result, the SSSVM sometimes underestimated label occurrence at boundaries. Globally, however, as the tables show, the use of spatial information typically enabled generation of more reasonable outcomes.

## Dataset 2: Munich Dataset

Table 5.6: Classification accuracies of the three classifiers on dataset 2.

|         | SVM   | SSVM  | SSSVM        |
|---------|-------|-------|--------------|
| Sens    | 79.17 | 67.34 | 61.74        |
| Spec    | 64.50 | 79.21 | 85.82        |
| AVG     | 71.83 | 73.28 | <b>73.78</b> |
| Hamming | 5.43  | 3.53  | <b>2.67</b>  |

Table 5.7: Statistical comparison based on McNemar’s statistical test among the three models for dataset 2.

|       | SVM    | SSVM   |
|-------|--------|--------|
| SSVM  | -360.6 |        |
| SSSVM | -420.7 | -195.6 |

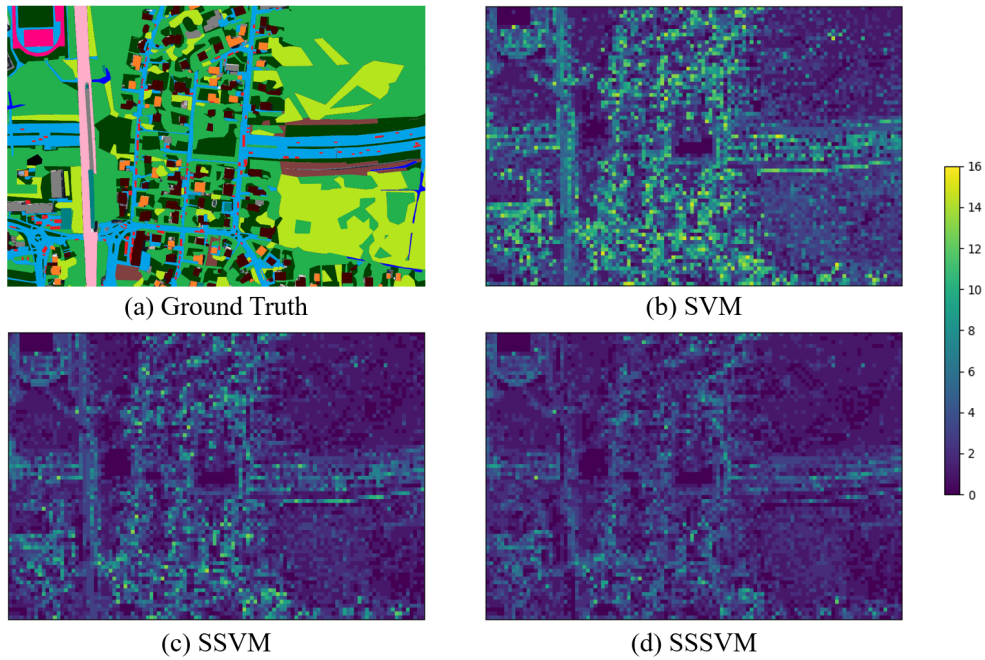


Figure 5.4: Ground truth for one test image in dataset 2 (a), and classification results obtained by the SVM (b), SSVM (c), and SSSVM (d). Each tile is colored according to the Hamming distance between the ground truth and prediction.

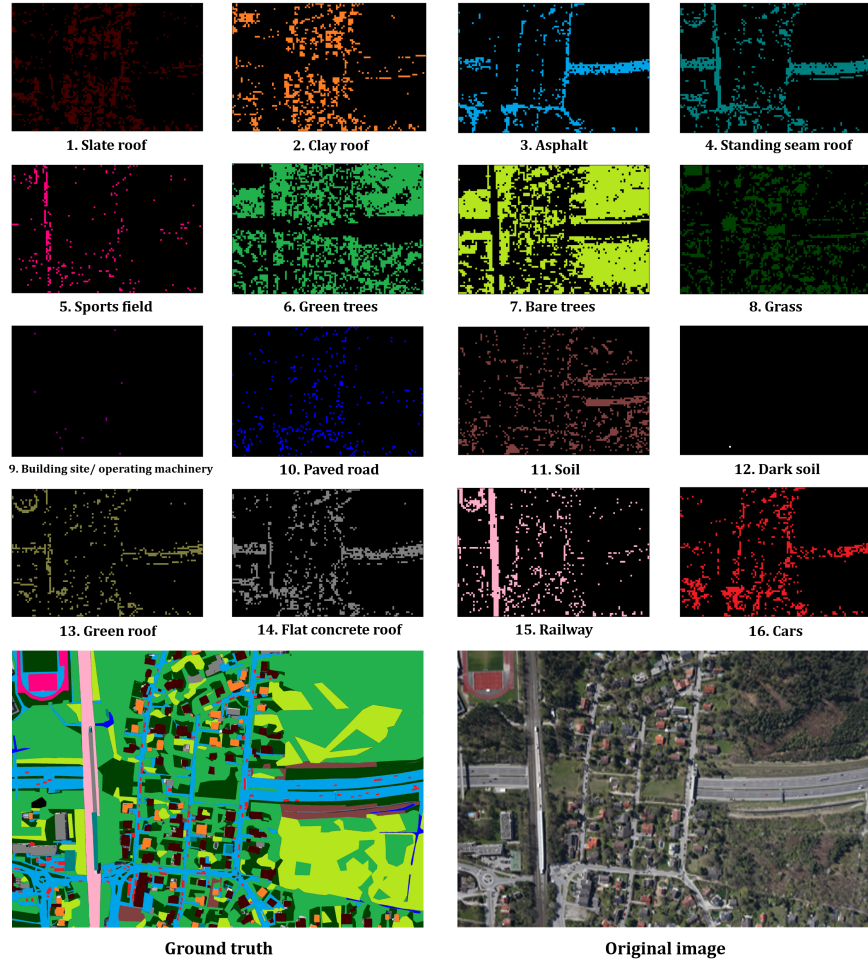


Figure 5.5: Multilabel classification maps obtained by the SSSVM classifier on one test image in dataset 2 (the same as in Fig.5.4), along with the corresponding ground truth and original image.

On dataset 2, the proposed SSSVM also outperformed the two reference models. Table 5.6 lists the classification accuracies. The average accuracy without structure was 71.83%. In contrast, the structured models, SSVM and SSSVM, achieved accuracy gains of approximately 1.5% and 2.0%, respectively. The inferred structure for these models was shown in Fig.5.1(b). By exploiting spatial information, the SSSVM performed better than the other two methods did, with an average metric and Hamming loss of 73.78% and 2.67, respectively. According to the McNemar's statistical test results listed in Table 5.7, the SSSVM significantly improved accuracy over the SVM and SSVM. The results suggest the potential of output structure with spatial modeling to make classification on EHR imagery more reliable. Figure 5.4 provides visual confirmation of this improvement, while Fig.5.5 shows the SSSVM classification maps obtained on one of the test images.

Table 5.8: Class-by-class accuracy achieved by the three models on dataset 2.

|       | SVM  |      |             | SSVM |      |             | SSSVM |       |             |
|-------|------|------|-------------|------|------|-------------|-------|-------|-------------|
| Class | Sens | Spec | AVG         | Sens | Spec | AVG         | Sens  | Spec  | AVG         |
| 1     | 87.5 | 63.9 | 75.7        | 81.2 | 72.5 | 76.8        | 76.5  | 78.0  | <b>77.3</b> |
| 2     | 94.0 | 70.1 | 82.1        | 91.4 | 76.7 | 84.0        | 89.9  | 81.3  | <b>85.6</b> |
| 3     | 77.8 | 77.9 | <b>77.9</b> | 62.8 | 88.7 | 75.7        | 52.9  | 92.7  | 72.8        |
| 4     | 88.3 | 62.9 | 75.6        | 83.0 | 72.1 | 77.5        | 82.4  | 73.8  | <b>78.1</b> |
| 5     | 58.0 | 60.1 | <b>59.5</b> | 42.3 | 76.2 | 59.2        | 13.6  | 93.6  | 53.6        |
| 6     | 76.0 | 79.7 | <b>77.9</b> | 62.6 | 88.8 | 75.7        | 55.8  | 91.8  | 73.8        |
| 7     | 85.3 | 62.0 | 73.7        | 71.7 | 77.8 | 74.8        | 70.2  | 81.1  | <b>75.7</b> |
| 8     | 74.0 | 76.9 | <b>75.5</b> | 65.3 | 84.7 | 75.0        | 60.0  | 88.2  | 74.1        |
| 9     | 81.8 | 59.9 | <b>70.9</b> | 42.5 | 90.4 | 66.5        | 6.91  | 99.4  | 53.2        |
| 10    | 60.2 | 57.0 | 58.6        | 45.2 | 74.3 | <b>59.8</b> | 20.5  | 89.5  | 55.0        |
| 11    | 87.5 | 56.8 | 72.2        | 76.7 | 73.5 | 75.1        | 74.9  | 81.2  | <b>78.1</b> |
| 12    | 97.8 | 54.8 | <b>76.3</b> | 4.80 | 99.1 | 52.0        | 0.0   | 100.0 | 50.0        |
| 13    | 76.6 | 62.6 | 69.9        | 71.0 | 72.8 | 71.9        | 66.3  | 85.1  | <b>75.7</b> |
| 14    | 77.3 | 62.6 | 69.9        | 66.4 | 71.9 | 69.2        | 63.9  | 76.3  | <b>70.1</b> |
| 15    | 91.3 | 63.2 | 77.3        | 91.0 | 75.8 | 83.4        | 90.0  | 85.2  | <b>87.6</b> |
| 16    | 80.7 | 76.1 | 78.4        | 76.8 | 79.4 | 78.1        | 81.3  | 78.4  | <b>79.9</b> |

Table 5.8 lists the accuracies achieved for each class. The SVM model again performed well for large classes such as classes 3 (“Asphalt”), 6 (“Green trees”), and 8 (“Grass”). The SSSVM classifier handled medium-size classes well, while the SSVM tended to preserve small classes such as class 5 (“Sports field”) and class 9 (“Building site”), which benefitted from the inter-label relationships captured in the structured model.

## 5.4 Summary

Up to this chapter, we have reviewed an EHR imagery classification strategy composed of coarse description, feature extraction, and multilabel classification. Coarse image description was achieved by subdividing images into tiles. Multilabel binary descriptors were then added to the tiles, with the goal of creating more understandable classification maps than those obtained by simple pixel-

wise classification. Discriminative feature representation for the tiles was implemented with the pre-trained GoogLeNet feature extractor. The next step was to develop a spatial and structural multilabel classifier, based on a structural SVM, by reformulating the original optimization problem. Through experiments on artificial data, HSI, and UAV imagery, the novel classifier performed better than conventional approaches did in both multiclass and multilabel classification.

## Chapter 6

# Unsupervised Feature Extraction Using Autoencoders

Up to this chapter, we have examined how to embed both spatial and structural information into classifiers and verified the effectiveness of the new approach. We have not, however, fully discussed feature representation. Instead, the pre-trained GoogLeNet was simply applied to generate features. The subsequent experimental results in **Chapters 4 and 5** showed, however, that generating a discriminative feature set is necessary to fully benefit from the potential of spatial modeling.

Assume that a spectral image of size  $height \times width \times bands$  is given. The objective of feature extraction in spectral image analysis is to represent the image with extracted  $l$ -dimensional features (typically,  $l < bands$ ), i.e., a compressed image of size  $height \times width \times l$ . Although many feature extraction approaches have been developed so far, the demand for them is still high (see, e.g., Romero et al. (2016)). With recent developments in sensor technology, as mentioned in the introductory chapter, we can obtain imagery containing hundreds of types of spectral information at high spectral resolution. Such images, however, have complex statistical properties such as redundancy and multi-collinearity, which make it harder to implement good classifiers. In addition, not much labeled data is accessible in this research community. Such challenges thus motivate consideration of more sophisticated feature extractors.

This chapter addresses the issue of feature extraction and is organized as follows. We first review the literature of feature extraction approaches for hyperspectral imagery (HSI). Then, a novel autoencoder-based unsupervised feature extraction approach is proposed. The results of

numerical experiments on several HSI datasets are used to evaluate both the new approach and conventional approaches.

## 6.1 Related Works

Early works on unsupervised feature extraction mainly extracted compressed spectral features from original inputs, through methods including principal component analysis (PCA) (Rodarmel and Shan, 2002; Ren et al., 2014) and the wavelet transform (Bruce et al., 2002). To extract higher-order topological structures in HSI, nonlinear feature transformation approaches like kernel PCA (KPCA) (Fauvel et al., 2009) and locally linear embedding (LLE) (Han and Goodenough, 2005) have been applied.

As we have often mentioned in this dissertation, an important characteristic of VHR hyperspectral data is its spatial property. Combining spatial and spectral information conveys more reasonable outcomes, but conventional methods often disregard spatial information. One commonly used approach is the so-called morphological profile (MP) (Fauvel et al., 2007). It was designed to extract spatial information from VHR imagery and extended by means of KPCA (Fauvel et al., 2009). Huang and L. Zhang (2009) proposed a mean-shift-based spatial feature extraction approach for spectral features constructed by nonnegative matrix factorization. Its performance demonstrated that spatial features can effectively complement spectral features. Tarabalka et al. (2009) proposed a spatial-spectral classification scheme combining segmentation and pixel-wise classification. It assigns a class label to each segmented region according to its most dominant class as given by pixel-wise classification.

The patch-based approach is a straightforward spatial extraction process (Li et al., 2015; Shu et al., 2018). It represents each pixel not only with its spectral intensity but also with those of its neighbors, meaning that the feature representation is given as a cuboid. Figure 6.1 illustrates this scheme, which crops a cuboid of size  $L \times L \times bands$  as the feature representation. Although this process increases the feature size, it enables simple application of spectral feature extraction approaches to vectorized inputs containing spatial context. For instance, Shu et al. (2018) applied K-means and PCA to such patches to organize spatial-spectral features. Fang et al. (2018) proposed a more flexible patch-based scheme that selects which neighboring pixels should be included. The process can be useful for robust feature extraction, especially on object boundaries.

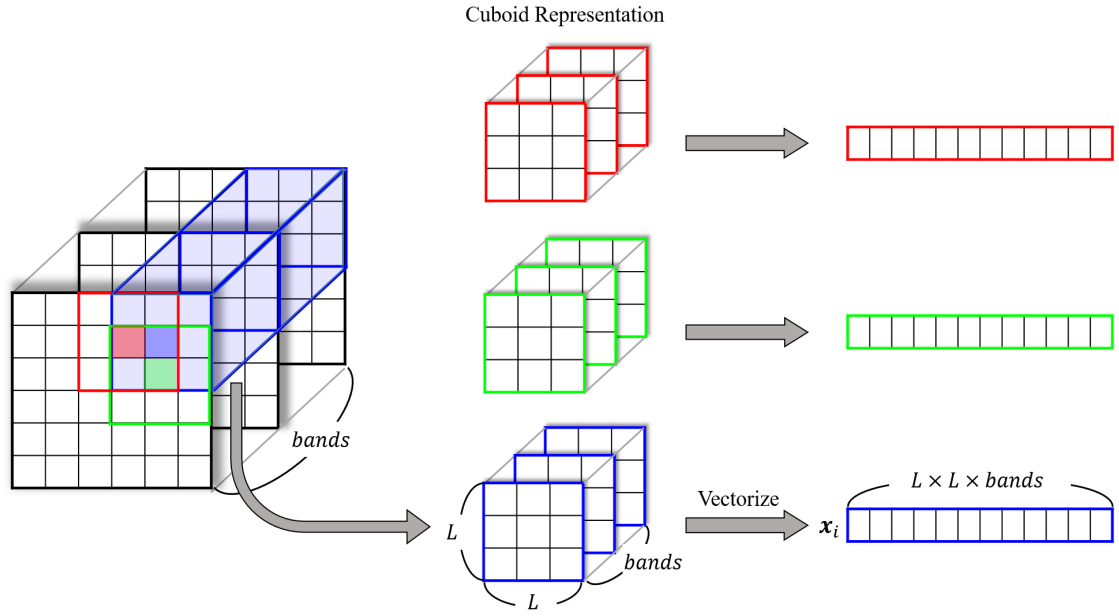


Figure 6.1: Illustration of a patch-based spatial feature extraction scheme.

An emerging feature learning tool in recent years is neural-network- (NN-)based feature extraction. In contrast to the handcrafted methods mentioned above, NN-based methods can extract hierarchical features automatically. In the context of supervised learning, they can also combine the feature extraction and classification steps, unlike handcrafted, feature-based methods. The autoencoder (AE) is one NN-based unsupervised feature learning architecture. The underlying idea is that the AE encodes features that are useful in reconstructing the original inputs. In this sense, AEs are often used as a tool for reconstructing missing data. For instance, Malek et al. (2018) used AEs to reconstruct areas obscured by clouds in multispectral imagery. As described in the literature, encoded features also have potential use in discriminating classes. Chen et al. (2014) and Tao et al. (2015) applied stacked AEs to HSI by training networks in a greedy, layer-wise manner to efficiently optimize the network parameters. Stacked denoising AEs and sparse AEs have also been applied to achieve more robust representation (Xing et al., 2016; Zhao et al., 2017). Ma et al. (2016) proposed an AE training process that can model relationships among samples. CNN-based AEs have also been applied, with cuboids used directly as inputs. The process of a CNN-based AE mainly consists of convolution, pooling, and deconvolution layers to reconstruct inputs from encoded features. Mou et al. (2018) designed an AE network based on a residual network architecture for feature extraction on HSI.

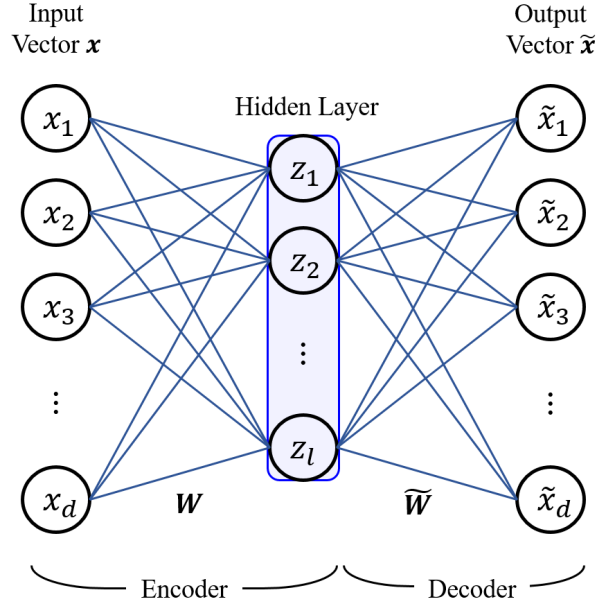


Figure 6.2: General architecture of an autoencoder (AE) network.

## 6.2 Strategy

The purpose of this study is to develop an efficient feature extraction approach for spectral land cover imagery. Here we assume the following situation:

1. pixel- or tile-wise classification, and
2. no or little availability of labeled data.

An important property of pixel- and tile-wise classification is spatial dependency: neighboring pixels tend to have the same class labels. This chapter proposes a novel, AE-based unsupervised feature extractor satisfying this property. The goal is to force a classifier followed by feature extraction to make smooth predictions. This should be achievable by forcing neighboring pixels to have similar feature representations in the previous feature extraction step. Therefore, an AE is designed to generate such features.

### 6.2.1 Autoencoder

The AE is an artificial neural network (ANN) architecture that basically consists of three layers, as shown in Fig.6.2: an input layer, hidden layer, and output layer. Let us define an input as  $\mathbf{x}_i = (x_{i1}, \dots, x_{id}) \in \mathbb{R}^d$  ( $i = 1, \dots, n$ ), where  $n$  and  $d$  denote the numbers of samples and

their dimension, respectively. Each  $\tilde{\mathbf{x}}_i \in \mathbb{R}^d$  is the reconstructed object of the corresponding input. The AE encodes every input into a different representation,  $\mathbf{z}_i = (z_{i1}, \dots, z_{il}) \in \mathbb{R}^l$ , in its encoding part, where  $l$  denotes the dimension of the encoded feature. The decoding part then generates the reconstructed input  $\tilde{\mathbf{x}}_i$ . Let us define the network parameters as  $\mathbf{W} = (\mathbf{w}_1, \dots, \mathbf{w}_l)^T$  and  $\mathbf{b} = (b_1, \dots, b_l)$ , which are a weight matrix of size  $l \times d$  and a bias vector of dimension  $l$  for the encoding part. The parameters in the decoding part,  $\tilde{\mathbf{W}} = (\tilde{\mathbf{w}}_1, \dots, \tilde{\mathbf{w}}_d)^T$  and  $\tilde{\mathbf{b}} = (\tilde{b}_1, \dots, \tilde{b}_d)$ , are similarly defined. The outputs of the hidden and reconstruction layers can be derived respectively as

$$\begin{aligned} z_{.j} &= a(\mathbf{w}_j^T \mathbf{x} + b_j) \quad (j = 1, \dots, l), \\ \tilde{x}_{.j} &= a(\tilde{\mathbf{w}}_j^T \mathbf{z} + \tilde{b}_j) \quad (j = 1, \dots, d), \end{aligned}$$

where  $\mathbf{w}_j$  and  $\tilde{\mathbf{w}}_j$  are the  $j$ -th rows of the respective weight matrices,  $\mathbf{W}$  and  $\tilde{\mathbf{W}}$ ;  $b_j$  and  $\tilde{b}_j$  are the  $j$ -th elements of the respective bias vectors,  $\mathbf{b}$  and  $\tilde{\mathbf{b}}$ ; and  $a : \mathbb{R} \rightarrow \mathbb{R}$  is a nonlinear activation function.

The network parameters can be derived by minimizing the empirical error between original inputs and their reconstructions. For the sake of simplicity, let us define a combined network parameter as  $\boldsymbol{\theta}$  and the corresponding space as  $\Theta$ , such that  $\boldsymbol{\theta} \in \Theta$ . Also, let us denote the combined encoder-decoder function as  $f : \mathbb{R}^d \rightarrow \mathbb{R}^d$ . Then, the optimal network parameters can be derived as

$$\begin{aligned} \boldsymbol{\theta}^* &= \arg \min_{\boldsymbol{\theta} \in \Theta} \sum_{i=1}^n \|\tilde{\mathbf{x}}_i - \mathbf{x}_i\|_2^2 \\ &= \arg \min_{\boldsymbol{\theta} \in \Theta} \sum_{i=1}^n \|f(\mathbf{x}_i; \boldsymbol{\theta}) - \mathbf{x}_i\|_2^2. \end{aligned}$$

The training is done by *back propagation* in the same way as for a standard NN. We could, of course, adopt a deeper network architecture by concatenating many hidden layers. In this case, greedy layer-wise training would be applied to efficiently train the network.

In the context of feature learning on remotely sensed data, the simplest approach assumes a network input of spectral information from each pixel. The AE network is thus trained to reconstruct inputs pixel by pixel. The new feature representation is then given by the encoded features, i.e.,  $\mathbf{z} = (z_1, \dots, z_l) \in \mathbb{R}^l$ . Patch-based approaches are often used to learn spatial context, in which case the inputs are given as cuboids or flattened spectral vectors.

### 6.2.2 Spatial Embedding into Autoencoder

Even though patch-based training can capture spatial context to some extent, the encoded features do not always satisfy the spatial property of neighboring pixels having similar representation. The desired feature representation is for a translated feature set of inputs to hold both spatial context information and similarity with the geometric neighborhood. The following optimization framework for AEs was designed to achieve that goal:

$$\begin{aligned}\theta^* &= \arg \min_{\theta \in \Theta} \sum_{i=1}^n \sum_{j \in N_i} \|\tilde{x}_i - x_j\|_2^2 \\ &= \arg \min_{\theta \in \Theta} \sum_{i,j=1}^n B_{ij} \|f(x_i; \theta) - x_j\|_2^2,\end{aligned}\tag{6.1}$$

where  $N_i$  is the set of indexes of the pixels neighboring pixel  $i$  (including  $i$  itself), and  $B$  is an adjacency matrix. This formulation requires that the reconstructed input of  $x_i$ ,  $\tilde{x}_i$ , should approximate not only its original input  $x_i$  but also its neighbors  $x_j$  ( $j \in N_i$ ). The embedded features of input  $x_i$  should also represent its neighborhood well, meaning that some of its features are shared among its neighbors. As a result, adjacent pixels are expressed by a feature representation resembling their neighborhood in a compressed space. As another effect, we can extract a potential representation by considering reconstruction of neighbors as well as original inputs.

Figure 6.3 graphically summarizes the new AE's workflow. In the generalized AE, the spectral intensities of pixels or their flattened patches are included in the network as inputs. They are forward-propagated through the network, and the encoded features reconstruct the inputs at the output layer. The reconstruction error is evaluated in terms of the original input and its neighboring inputs. The error is then back-propagated through the network, and the gradient updates the network parameters. After network training terminates, we extract the encoded features of the samples given as outputs of the hidden layer and use those features as explanatory variables in various kinds of analysis, such as image classification, segmentation, and restoration.

This formulation is not limited to simple AEs: it can be applied even to deep networks and convolutional AEs. Moreover, combining it with patch-based training can yield contextual features shared by neighbors. In this case, the formulation becomes

$$\theta^* = \arg \min_{\theta \in \Theta} \sum_{i,j=1}^n B_{ij} \|F(X_i; \theta) - X_j\|_2^2,$$

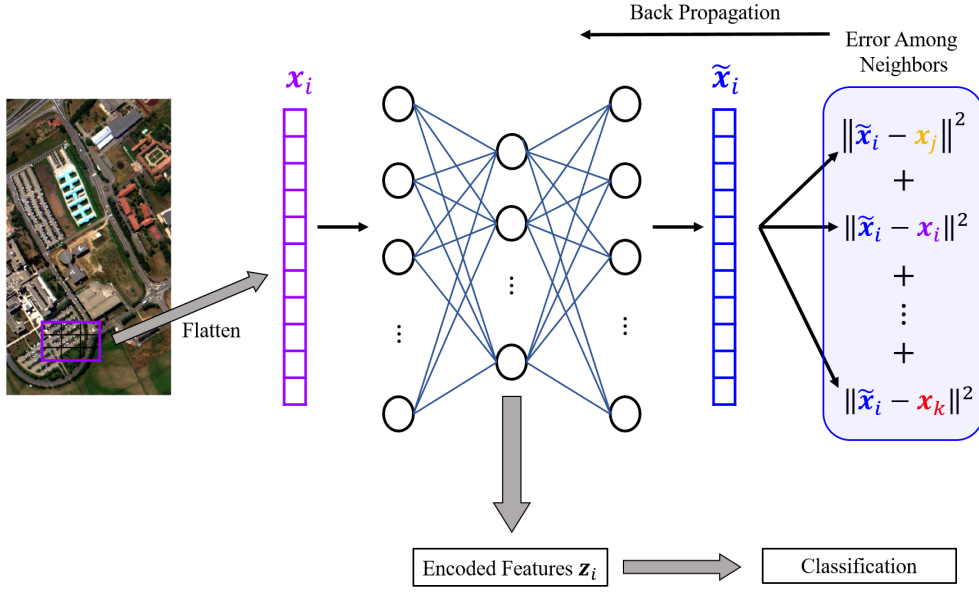


Figure 6.3: AE workflow on spectral imagery.

where each  $X_i$  denotes an input cuboid or vectorized patch, and  $F(\cdot)$  is an encoder-decoder function defined on the corresponding space.

### Relationships with Other Methodologies

The approach here was inspired by Wang et al. (2014), which termed the formulation as a *generalized autoencoder*. The original work sought to address issues in the computer vision field such as manifold learning and image processing. Their AE network explicitly minimized the residuals among neighbors through a framework like Eq.(6.1), with the neighbors defined according to the feature distance in the original input space. That is, the neighborhood of the  $i$ -th sample was basically determined according to  $\|x_i - x_j\|_2^2 (j = 1, \dots, n)$ . For remote sensing images, on the other hand, we can define the neighborhood more simply because pixels or patches naturally maintain a geographical spatial dependency defined by grids.

*Graph embedding* has a close relationship with the generalized AE. It constructs a low-dimensional manifold of the original input space or a graph by minimizing the following:

$$\sum_{i,j=1} B_{ij} \|z_i - z_j\|_2^2, \quad (6.2)$$

under a constraint to avoid a trivial solution. We can interpret this as direct control of sample relations described not in the original space but in the embedded space (Yan et al., 2007). In the

generalized AE, assume that  $\widetilde{W} = W^T$ , the dimension of the embedded feature  $l = 1$ , and the activation is an identity. Then, the optimal solution  $\mathbf{w}^*$  of the generalized AE can be derived as

$$\mathbf{w}^* = \arg \min \sum_{i,j} B_{ij} \|\mathbf{w} \mathbf{w}^T \mathbf{x}_i - \mathbf{x}_j\|_2^2. \quad (6.3)$$

Letting  $\mathbf{w}^T \mathbf{w} = c$  and  $z_i = \mathbf{w}^T \mathbf{x}_i$ , the equation becomes

$$\mathbf{w}^* = \arg \min_{\mathbf{w}^T \mathbf{w} = c} \sum_{i,j} B_{ij} \{ \|\mathbf{w}^T \mathbf{x}_i - \mathbf{w}^T \mathbf{x}_j\|_2^2 + (c - 2) z_i^2 \}. \quad (6.4)$$

When  $c = 2$ , i.e., the second term inside the sum operation vanishes, the generalized AE behaves similarly to graph embedding. When  $c > 2$ , the additional term prevents the encoded representation from being too large.

Furthermore, we can jointly minimize the trade-off between the reconstruction loss and the residuals of embedded features (see, e.g., Wang et al. (2016); Ma et al. (2016)). The formulation of Eq.(6.1), however, implicitly preserves sample relationships in the input space even in the embedded space. Wang et al. (2014) showed in their experiments with the MNIST handwritten image dataset that the embedded features obtained by the generalized AE constituted a more separable feature space than those obtained by normal AEs. These embedded features made the inter- and intra-class variability among samples smaller and larger, respectively.

A spectral feature pattern of a pixel is generally close to those of the neighbors, that is, it can be regarded as a feature of adjacent pixels added small noise. In this sense, we can expect that our AE plays in the same way as denoising AEs (Vincent et al., 2008; Xing et al., 2016). In denoising AEs, inputs are intentionally corrupted, which can make the trained network be robust to distortion of inputs and generate more generalized features.

### 6.3 Numerical Experiments

Numerical experiments were conducted on real-world remote sensing data to evaluate the potential of feature extraction based on the generalized AE with multiclass land cover classification. The experiments used the workflow and experimental design shown in Fig. 6.3.

### 6.3.1 Datasets

The experiments used six HSI datasets: Botswana, Indian Pines, Kennedy Space Center (KSC), Pavia Centre, Pavia University, and Salinas. Table 6.1 below outlines these datasets. we previously described two of the datasets, Indian Pines and Pavia University, in detail in **Chapter 4**. The details of all the datasets are available on the web, as well<sup>2</sup>.

Table 6.1: Characteristics of the HSI datasets used in the AE experiments.

| Data             | Image size         | Resolution/pixel | Labeled pixels | Bands | Classes |
|------------------|--------------------|------------------|----------------|-------|---------|
| Botswana         | $1476 \times 256$  | 30 m             | 3248           | 145   | 14      |
| Indian Pines     | $145 \times 145$   | 20 m             | 10249          | 200   | 16      |
| KSC              | $512 \times 614$   | 18 m             | 5211           | 176   | 13      |
| Pavia Centre     | $1096 \times 1096$ | 1.3 m            | 148252         | 102   | 9       |
| Pavia University | $610 \times 340$   | 1.3 m            | 42776          | 103   | 9       |
| Salinas          | $512 \times 217$   | 3.7 m            | 54129          | 204   | 16      |

### 6.3.2 Experimental Setups

#### Network Architecture

The standard AE network architecture was adopted with one hidden layer, as shown in Fig.6.2. The number of nodes in the hidden layer, i.e., the dimension of the reduced feature representation, was  $l \in \{16, 32, 64, 128, 256\}$ . The data values were normalized into the range  $[0, 1]$ . The sigmoid activation function was applied in the output layer to normalize the propagated inputs into that range. The output of the hidden layer was batch-normalized and activated with the ReLU activation function.

For training, the Adam optimizer was used with a learning rate of  $1.0 \times 10^{-3}$ . Other parameters had the default parameters of Tensorflow, the deep learning software used here. The number of epochs was 1000. In each epoch, the network was trained with shuffled samples. The number of samples used in each epoch depended on the dataset size (Pavia Centre: 30000; Indian Pines, Pavia University, Salinas: 10000; Botswana, KSC: 2000). The training batch size was 64 for all

<sup>2</sup>[http://www.ehu.eus/ccwintco/index.php/Hyperspectral\\_Remote\\_Sensing\\_Scenes](http://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes)

datasets.

In the experiments, the generalized AE was compared with normal, patch-based spatial-spectral AEs (see, e.g., Tao et al. (2015); Mou et al. (2018)) with patch sizes of 1, 3, and 5. For the generalized AE, the patch sizes were 1 and 3, and both first- and second-order neighborhood systems were used. Table 6.2 lists the various networks used in this comparison.

Table 6.2: Network architectures used in the AE experiments.

| Abbreviation       | Patch size | Loss type   | Neighborhood |
|--------------------|------------|-------------|--------------|
| Normal Patch-1     | 1          | Normal      | N/A          |
| Normal Patch-3     | 3          | Normal      | N/A          |
| Normal Patch-5     | 5          | Normal      | N/A          |
| General Patch-1 N1 | 1          | Generalized | First-order  |
| General Patch-1 N2 | 1          | Generalized | Second-order |
| General Patch-3 N1 | 3          | Generalized | First-order  |

## Performance Criteria

After AE training terminated,  $k$ -nearest neighbor (kNN) and linear SVM classifiers were applied to the encoded feature sets to assess the feature extraction capability. The number of neighbors in the kNN classifier was  $k = 5$ . The trade-off parameter of the linear SVM was 1.0. The labeled samples of each dataset were divided into training and test sets, as listed in the performance tables shown later. The performance criteria were the overall accuracy (OA) and average accuracy (AA) for each class. Five runs were performed, and the average OA and AA were compared among the models.

### 6.3.3 Results

#### Performance Overview

To overview the classification performance, we first fixed the dimension of the reduced feature representation as  $l = 64$ . Tables 6.3 and 6.4 list the results obtained by the kNN and SVM classifiers, respectively, on all six datasets. For almost all datasets and both classifiers, the General

Patch-3 N1 AE performed better than the other networks did. There were two types of exceptions. First, for the results obtained on the KSC dataset by the kNN classifier, the normal AE performed better than the generalized AE did. The SVM classifiers based on the generalized AE, however, produced higher accuracies than those of the kNN classifiers with the normal AE. This may have resulted from compatibility between features and classifiers or the parameter selections of the classifiers. Second, the accuracies of the Normal Patch-5 AE with the SVM were better on the Pavia Centre and Salinas datasets than those of the General Patch-3 N1 AE. Nevertheless the differences in OA and AA were only 0.1% at most. Comparing the results in Table 6.4 generated by networks with corresponding parameters (e.g., Normal and General Patch-3) shows that the generalized AEs always outperformed the normal AEs. With only a few exceptions, a similar situation occurred in the results obtained by kNN classifiers. In most cases, therefore, the generalized AEs performed better than the normal AEs did.

Table 6.3: Results obtained with kNN classifiers ( $k = 5, l = 64$ ).

|                  |        | Normal  |              |         | General    |            |              |
|------------------|--------|---------|--------------|---------|------------|------------|--------------|
| Dataset          | Metric | Patch-1 | Patch-3      | Patch-5 | Patch-1 N1 | Patch-1 N2 | Patch-3 N1   |
| Botswana         | OA     | 0.899   | 0.947        | 0.949   | 0.913      | 0.918      | <b>0.954</b> |
|                  | AA     | 0.916   | 0.956        | 0.957   | 0.928      | 0.932      | <b>0.963</b> |
| Indian Pines     | OA     | 0.533   | 0.612        | 0.655   | 0.635      | 0.657      | <b>0.675</b> |
|                  | AA     | 0.611   | 0.732        | 0.774   | 0.720      | 0.757      | <b>0.802</b> |
| KSC              | OA     | 0.714   | <b>0.776</b> | 0.748   | 0.740      | 0.746      | 0.755        |
|                  | AA     | 0.617   | <b>0.728</b> | 0.716   | 0.643      | 0.655      | 0.701        |
| Pavia Centre     | OA     | 0.981   | 0.988        | 0.988   | 0.982      | 0.984      | <b>0.990</b> |
|                  | AA     | 0.942   | 0.967        | 0.967   | 0.948      | 0.954      | <b>0.973</b> |
| Pavia University | OA     | 0.855   | 0.911        | 0.924   | 0.894      | 0.903      | <b>0.934</b> |
|                  | AA     | 0.864   | 0.905        | 0.906   | 0.900      | 0.902      | <b>0.930</b> |
| Salinas          | OA     | 0.898   | 0.920        | 0.931   | 0.906      | 0.906      | <b>0.934</b> |
|                  | AA     | 0.950   | 0.958        | 0.962   | 0.954      | 0.955      | <b>0.964</b> |

Table 6.4: Results obtained with SVM classifiers ( $C = 1.0, l = 64$ ).

|                  |        | Normal  |         |              | General    |              |              |
|------------------|--------|---------|---------|--------------|------------|--------------|--------------|
| Dataset          | Metric | Patch-1 | Patch-3 | Patch-5      | Patch-1 N1 | Patch-1 N2   | Patch-3 N1   |
| Botswana         | OA     | 0.919   | 0.961   | 0.966        | 0.929      | 0.930        | <b>0.974</b> |
|                  | AA     | 0.932   | 0.969   | 0.972        | 0.941      | 0.941        | <b>0.979</b> |
| Indian Pines     | OA     | 0.649   | 0.663   | 0.688        | 0.748      | <b>0.753</b> | 0.739        |
|                  | AA     | 0.729   | 0.776   | 0.808        | 0.835      | 0.840        | <b>0.854</b> |
| KSC              | OA     | 0.747   | 0.765   | 0.719        | 0.795      | <b>0.801</b> | 0.780        |
|                  | AA     | 0.664   | 0.710   | 0.677        | 0.725      | 0.727        | <b>0.735</b> |
| Pavia Centre     | OA     | 0.983   | 0.992   | <b>0.992</b> | 0.984      | 0.986        | 0.992        |
|                  | AA     | 0.937   | 0.970   | <b>0.973</b> | 0.939      | 0.950        | 0.972        |
| Pavia University | OA     | 0.896   | 0.926   | 0.931        | 0.915      | 0.911        | <b>0.948</b> |
|                  | AA     | 0.899   | 0.925   | 0.920        | 0.912      | 0.910        | <b>0.947</b> |
| Salinas          | OA     | 0.913   | 0.926   | <b>0.932</b> | 0.914      | 0.921        | 0.931        |
|                  | AA     | 0.954   | 0.959   | <b>0.965</b> | 0.955      | 0.960        | 0.964        |

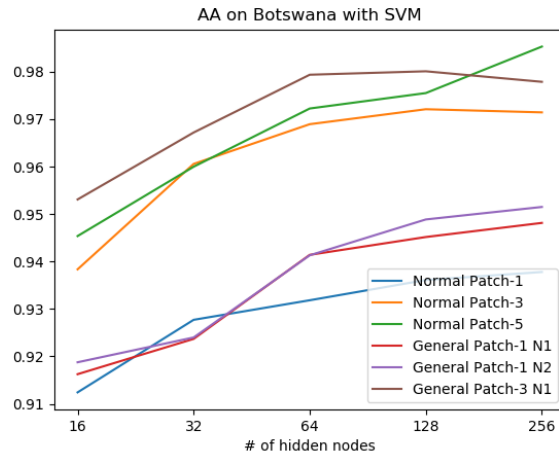
## Detailed Results

Here, let us consider the classification results in more detail. Figures 6.4-6.9 and their corresponding tables show the performance details on each dataset. Each figure shows the AA values obtained by SVM classifiers with the various AE architectures and the number of hidden nodes set as  $l \in \{16, 32, 64, 128, 256\}$ . Each table lists the accuracies obtained for each class by the Normal Patch-5 AE and General Patch-3 AE ( $l = 64$ ), which performed the best in their respective categories, as noted before.

First, note that the generalized AE provided an especially large gain when the number of hidden nodes was small. Consider the Pavia University results shown in Fig.6.8 as an example of this situation. The difference between the green curve (Normal Patch-5) and brown curve (General Patch-3) decreases as the number of hidden nodes increases. This indicates that the generalized AE could supply more general features to compensate the lack of information in spectral features when they were substantially compressed. Such phenomena could be ascertained on other datasets, as well.

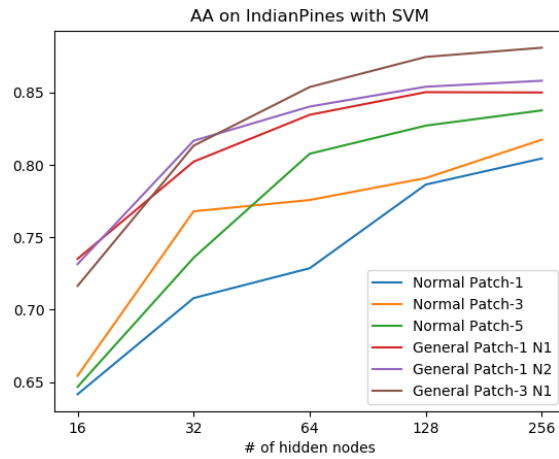
Second, features provided by the generalized AE were more informative especially when classification performance was poor. When classification accuracies were high ( $> 90\%$ ), e.g., the results on the Botswana, Pavia Centre, and Salinas datasets, even the normal AE could generate reasonable outcomes, meaning that the spectral features contained a certain amount of discriminative information. By contrast, on the Indian Pines, KSC, and Pavia University ( $l \leq 32$ ) data, the generalized AE had a larger contribution to accuracy improvement (see Figs.6.5, 6.6, and 6.8). Even the generalized AE with a patch size of 1 performed better than the normal AEs did, particularly when the number of hidden nodes was low. Neighborhood spectral information can thus make compressed features more discriminative even when the spectral information itself is not fully discriminative.

Finally, note that small classes were well discriminated even for VHR imagery of suburban areas, such as the Pavia Centre and Pavia University datasets. In these experiments, the uniform adjacency matrix worked, but there is still room for performance improvement by using weighted adjacency matrices.



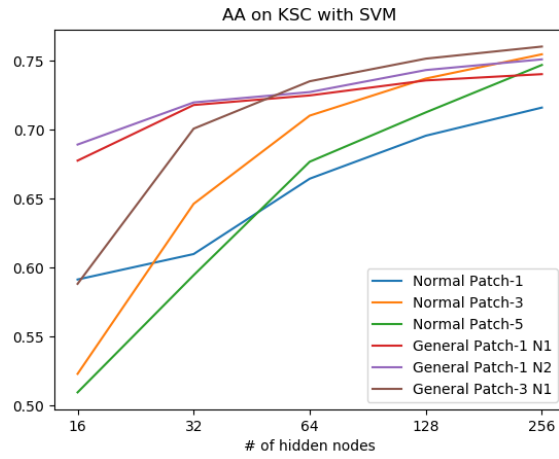
| Class | Train | Test | Normal Patch-5 | General Patch-3 |
|-------|-------|------|----------------|-----------------|
| 1     | 50    | 220  | <b>1.0</b>     | <b>1.0</b>      |
| 2     | 50    | 51   | 0.998          | <b>1.0</b>      |
| 3     | 50    | 201  | 0.999          | <b>1.0</b>      |
| 4     | 50    | 165  | <b>1.0</b>     | 0.999           |
| 5     | 50    | 219  | 0.897          | <b>0.902</b>    |
| 6     | 50    | 219  | 0.890          | <b>0.920</b>    |
| 7     | 50    | 209  | 0.995          | <b>1.0</b>      |
| 8     | 50    | 153  | <b>1.0</b>     | <b>1.0</b>      |
| 9     | 50    | 264  | 0.930          | <b>0.955</b>    |
| 10    | 50    | 198  | 0.974          | <b>0.975</b>    |
| 11    | 50    | 255  | 0.952          | <b>0.965</b>    |
| 12    | 50    | 131  | <b>0.998</b>   | <b>0.998</b>    |
| 13    | 50    | 218  | 0.987          | <b>0.996</b>    |
| 14    | 50    | 45   | <b>1.0</b>     | <b>1.0</b>      |
| OA    | 700   | 2548 | 0.966          | <b>0.974</b>    |
| AA    |       |      | 0.972          | <b>0.979</b>    |

Figure 6.4: AA and accuracies for each class on the Botswana dataset with the SVM.



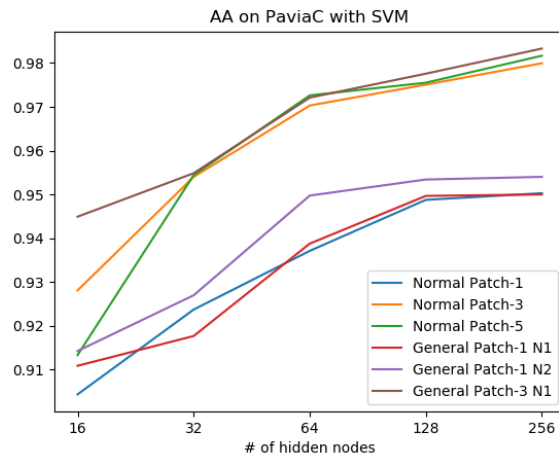
| Class | Train | Test | Normal Patch-5 | General Patch-3 |
|-------|-------|------|----------------|-----------------|
| 1     | 15    | 31   | 0.935          | <b>0.948</b>    |
| 2     | 50    | 1378 | 0.562          | <b>0.633</b>    |
| 3     | 50    | 780  | 0.624          | <b>0.656</b>    |
| 4     | 50    | 187  | 0.756          | <b>0.894</b>    |
| 5     | 50    | 433  | 0.879          | <b>0.932</b>    |
| 6     | 50    | 680  | 0.966          | <b>0.973</b>    |
| 7     | 15    | 13   | <b>0.969</b>   | 0.938           |
| 8     | 50    | 428  | <b>0.996</b>   | 0.986           |
| 9     | 15    | 5    | <b>1.0</b>     | <b>1.0</b>      |
| 10    | 50    | 922  | 0.637          | <b>0.679</b>    |
| 11    | 50    | 2405 | 0.561          | <b>0.567</b>    |
| 12    | 50    | 543  | 0.519          | <b>0.734</b>    |
| 13    | 50    | 155  | <b>0.996</b>   | 0.994           |
| 14    | 50    | 1215 | 0.841          | <b>0.921</b>    |
| 15    | 50    | 336  | 0.686          | <b>0.807</b>    |
| 16    | 50    | 43   | <b>1.0</b>     | <b>1.0</b>      |
| OA    | 695   | 9554 | 0.689          | <b>0.739</b>    |
| AA    |       |      | 0.808          | <b>0.854</b>    |

Figure 6.5: AA and accuracies for each class on the Indian Pines dataset with the SVM.



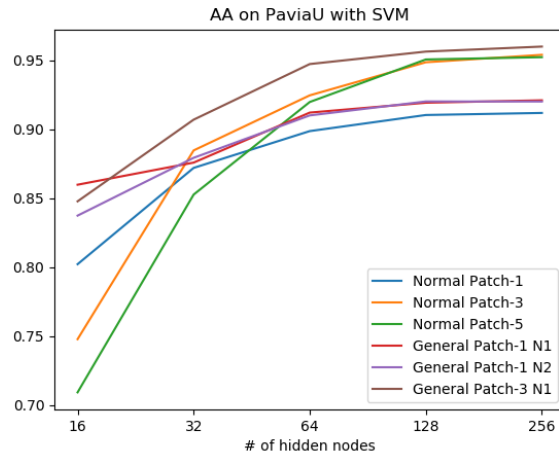
| Class | Train | Test | Normal Patch-5 | General Patch-3 |
|-------|-------|------|----------------|-----------------|
| 1     | 50    | 711  | <b>0.956</b>   | 0.926           |
| 2     | 50    | 193  | 0.690          | <b>0.844</b>    |
| 3     | 50    | 206  | 0.895          | <b>0.942</b>    |
| 4     | 50    | 202  | 0.124          | <b>0.171</b>    |
| 5     | 50    | 111  | 0.627          | <b>0.661</b>    |
| 6     | 50    | 179  | 0.392          | <b>0.447</b>    |
| 7     | 50    | 55   | <b>0.986</b>   | 0.956           |
| 8     | 50    | 381  | <b>0.692</b>   | 0.571           |
| 9     | 50    | 470  | 0.795          | <b>0.821</b>    |
| 10    | 50    | 354  | 0.412          | <b>0.711</b>    |
| 11    | 50    | 369  | 0.965          | <b>0.977</b>    |
| 12    | 50    | 453  | 0.441          | <b>0.614</b>    |
| 13    | 50    | 877  | 0.825          | <b>0.919</b>    |
| OA    | 650   | 4561 | 0.719          | <b>0.780</b>    |
| AA    |       |      | 0.677          | <b>0.735</b>    |

Figure 6.6: AA and accuracies for each class on the KSC dataset with the SVM.



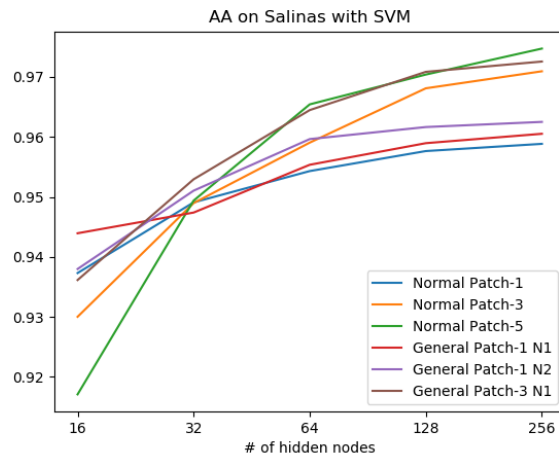
| Class | Train | Test   | Normal Patch-5 | General Patch-3 |
|-------|-------|--------|----------------|-----------------|
| 1     | 500   | 65471  | <b>1.0</b>     | <b>1.0</b>      |
| 2     | 500   | 7098   | <b>0.982</b>   | 0.97            |
| 3     | 150   | 2940   | <b>0.909</b>   | 0.907           |
| 4     | 150   | 2535   | 0.929          | <b>0.936</b>    |
| 5     | 500   | 6084   | <b>0.990</b>   | 0.985           |
| 6     | 500   | 8748   | 0.983          | <b>0.99</b>     |
| 7     | 500   | 6787   | 0.963          | <b>0.964</b>    |
| 8     | 500   | 42326  | <b>0.998</b>   | 0.998           |
| 9     | 150   | 2713   | 0.999          | <b>1.0</b>      |
| OA    | 3450  | 144702 | <b>0.992</b>   | 0.992           |
| AA    |       |        | <b>0.973</b>   | 0.972           |

Figure 6.7: AA and accuracies for each class on the Pavia Centre dataset with the SVM.



| Class | Train | Test  | Normal Patch-5 | General Patch-3 |
|-------|-------|-------|----------------|-----------------|
| 1     | 500   | 6131  | 0.924          | <b>0.944</b>    |
| 2     | 500   | 18149 | <b>0.959</b>   | 0.955           |
| 3     | 150   | 1949  | 0.866          | <b>0.902</b>    |
| 4     | 150   | 2914  | <b>0.928</b>   | 0.928           |
| 5     | 150   | 1195  | 0.991          | <b>0.998</b>    |
| 6     | 500   | 4529  | 0.937          | <b>0.968</b>    |
| 7     | 150   | 1180  | 0.858          | <b>0.917</b>    |
| 8     | 150   | 3532  | 0.821          | <b>0.917</b>    |
| 9     | 150   | 797   | 0.992          | <b>0.996</b>    |
| OA    |       |       | 0.931          | <b>0.948</b>    |
| AA    | 2400  | 40376 | 0.92           | <b>0.947</b>    |

Figure 6.8: AA and accuracies for each class on the Pavia University dataset with the SVM.



| Class | Train | Test  | Normal Patch-5 | General Patch-3 |
|-------|-------|-------|----------------|-----------------|
| 1     | 150   | 1859  | 0.984          | <b>0.998</b>    |
| 2     | 150   | 3576  | 0.998          | <b>1.0</b>      |
| 3     | 150   | 1826  | 0.985          | <b>0.989</b>    |
| 4     | 150   | 1244  | <b>0.998</b>   | 0.997           |
| 5     | 150   | 2528  | 0.986          | <b>0.994</b>    |
| 6     | 150   | 3809  | 0.998          | <b>1.0</b>      |
| 7     | 150   | 3429  | 0.995          | <b>0.996</b>    |
| 8     | 500   | 10771 | 0.852          | <b>0.860</b>    |
| 9     | 500   | 5703  | 0.997          | <b>0.999</b>    |
| 10    | 150   | 3128  | <b>0.923</b>   | 0.905           |
| 11    | 150   | 918   | <b>0.967</b>   | 0.956           |
| 12    | 150   | 1777  | <b>1.0</b>     | <b>1.0</b>      |
| 13    | 150   | 766   | <b>0.996</b>   | 0.994           |
| 14    | 150   | 920   | <b>0.990</b>   | 0.982           |
| 15    | 500   | 6768  | <b>0.790</b>   | 0.770           |
| 16    | 150   | 1657  | 0.987          | <b>0.991</b>    |
| OA    |       |       | <b>0.932</b>   | 0.931           |
| AA    | 3450  | 50679 | <b>0.965</b>   | 0.964           |

Figure 6.9: AA and accuracies for each class on the Salinas dataset with the SVM.

## 6.4 Summary

In this chapter, we have discussed spectral-spatial information extraction techniques for VHR HSI. We reviewed conventional approaches mainly based on neural networks such as CNNs and AEs. Then, we explained a novel feature extraction methodology based on those approaches, for remote sensing imagery. The formulation generalizes the loss functions of any AE network architecture to incorporate the spatial property of neighboring pixels having similar feature representations. Numerical experiments on six real-world HSI datasets showed that the new approach can supply more informative features for land cover classification.

## Chapter 7

# Conclusion

The dissertation considers the challenges of remote sensing data analysis raised by emerging sensing technologies in recent years. The technological developments in sensing platforms have enabled acquisition of very-/extremely- high resolution (VHR/EHR) images by unmanned aerial vehicles (UAVs). In addition, technologies such as hyperspectral sensors now provide images characterized by hundreds of spectral bands. Such evolving technologies have opened the door to development of more practical applications in various fields. At the same time, these technologies exhibit difficulties with which traditional analysis tools cannot deal. Thus, we have researched analytical tools to unlock the full benefits of data containing abundant information, such as EHR and VHR imagery and hyperspectral imagery (HSI).

The first part of this dissertation (**chapters 2-5**) focused on land cover classification on EHR imagery.

In **chapter 2**, we discussed the basic research strategies, including the problem formulation, feature extraction, and classification. To efficiently create land use maps on EHR images, the coarse description strategy is deployed. Following this formulation, images are subdivided into non-overlapping tiles, and labels are assigned to each tile. To represent each tile with discriminative features, a deep neural network (NN) called GoogLeNet was used as a feature extractor. We also reviewed support vector machines (SVMs) in this chapter to exploit their promise as multilabel classifiers in the following chapter.

In **chapter 3**, we reviewed the concepts and mathematical formulations of structural learning. Because EHR images typically contain label-label relations in the output space, we must in-

corporate such correlations into modeling. Structural SVMs can satisfy this requirement through a natural adaptation of traditional binary SVMs. For structural SVMs we thus reviewed their formulation, training process based on the cutting-plane algorithm, and applications to remote sensing data.

**Chapter 4** introduced a novel spatial classification methodology based on a structural SVM. The new methodology adds a term to enhance prediction smoothness into the objective function of the structural SVM. The new objective function is still formulated as a quadratic programming problem. By solving this spatial and structural optimization problem, we can incorporate not only output structure but also spatial contiguity into a multilabel SVM classifier with a single training step. To evaluate the effectiveness of the spatial modeling alone, numerical experiments on multiclass land cover classification on HSI were conducted without considering any structure. The results qualitatively and quantitatively showed the superiority of spatial modeling on this task.

**Chapter 5** covered numerical assessments of the new strategy on real-world EHR imagery. The spatial and structural SVM was applied to two types of EHR image datasets, along with previous approaches for reference. The experiments verified the positive impacts of incorporating both structural and spatial information into multilabel land cover classification on EHR imagery.

In **Chapter 6**, we addressed another challenge raised by recent developments in sensing technology: feature extraction for VHR HSI. An autoencoder- (AE-) based unsupervised feature extraction approach was proposed to reduce the divergence of feature representation among neighboring pixels. This approach was applied to the task of land cover classification on HSI. The experimental assessments showed that the novel approach can extract more discriminative, robust features for efficient land cover classification.

One of the main drawbacks of spatial modeling, including both spatial SVMs for classification and spatial AEs for feature extraction, concerns the treatment of small classes. Although the proposed methods performed well overall, this was not always the case for small classes. Imposing heavy penalties for misclassification of small classes or optimizing the weights in adjacency matrices might mitigate this problem (Xu and Chan, 2003; Sun et al., 2009).

As for tile-wise classification, the tile size must have a huge impact on object recognition performance, but we have not discussed the issue in this dissertation. Small-tile representation can

provide precise information about land use, but then recognition itself becomes hard. Similarly, large-tile representation also has pros and cons. We should determine the tile size by considering various aspects such as the object sizes under surveillance and the image resolution. General guidelines for such determination have not been fully discussed yet.

Another aspect to be considered is computational cost. Structural SVMs require repeatedly searching for the most violated constraint in every step. Fortunately, we can parallelize this process. In addition, an improved substitute for the cutting-plane algorithm has been developed (Guzman-Rivera et al., 2013). Systemizing these improvements in the future would make the proposed methodology more widely used in remote sensing data analysis.

Through this dissertation, we have focused mainly on land cover classification on VHR/EHR imagery. The proposed methodology can be applied, however, to various kinds of recognition problems. The input dependency is not limited to spatial relationships defined by image pixels. For instance, we believe that our model could also be applied to data containing temporal dependency. Moreover, the generalized AE-based feature extraction approach should have wide application in areas such as image restoration (Malek et al., 2018), because it is trained in an unsupervised fashion.

# Bibliography

- W. G. Aguilar, M. A. Luna, J. F. Moya, V. Abad, H. Parra and H. Ruiz, Pedestrian detection for UAVs using cascade classifiers with Meanshift, in *Proc. of IEEE 11th International Conference on Semantic Computing (ICSC)*, San Diego, CA, pp. 509-514, 2017.
- E. Alvarez-Cherman, J. Metz and M.C. Monard, Incorporating label dependency into the binary relevance framework for multi-label classification, *Expert Systems with Applications*, 39(2): 1647-1655, 2012.
- N. Audebert, B. L. Saux and S. Lefevre, How useful is region-based classification of remote sensing images in a deep learning framework?, in *Proc. of 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, Beijing, China, pp. 5091-5094, 2016.
- N. Audebert, B. L. Saux and S. Lefevre, Segment-before-Detect: Vehicle detection and classification through semantic segmentation of aerial images, *Remote Sensing*, 9(4), 368, 2017.
- V. Badrinarayanan, A. Kendall and R. Cipolla, SegNet: A deep convolutional encoder-decoder architecture for image segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12): 2481-2495, 2017
- R. B. Bapat, The Laplacian matrix of a graph, *The Mathematics Student*, vol. 65, pp. 214-223, 1996.
- M. B. Bejiga, A. Zeggada, A. Nouffidj and F. Melgani, A convolutional neural network approach for assisting avalanche search and rescue operations with UAV imagery, *Remote Sensing*, 9(2), 100, 2017.
- C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer-Verlag New York, 2006.
- J. A. Benediktsson, P. H. Swain and O. K. Ersoy, Neural network approaches versus statistical methods in classification of multisource remote sensing data, *IEEE Transactions on Geoscience and Remote Sensing*, 28(4): 540-552, 1990.
- J. M. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader and J. Chanussot, Hyperspectral unmixing overview: geometrical, statistical, and sparse regression-based approaches, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 5(2): 354-379, 2012.
- M. R. Boutell, J. Luo, X. Shen and C. M. Brown, Learning multi-label scene classification, *Pattern Recognition*, 37(9): 1757-1771, 2004.

- E. J. Bredensteiner and K. P. Bennett, Multicategory classification by support vector machines, *Computational Optimization and Applications*, 12(1-3): 53-79, 1999.
- L. M. Bruce, C. H. Koger and J. Li, Dimensionality reduction of hyperspectral data using discrete wavelet transform feature extraction, *IEEE Transactions on Geoscience and Remote Sensing*, 40(10): 2331-2338, 2002.
- Y. Chen, Z. Lin, X. Zhao, G. Wang and Y. 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.
- Y. Chen, H. Jiang, C. Li, X. Jia and P. Ghamisi, Deep feature extraction and classification of hyperspectral images based on convolutional neural networks, *IEEE Transactions on Geoscience and Remote Sensing*, 54(10): 6232-6251, 2016.
- M. K. Choi, H. G. Lee and S. C. Lee, Weighted SVM with classification uncertainty for small training samples, in *Proc. of 2016 IEEE International Conference on Image Processing (ICIP)*, Phoenix, AZ, USA, pp. 4438-4442, 2016.
- C. K. Chow and C. N. Liu, Approximating discrete probability distributions with dependence trees, *IEEE Transactions on Information Theory*, 14(3): 462-467, 1968.
- E. Chuvieco and R. G. Congalton, Application of remote sensing and geographic information systems to forest fire hazard mapping, *Remote Sensing of Environment*, 29(2): 147-159, 1989.
- P. Pereira Coltri, J. Zullo, R. Ribeiro do Valle Gonalves, L. A. S. Romani and H. S. Pinto, Coffee crop's biomass and carbon stock estimation with usage of high resolution satellites images, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 6(3): 1786-1795, 2013.
- C. Cortes and V. Vapnik, Support-vector networks, *Machine Learning*, 20(3): 273-297, 1995.
- K. Crammer and Y. Singer, On the algorithmic implementation of multiclass kernel-based vector machines, *Journal of Machine Learning Research*, vol. 2, pp. 265-292, 2001.
- L. David and A. H. Ballado, Vegetation indices and textures in object-based weed detection from UAV imagery, in *Proc. of 6th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*, Batu Ferringhi, Malaysia, pp. 273-278, 2016.
- U. Dogan, T. Glasmachers and C. Igel, A unified view on multi-class support vector classification, *Journal of Machine Learning Research*, 17(45): 1-32, 2016.
- K-B. Duan and S. S. Keerthi, Which is the best multiclass SVM method? An empirical study, in *Proc. of the 6th international conference on Multiple Classifier Systems*, Seaside, CA, USA, pp.278-285, 2005.
- M. Dundar, J. Theiler and S. Perkins, Incorporating spatial contiguity into the design of a support vector machine classifier, in *Proc. of 2006 IEEE International Symposium on Geoscience and Remote Sensing (IGARSS)*, Denver, CO, USA, pp. 364-367, 2006.

- L. Fang, N. He, S. Li, A. J. Plaza and J. Plaza, A new spatial-spectral feature extraction method for hyperspectral images using local covariance matrix representation, *IEEE Transactions on Geoscience and Remote Sensing*, doi: 10.1109/TGRS.2018.2801387, 2018.
- M. Fauvel, J. Chanussot, J. A. Benediktsson and J. R. Sveinsson, Spectral and spatial classification of hyperspectral data using SVMs and morphological profiles, in *Proc. of IEEE International Geoscience and Remote Sensing Symposium*, Barcelona, Spain, pp. 4834-4837, 2007.
- M. Fauvel, J. Chanussot and J.A. Benediktsson, Kernel principal component analysis for the classification of hyperspectral remote sensing data over urban areas, *EURASIP Journal on Advances in Signal Processing*, vol. 2009, Article ID 783194, 2009.
- R. Flamary and A. Rakotomamonjy, Support Vector Machine with spatial regularization for pixel classification, in *Proc. of International Workshop on Advances in Regularization, Optimization, Kernel Methods and Support Vector Machines: Theory and Applications*, Leuven, Belgium, pp. 73-74, 2013.
- G. M. Foody, Thematic map comparison: Evaluating the statistical significance of differences in classification accuracy, *Photogrammetric Engineering and Remote Sensing*, 70(5): 627-633, 2004.
- J. Gao, L. Xu, and F. Huang, A spectraltextural kernel-based classification method of remotely sensed images, *Neural Computing and Applications*, 27(2): 431-446, 2016.
- Y. Gu, Q. Wang and B. Xie, Multiple kernel sparse representation for airborne LiDAR data classification, *IEEE Transactions on Geoscience and Remote Sensing*, 55(2): 1085-1105, 2017.
- A. Guzman-Rivera, P. Kohli and D. Batra, DivMCuts: Faster training of structural SVMs with diverse m-best cutting-planes, in *Proc. of the 16th International Conference on Artificial Intelligence and Statistics*, Scottsdale, AZ, USA, pp.316-324, 2013.
- T. Han and D. G. Goodenough, Nonlinear feature extraction of hyperspectral data based on locally linear embedding (LLE), in *Proc. of 2005 IEEE International Geoscience and Remote Sensing Symposium*, 2005, pp.1237-1240, 2005.
- S. Hare, S. Golodetz, A. Saffari, V. Vineet, M-M. Cheng, S.L. Hicks and P.H.S. Torr, Struck: structured output tracking with kernels, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(10): 2096-2109, 2016.
- K. He, X. Zhang, S. Ren and J. Sun, Deep residual learning for image recognition, in *Proc. of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, pp.770-778, 2016.
- C-W. Hsu and C-J. Lin, A comparison of methods for multiclass support vector machines, *IEEE Transactions on Neural Networks*, 13(2): 415-425, 2002.
- X. Huang and L. Zhang, A comparative study of spatial approaches for urban mapping using hyperspectral ROSIS images over Pavia City, northern Italy, *International Journal of Remote Sensing*, 30(12): 3205-3221, 2009.

- S. Ioffe and C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, in *Proc. of the 32nd International Conference on Machine Learning*, Lille, France, vol. 37, pp. 448-456, 2015.
- Q. Jackson and D. A. Landgrebe, Adaptive Bayesian contextual classification based on Markov random fields, *IEEE Transactions on Geoscience and Remote Sensing*, 40(11): 2454-2463, 2002.
- T. Joachims, T. Galor and R. Elber, Learning to align sequences: a maximum-margin approach, *New Algorithms for Macromolecular Simulation*, pp. 57-69, Springer Berlin Heidelberg, 2006.
- R. Ke, Z. Li, S. Kim, J. Ash, Z. Cui and Y. Wang, Real-time bidirectional traffic flow parameter estimation from aerial videos, *IEEE Transactions on Intelligent Transportation Systems*, 18(4): 890-901, 2017.
- Y. Ke, L.J. Quackenbush and J. Im, Synergistic use of QuickBird multispectral imagery and LIDAR data for object-based forest species classification, *Remote Sensing of Environment*, 114(6):1141-1154, 2010.
- S. Koda, A. Zeggada, F. Melgani and R. Nishii, Spatial and structured SVM for multilabel image classification, *IEEE Transactions on Geoscience and Remote Sensing*, to appear.
- A. Krizhevsky, I. Sutskever and G. E. Hinton, ImageNet classification with deep convolutional neural networks, in *Proc. of Advances in Neural Information Processing Systems 25 (NIPS)*, Lake Tahoe, NV, USA, pp. 1097-1105, 2012.
- H. S. Le, I. Oparin, A. Allauzen, J. L. Gauvain and F. Yvon, Structured output layer neural network language model, in *Proc. of 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Prague, Czech Republic, pp. 5524-5527, 2011.
- J. Li, D. H. Ye, T. Chung, M. Kolsch, J. Wachs and C. Bouman, Multi-target detection and tracking from a single camera in Unmanned Aerial Vehicles (UAVs), in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, Korea, pp. 4992-4997, 2016.
- W. Li, C. Chen, H. Su and Q. Du, Local binary patterns and extreme learning machine for hyperspectral imagery classification, *IEEE Transactions on Geoscience and Remote Sensing*, 53(7): 3681-3693, 2015.
- Y. Li, H. Zhang and Q. Shen, Spectralspatial classification of hyperspectral imagery with 3D convolutional neural network, *Remote Sensing*, 9(1), 67, 2017.
- K. Liu and G. Mattyus, Fast multiclass vehicle detection on aerial images, *IEEE Geoscience and Remote Sensing Letters*, 12(9): 1938-1942, 2015.
- T. Liu, Y. Gu, X. Jia, J. A. Benediktsson and J. Chanussot, Class-specific sparse multiple kernel learning for spectralspatial hyperspectral image classification, *IEEE Transactions on Geoscience and Remote Sensing*, 54(12): 7351-7365, 2016.
- X. Ma, H. Wang and J. Geng, Spectralspatial classification of hyperspectral image based on deep auto-encoder, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(9): 4073-4085, 2016.

- K. Makantasis, K. Karantzas, A. Doulamis and N. Doulamis, Deep supervised learning for hyperspectral data classification through convolutional neural networks, in *Proc. of 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, Milan, Italy, pp. 4959-4962, 2015.
- S. Malek, F. Melgani, Y. Bazi and N. Alajlan, Reconstructing cloud-contaminated multispectral images with contextualized autoencoder neural networks, *IEEE Transactions on Geoscience and Remote Sensing*, 56(4): 2270-2282, 2018.
- F. Melgani and L. Bruzzone, Classification of hyperspectral remote sensing images with support vector machines, *IEEE Transactions on Geoscience and Remote Sensing*, 42(8): 1778-1790, 2004.
- T. Moranduzzo, F. Melgani, M. L. Mekhalfi and Y. Bazi, Multiclass coarse analysis for UAV imagery, *IEEE Transactions on Geoscience and Remote Sensing*, 53(12): 6394-6406, 2015.
- G. Moser and S. 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.
- L. Mou, P. Ghamisi and X. X. Zhu, Unsupervised spectralspatial feature learning via deep residual convdeconv network for hyperspectral image classification, *IEEE Transactions on Geoscience and Remote Sensing*, 56(1): 391-406, 2018.
- G. Mountrakis, J. Im and C. Ogole, Support vector machines in remote sensing: A review, *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(3): 247-259, 2011.
- A. C. Muller and S. Behnke, pystruct - Learning structured prediction in Python, *Journal of Machine Learning Research* vol. 15, pp. 2055-2060, 2014.
- J. S. Nath and C. Bhattacharyya, Maximum margin classifiers with specified false positive and false negative error rates, in *Proc. of the 2007 SIAM International Conference on Data Mining*, pp.35-46, 2007.
- V. N. Navid, S. L. Roberto and W. Stefan, Structured output prediction with hierarchical loss functions for seafloor imagery taxonomic categorization, in *Proc. of 7th Iberian Conference on Pattern Recognition and Image Analysis*, pp. 173-183, 2015.
- R. Nishii and S. Eguchi, Supervised image classification by contextual AdaBoost based on posteriors in neighborhoods. *IEEE Transactions on Geoscience and Remote Sensing*, 43(11): 2547-2554, 2005.
- C. Paris and L. Bruzzone, A three-dimensional model-based approach to the estimation of the tree top height by fusing low-density LiDAR data and very high resolution optical images, *IEEE Transactions on Geoscience and Remote Sensing*, 53(1): 467-480, 2015.
- J. E. Patino and J. C. Duque, A review of regional science applications of satellite remote sensing in urban settings, *Computers, Environment and Urban Systems*, Vol. 37, pp. 1-17, 2013.
- J. C. Platt, N. Cristianini and J. Shawe-Taylor, Large margin DAG's for multiclass classification, in *Proc. of Advances in Neural Information Processing Systems, NIPS'99*, MIT Press, pp.547-553, 2000.

- J. Ren, J. Zabalza, S. Marshall and J. Zheng, Effective feature extraction and data reduction in remote sensing using hyperspectral imaging [Applications Corner], *IEEE Signal Processing Magazine*, 31(4): 149-154, 2014.
- C. Rodarmel and J. Shan, Principal component analysis for hyperspectral image classification, *Surveying and Land Information Science; Gaithersburg*, 62(2):115-122, 2002.
- A. Romero, C. Gatta and G. Camps-Valls, Unsupervised deep feature Extraction for remote sensing image classification, *IEEE Transactions on Geoscience and Remote Sensing*, 54(3): 1349-1362, 2016.
- B. Salehi, Y. Zhang and M. Zhong, Automatic moving vehicles information extraction from single-pass WorldView-2 imagery, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 5(1):135-145, 2012.
- S. K. Sarkar, J. Das, R. Ehsani and V. Kumar, Towards autonomous phytopathology: Outcomes and challenges of citrus greening disease detection through close-range remote sensing, in *Proc. of 2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden, pp. 5143-5148, 2016.
- L. Shu, K. McIsaac and G. R. Osinski, Learning spatial-spectral features for hyperspectral image classification, *IEEE Transactions on Geoscience and Remote Sensing*, doi: 10.1109/TGRS.2018.2809912, 2018.
- P. Sollich, Bayesian methods for support vector machines: Evidence and predictive class probabilities, *Machine Learning*, vol.46, pp:21-52, 2002.
- A. L. Sumalan, D. Popescu and L. Ichim, Flood evaluation in critical areas by UAV surveillance, in *Proc. of 8th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, Ploiesti, Romania, pp. 1-6, 2016.
- A. Sun, E-P. Lim and Y. Liu, On strategies for imbalanced text classification using SVM: A comparative study, *Decision Support Systems*, 48(1):191-201, 2009.
- C. Szegedy, L. Wei, J. Yangqing, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, Going deeper with convolutions, in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, pp. 1-9, 2015.
- C. Tao, H. Pan, Y. Li and Z. Zou, Unsupervised spectralspatial feature learning with stacked sparse autoencoder for hyperspectral imagery classification, *IEEE Geoscience and Remote Sensing Letters*, 12(12): 2438-2442, 2015.
- Y. Tarabalka, J. A. Benediktsson and J. Chanussot, Spectralspatial classification of hyperspectral imagery based on partitional clustering techniques, *IEEE Transactions on Geoscience and Remote Sensing*, 47(8): 2973-2987, 2009.
- J. Thorsten, F. Thomas and Y. J. Chun-Nam, Cutting-plane training of structural SVMs, *Machine Learning*, 77(1): 27-59, 2009.
- I. Tschantaridis, T. Joachims, T. Hofmann and Y. Altun, Large margin methods for structured and interdependent output variables, *Journal of Machine Learning Research*, vol. 6, pp. 1453-1484, 2005.

- D. Tuia, J. Munoz-Mari, M. Kanevski and G. Camps-Valls, Structured output SVM for remote sensing image classification, *Journal of Signal Processing Systems*, 65(3): 301-310, 2011.
- V. Vapnik, An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 10(5): 988-999, 1999.
- P. Vincent, H. Larochelle, Y. Bengio and P-A. Manzagol, Extracting and composing robust features with denoising autoencoders, in *Proc. of the 25th international conference on Machine learning*, Helsinki, Finland, pp. 1096-1103, 2008.
- D. Wang, P. Cui, and W. Zhu, Structural deep network embedding, in *Proc. of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*, San Francisco, CA, USA, pp. 1225-1234, 2016.
- F. Wang, Fuzzy supervised classification of remote sensing images, *IEEE Transactions on Geoscience and Remote Sensing*, 28(2): 194-201, 1990.
- W. Wang, Y. Huang, Y. Wang, and L. Wang, Generalized autoencoder: a neural network framework for dimensionality reduction, in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition Workshops 2014*, Columbus, OH, USA, pp. 496-503, 2014.
- Y. Wang, Q. Chen, L. Liu, D. Zheng, C. Li and K. Li, Supervised classification of power lines from airborne LiDAR data in urban areas, *Remote Sensing*, 9(8), 771, 2017.
- J. Weston and C. Watkins, Support vector machines for multiclass pattern recognition, in *Proc. of European Symposium on Artificial Neural Networks 1999*, Bruges, Belgium, pp. 219-224, 1999.
- D. Wu and D-W. Sun, Advanced applications of hyperspectral imaging technology for food quality and safety analysis and assessment: A review — Part I: Fundamentals, *Innovative Food Science and Emerging Technologies*, vol. 19, pp. 1-14, 2013.
- S-H. Wu, K-P. Lin, C-M. Chen and M-S. Chen, Asymmetric support vector machines: low false-positive learning under the user tolerance, in *Proc. of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Las Vegas, Nevada, USA, pp.749-757, 2008.
- C. Xing, L. Ma, and X. Yang, Stacked denoise autoencoder based feature extraction and classification for hyperspectral Images, *Journal of Sensors*, vol. 2016, Article ID 3632943, 2016.
- P. Xu and A. K. Chan, Support vector machines for multi-class signal classification with unbalanced samples, in *Proc. of the International Joint Conference on Neural Networks*, vol. 2, pp. 1116-1119, 2003.
- S. Yan, D. Xu, B. Zhang, H. j. Zhang, Q. Yang and S. Lin, Graph embedding and extensions: A general framework for dimensionality reduction, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1): 40-51, 2007.
- X. Yang, Q. Song and A. Cao, Weighted support vector machine for data classification, in *Proc. of 2005 IEEE International Joint Conference on Neural Networks*, vol. 2, pp. 859-864, 2005.

- M-L. Zhang and Z-H. Zhou, ML-KNN: A lazy learning approach to multi-label learning, *Pattern Recognition*, 40(7): 2038-2048, 2007.
- M-L. Zhang, ML-RBF: RBF neural networks for multi-label learning, *Neural Processing Letters*, 29(2): 61-74, 2009.
- X. Zhang, G. Chen, W. Wang, Q. Wang and F. Dai, Object-based land-cover supervised classification for very-high-resolution UAV images using stacked denoising autoencoders, *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(7): 3373-3385, 2017.
- A. Zeggada and F. Melgani, Multilabel classification of UAV images with convolutional neural networks, *Proc. of 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, Beijing, China, pp. 5083-5086, 2016.
- A. Zeggada, F. Melgani and Y. Bazi, A deep learning approach to UAV image multilabeling, *IEEE Geoscience and Remote Sensing Letters*, 14(5): 694-698, 2017.
- A. Zeggada, Advanced classification methods for UAV imagery, Ph.D. dissertation, University of Trento, Italy, 2018.
- C. Zhao, X. Wan, G. Zhao, B. Cui, W. Liu and B. Qi, Spectral-spatial classification of hyperspectral imagery based on stacked sparse autoencoder and random forest, *European Journal of Remote Sensing*, 50(1): 47-63, 2017.

## Chapter 8

# Other Topic: Diurnal Transcriptome and Gene Network Represented Through Sparse Modeling in *Brachypodium distachyon*

As another topic, this chapter presents an application of statistical analysis to an issue in plant biology.

### 8.1 Introduction

#### 8.1.1 Research Background

Exploitation of environmentally friendly energy sources become an urgent issue in global terms. *Biofuels* are known as one such potential source. A biofuel is produced through biological processes such as agriculture and anaerobic digestion. Unlike fossil fuels, the sources of biofuels such as plants and crops are both renewable and ecologically beneficial because they absorb carbon dioxide during the growing process. National strategies all over the world have actually encouraged the use of bioethanol, an alcohol made mainly from starch crops, as a fuel for vehicles and other applications (see, e.g., Araujo et al. (2017)).

This situation requires growing large amounts of the plants used as biofuels. Crop yields in

recent years, however, have not been sufficient, often leading to imbalance between these crops' uses as food and as biofuel resources. Furthermore, the demand for biofuels is expected to increase in the future. We therefore need to develop techniques for mass production of biofuel energy resources.

One potential solution to this problem is *heterosis*, which means improved functionality of any biological quality in hybrid offspring. In the context of plant science, the improved functionalities concern plant biomass, growth speed, stress tolerance, and so on. Heterosis has provided such plants via genetic hybridization in natural fields on a long-term basis. If we could artificially induce heterosis into plants, we could produce biofuel resources more massively, constantly, and widely. This is still unachievable, however, because the molecular systems of heterosis have not yet been characterized. Once we understand these systems, they might be helpful in developing artificial heterosis technology and could even contribute to achieving a low-carbon society. One effective means for understanding the biological systems of plants, including heterosis, is the gene regulatory networks (GRN).

### **8.1.2 Gene Regulatory Network**

GRNs represent computationally inferred and experimentally verified molecular relations between genes, which facilitates understanding of gene functions in the context of biological processes (Barabasi and Oltvai, 2004; Davidson and Levin, 2005; Emmert-Streib et al., 2014a). With high-throughput data acquisitions in omics analyses (Fukushima et al., 2009; Moreno-Risueno et al., 2010), GRNs have been applied in a broad range of biological interests. They have been used to identify interactions of genes involved in particular biological processes, such as development (Pattanaik et al., 2014), physiological responses (Urano et al., 2010), metabolism (Tohge and Fernie, 2012), diseases (Emmert-Streib et al., 2014b), and evolution (Erwin and Davidson, 2009).

Being sessile, higher plants have evolved physiological systems to rapidly respond and adapt to environmental changes through alteration of their gene expression states (Hirayama and Shinzaki, 2010; Lopez-Maury et al., 2009; Scheres and van der Putten, 2017). To date, well-studied GRNs of some biological systems have been reported in plants (Krouk et al., 2013), such as auxin signaling (Havens et al., 2012; Sankar et al., 2011; Vernoux et al., 2014), circadian clock (Akman et al., 2012; Pokhilko et al., 2012; Salazar et al., 2009), and flower development (Espinosa-Soto,

2004; Jaeger et al., 2013; La Rota et al., 2011), mainly based on available information regarding the molecular bases of these biological functions in the model plant *Arabidopsis thaliana* (Krouk et al., 2013). Despite methodological advances in molecular biology to experimentally identify gene and/or protein interactions (Boeva, 2016; Braun et al., 2013; Reece-Hoyes et al., 2011), inferring GRNs based on high-throughput biological datasets has been a long-standing challenge in computational biology in terms of throughput, comprehensiveness, and cost-efficiency in various plant species.

### **8.1.3 Algorithms for Gene Network Inference**

Various algorithms for the inference of relations between genes from high-throughput gene expression data have been proposed to decode gene networks. Although gene co-expression networks (GCNs) do not provide dependencies estimated between genes, they have been widely applied in various species using a series of large-scale transcriptome datasets. Specifically, to assess co-expressed gene pairs, mutual information and Pearson's correlation for predictions of non-linear and linear relations among genes, respectively, have been used in a wide range of species (Lachmann et al., 2016; Provart, 2012; Steuer et al., 2002; Tzfadia et al., 2016). To date, GCNs have been applied to various analyses, such as prioritization of candidate genes, identification of functional modules, and prediction of regulatory factors (van Dam et al., 2017). To infer GRNs from gene expression data, which represent possible causalities between genes, various computational methods have been proposed (Lee and Tzou, 2009). For example, Bayesian inference and state-space modeling have been applied using time-series gene expression datasets to estimate influences along with chronological state transitions (Lebre, 2009; Wu et al., 2011). Through a visualization of gene relations, GRNs enable us to identify possible gene-gene relationships and hub genes that possibly regulate many subordinated genes and participate in various biological processes (Karlebach and Shamir, 2008; Thompson et al., 2015).

A kind of state-space modeling, the autoregressive with exogenous variables (ARX) model, has been used to identify relations between genes based on their time-series data, including gene expression profiles (Michailidis and D'Alche-Buc, 2013). Owing to the sparseness of GRNs, statistical sparse estimation methods such as Lasso and Elastic net have been applied to GRN estimation, and these results have suggested their feasible functions for their network inference

(Fujita et al., 2007; Shimamura et al., 2009; Shojaie and Michailidis, 2011). Among such sparse regularization approaches, smoothly clipped absolute deviation (SCAD) specifically provides an oracle property, which is a desired asymptotic property for sparse estimation (Fan and Li, 2001). Thus, it is expected that the oracle property performs significant roles in variable selection processes as compared with the conventional methods, such as Lasso and Elastic net. For example, an advantage of SCAD is that it estimates possible regulatory functions of genes in a GRN to explore those encoding transcription factors from a time-series transcriptome dataset (Wang et al., 2007).

#### **8.1.4 Model Plant: *Brachypodium distachyon***

*Brachypodium distachyon* is a small grass of the subfamily Pooideae, which emerged as a model species for temperate grasses, biofuel crops, and cool-season cereals such as wheat, barley, rye, and oats (Brkljacic et al., 2011; Kellogg, 2015; Mochida and Shinozaki, 2013; Vogel, 2016). Owing to its tractable features, such as small size, simple growth requirements, self-fertility, short life cycle, high-efficiency of *Agrobacterium*-mediated transformation, and small diploid genome size, this species is an ideal system for exploration of genes involved in important traits in grass species. Despite the rapid growth of transcriptome datasets from *B. distachyon* that are useful for constructing GCNs (Sibout et al., 2017), they have not been exploited to infer GRNs that determine relations of dependencies among genes. Thus, in addition to constructing a GCN, it will help infer possible regulatory relations among genes for a deeper understanding of cellular systems in *B. distachyon*.

#### **8.1.5 Contribution**

In the present study, we inferred the gene network of *B. distachyon* based on its diurnal transcriptome dataset. To identify genes that show cyclic expression patterns in the *B. distachyon* transcriptome, we analyzed a series of diurnal transcriptome data of *Brachypodium* leaves and identified its periodic genes. With the expression data of the periodic genes, we first constructed a GCN of the periodic genes. Thereafter, we computed possible dependencies between the periodic genes based on the group SCAD algorithm for sparse modeling to infer GRNs. Moreover, we identified several hub genes highly connected with subordinate genes, and discussed the estimated gene networks in the diurnal transcriptome of *B. distachyon*.

## 8.2 Materials and Methods

### 8.2.1 Plant Materials

The accession Bd21 of *B. distachyon* (W6 36678), which is a single seed-derived inbred line from accession PI 254867, was provided by the National Plant Germplasm System of USDA-ARS. Dry seeds were incubated on wet filter paper in a petri dish at 4°C in the dark for three days to synchronize germination. The germinated seeds were grown in a growth chamber at 25°C under a 16 h day photoperiod ( $60 \mu\text{mol m}^{-2} \text{s}^{-1}$ ) for four days. The plants were transplanted in pots filled with autoclaved Pro-Mix BX mycorrhizae (Premier Tech, Quebec, Canada). The potted plants were grown in a growth chamber at 22°C under a 20 h day photoperiod ( $100 \mu\text{mol m}^{-2} \text{s}^{-1}$ ). The plants were watered with 5,000-fold diluted Professional Hyponex 10-30-20 (Hyponex Japan, Osaka, Japan) every two or three days.

### 8.2.2 Transcriptome Analysis

We sampled leaves from *B. distachyon* Bd21 plants grown in a growth chamber for 19-20 days after synchronized germination for our transcriptome analysis based on RNA-seq using three biological replicates per sample. Total RNA was extracted from the leaves using the RNeasy Plant Mini Kit (Qiagen K.K. Japan, Tokyo, Japan) according to manufacturer's instructions. The RNA samples were assessed using an Agilent 2100 Bioanalyzer (Agilent Technologies Japan, Ltd., Tokyo, Japan). Libraries for stranded-RNA sequencing were constructed using a TruSeq Stranded mRNA Sample Preparation Kit (Illumina K.K., Tokyo, Japan) according to the manufacturer's instructions (TruSeq Stranded mRNA Sample Preparation Guide Rev. E; Illumina K.K., Tokyo, Japan). The libraries were assessed using an Agilent 2100 Bioanalyzer (Agilent Technologies Japan). Clonal clusters of the libraries were generated using cBot with a TruSeq Rapid PE Cluster Kit (Illumina K.K.) and sequenced using a HiSeq2500 sequencer with a TruSeq Rapid SBS Kit paired-end sequencing method to obtain 100-bp sequences. The FASTQ file of the sequence data was generated using HCS v2.0.12, Real time analysis v1.17.21.3, and Consensus Assessment of Sequence and Variation v1.8.2 (Illumina K.K.). Library preparation and sequencing were conducted by the Takara Bio Dragon Genomic Center (Takara Bio, Yokkaichi, Japan). The FASTQ files of the raw sequencing reads were submitted to the DDBJ Sequence Read Archive under

accession number DRA006158.

### 8.2.3 Functional Annotation

Gene Ontology (GO) terms for the *B. distachyon* genes were used from the gene annotation information downloaded from Phytozome (Bdistachyon\_314\_v3.1.annotation\_info<sup>1</sup>). Additional GO terms were associated with the *B. distachyon* genes using GO terms that related to transcripts for *A. thaliana* and rice in the “Best-hit-arabi-name” and “Best-hit-rice-name” row in the annotation file. The GO terms for *A. thaliana* and rice were used from the gene annotation information downloaded from Phytozome (Athaliana\_167\_TAIR10.annotation\_info.txt<sup>2</sup> and Osativa\_204\_v7.0.annotation\_info<sup>3</sup>). To reduce bias, GO terms that were assigned to more than 3,000 *Brachypodium* genes were excluded. The dataset of *B. distachyon* genes encoding putative transcription factors was retrieved from the PlanTFDB website (v4.0, JGI v3.1<sup>4</sup>).

The closest homologs of the *B. distachyon* genes in rice and sorghum were identified by a BLASTP search queried with the deduced protein sequences of *B. distachyon* genes against those of sorghum (annotated in a genome annotation of Phytozome 2.0) and rice (annotated in a genome annotation of MSU ver. 7.0) with a threshold of e-value less than 1e-5.

### 8.2.4 Gene Expression Analysis

The raw RNA-Seq reads were trimmed using Trimmomatic v0.32 (Bolger et al., 2014) with the LEADING:20 TRAILING:20 SLIDINGWINDOW:4:15 MINLEN:36 commands. The trimmed reads were mapped to the *B. distachyon* Bd21 genome sequence retrieved from the Phytozome website (Bdistachyon\_314\_v3.0<sup>5</sup>). The *B. distachyon* RNA-seq reads were mapped to the *B. distachyon* Bd21 genome sequence retrieved from the Phytozome website (Bdistachyon\_314\_v3.0) using the tmap (v.3.1.4) program using the default parameter settings. Read counts were com-

---

<sup>1</sup><http://genome.jgi.doe.gov/pages/dynamicOrganismDownload.jsf?organism=Bdistachyon>

<sup>2</sup><http://genome.jgi.doe.gov/pages/dynamicOrganismDownload.jsf?organism=Athaliana>

<sup>3</sup><http://genome.jgi.doe.gov/pages/dynamicOrganismDownload.jsf?organism=Osativa>

<sup>4</sup><http://planttfdb.cbi.pku.edu.cn/index.php?sp=Bdi>

<sup>5</sup><http://genome.jgi.doe.gov/pages/dynamicOrganismDownload.jsf?organism=Bdistachyon>

puted in each annotated gene in the Bd21 genome using the featureCounts program<sup>6</sup>, and based on the reads counts, reads per million mapped reads (RPM) values were calculated. The genes showing RPM value  $\geq 1$  in all the three biological replicates in at least one sampling time were defined as expressed genes. The differentially expressed genes (DEGs) were identified using the DESeq2 program (v.1.10.1) (Love et al., 2014) in R v.3.2.4, with a threshold of  $q$ -value  $< 1e-5$  (computed using the Benjamini-Hochberg procedure) to assess the deviation of gene expression between two days for the sampling.

### 8.2.5 Identification of Periodically Expressed Genes

To evaluate the periodicity of the gene expression patterns, we used the wavelet transformation techniques, which are implemented into the WaveletComp package<sup>7</sup> in R. On the basis of  $p$ -values for significance of periodic power spectra computed for the *B. distachyon* diurnal transcriptome dataset, genes with highly significant power ( $p < 0.01$ ) of the 24 h cycle were selected as candidates for periodically expressed genes in the diurnal transcriptome. In addition, using randomization selection for six series of transcriptome datasets for 24 h and 30 permutations of the selected datasets, genes with significant periodicity ( $p < 0.01$ ) for all 30 permutations were considered periodic genes in the diurnal transcriptome. To verify corresponding relations between the putative periodically expressed genes identified by wavelet analysis and sampling time, we carried out a correspondence analysis using the CA package<sup>8</sup> in R.

### 8.2.6 Discovery of Cis-Regulatory Motifs in Promoter Regions of Periodically Expressed Genes

To discover *cis*-regulatory motifs located in the promoter regions of each periodically expressed gene found in *B. distachyon*, previously reported *cis*-motifs regulating circadian gene expression (Adams and Carre, 2014) were used as queries to search against -2 kb upstream sequences from the putative transcription start site of the periodically expressed gene. To compare the *cis*-regulatory motifs discovered in the promoter regions with other sequenced Pooideae species, the *cis*-motifs were also searched against -2 kb upstream sequences from the putative transcription start site of

---

<sup>6</sup><http://bioinf.wehi.edu.au/featureCounts/>

<sup>7</sup><https://cran.r-project.org/web/packages/WaveletComp/index.html>

<sup>8</sup><https://cran.r-project.org/web/packages/ca/index.html>

the closest homologs in rice and sorghum.

### 8.2.7 GRN Estimation Based on ARX Models with Group SCAD

We utilized ARX models with group SCAD to estimate possible dependencies of genes in our *B. distachyon* diurnal transcriptome dataset. We performed a regression analysis of the expression patterns of a gene with those of all genes in time order lags 1, 2, and 3 to assess all their possible dependencies corresponding to three models: ARX(1), ARX(2), and ARX(3), respectively. To estimate gene-to-gene relations without any dependencies of specific time lags, we adopted the group SCAD penalty as the regularization term, which enabled us to carry out group estimation. In the ARX(2) model, for instance, each gene gives two explanatory variables: 1- and 2-order delayed variables, in which we assume that the two variables belong to the same group. To cope with observation noise and insufficient data points of the dataset, we applied a bootstrap-like random sampling approach, which generated a set of multiple data matrices while retaining its original temporal structure. Thereafter, we estimated GRNs for each of the data matrices, and assessed confidence for every edge based on the multiple GRNs (Appendix 1 and 2).

## 8.3 Results and Discussion

### 8.3.1 Diurnal Transcriptome of *B. distachyon* Leaves

To identify genes that show rhythmic expression patterns in *B. distachyon*, we sequenced RNAs from the leaves of *B. distachyon* Bd21 sampled through a diurnal cycle over 48 h. We sampled leaves of *B. distachyon* Bd21 grown under a 20 h light/4 h dark diurnal cycle over 48 h periods at 4 h intervals, a “long day” condition. Although the photoperiod condition did not deviate from the natural conditions, the condition has been used in various studies as a standard condition for *B. distachyon* Bd21 in a growth chamber (An et al., 2016; Bredow et al., 2016; Rancour et al., 2015; Vogel and Bragg, 2009). Transcriptome sequencing revealed that more than 99% of the total reads in each sample were uniquely mapped reads. We found that 16,287-17,011 genes are significantly expressed, supported by RPM values  $\geq 1$  of all three replicated samples at each time point (Fig.8.1(a)). Moreover, we verified that the gene expression patterns between the series of the first and second days were highly correlated, and the numbers of differentially expressed genes

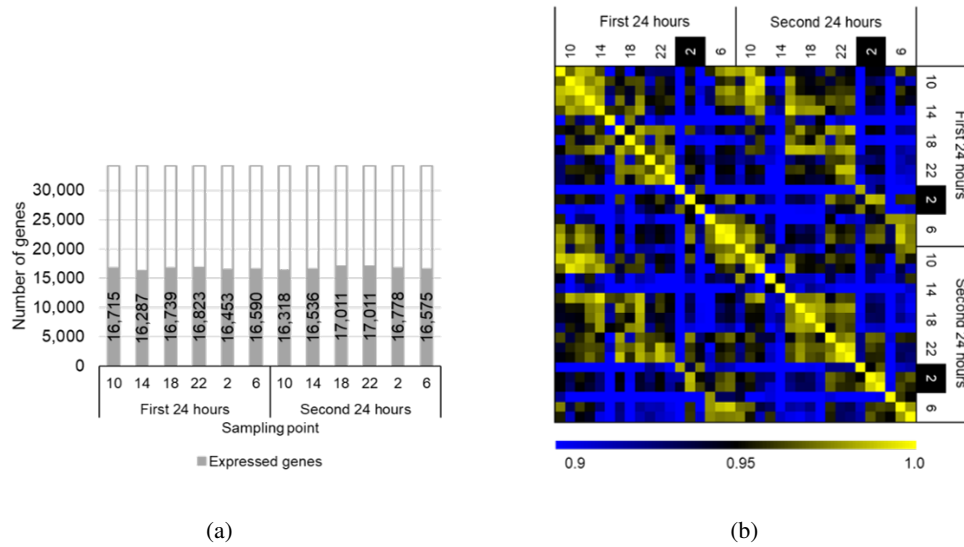


Figure 8.1: Overview of the diurnal transcriptome in *B. distachyon* leaves. (a) Number of genes significantly expressed (average RPM  $\geq 1$  in three biological replicates) in the diurnal transcriptome samples. (b) Correlation of the gene expression patterns obtained from the transcriptome datasets based on RPM values of 18,578 significantly expressed genes.

between the same time points of those two days were as low as 0.03% (i.e., six genes per 18,578 expressed genes across six time periods between two days; Fig.8.1(b)). This finding suggested that we sequenced the transcriptomes of similar physiological states of the growth stage in *B. distachyon* Bd21 over the 48 h periods. To examine gene expression changes through the diurnal samples, we compared gene expression patterns between all combinations across the six sampling time. As a result, we identified 5,520 DEGs with at least one sample combination, 47.4% of which are found between the samples from 2 to 6 o'clock, suggesting that about half of the DEGs alter their expression pattern from night to morning in response to light signaling in *B. distachyon*. Although 389 RNA-seq-based transcriptome datasets from *B. distachyon* are present in the public domain (NCBI SRA, as of September 4, 2017), our RNA-seq dataset is the first time-series dataset of a high-resolution diurnal transcriptome in the vegetative growth stage of *B. distachyon*. Therefore, the data will be valuable for understanding the physiological response through diurnal growth in the model grass. In the present study, we applied a 20 h light/4 h dark diurnal cycle, which is the most widely used growth condition for the short life cycle of *B. distachyon* under controlled conditions. It is expected that further analyses with diurnal transcriptome data from different day-night length conditions and/or different growth stages and tissues will facilitate a deeper understanding of molecular systems interacting with diurnal environmental perturbations

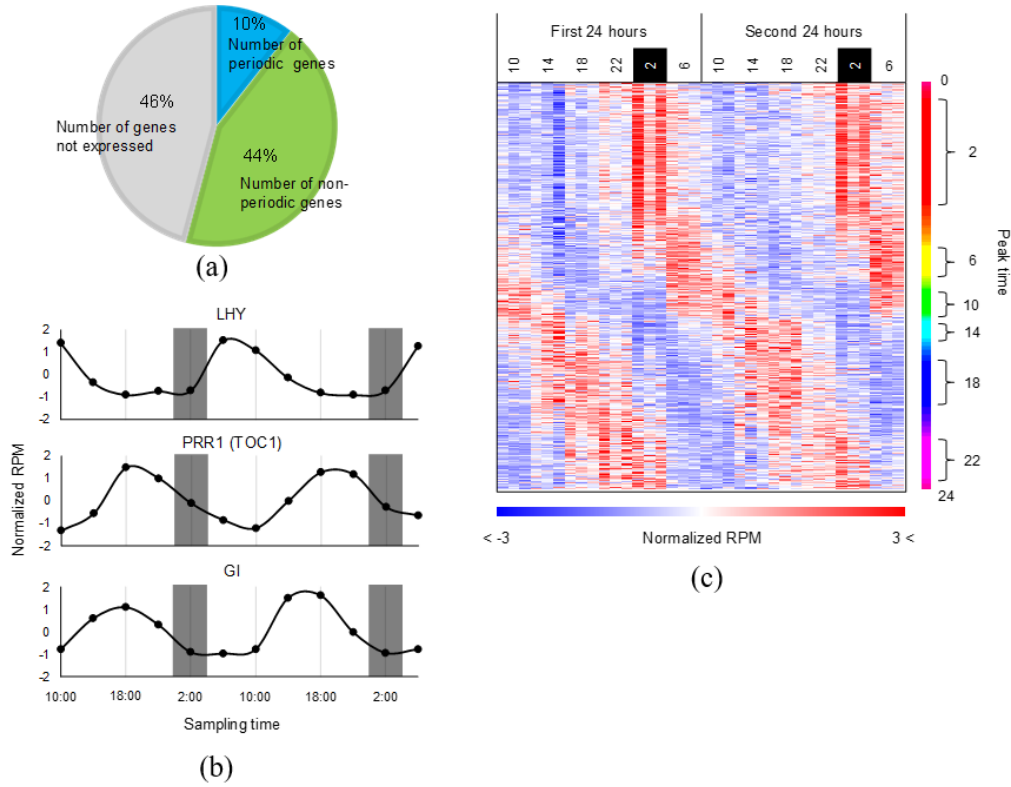


Figure 8.2: Periodic genes identified from the diurnal transcriptome data of *B. distachyon* leaves. (a) Distribution of the periodic genes expressed in *B. distachyon* leaves. The pie chart represents percentages of the periodic genes and non-periodic genes identified through our wavelet analysis. (b) Expression patterns of homologs of clock-related genes annotated in rice. (c) Expression patterns of the 3,621 periodic genes found in the *B. distachyon* transcriptome. The heatmap represents expression patterns of genes based on the normalized RPM values with a color gradient from blue to red. The genes are vertically sorted along with their estimated peak time.

in temperate grasses.

### 8.3.2 Identification of Periodic Genes in *B. distachyon*

To infer transcriptional relationships between genes, we first obtained a comprehensive list of genes whose expression patterns are either positively or negatively correlated in the diurnal transcriptome in *B. distachyon*. Using wavelet analysis, we found that 3,621 genes (i.e., 10% of the protein-coding genes annotated in the *B. distachyon* Bd21 genome and 19.5% of expressed genes (i.e., 3,621 genes per 18,578 expressed genes) in the series of RNA-seq data ( $\text{RPM} \geq 1$ ))

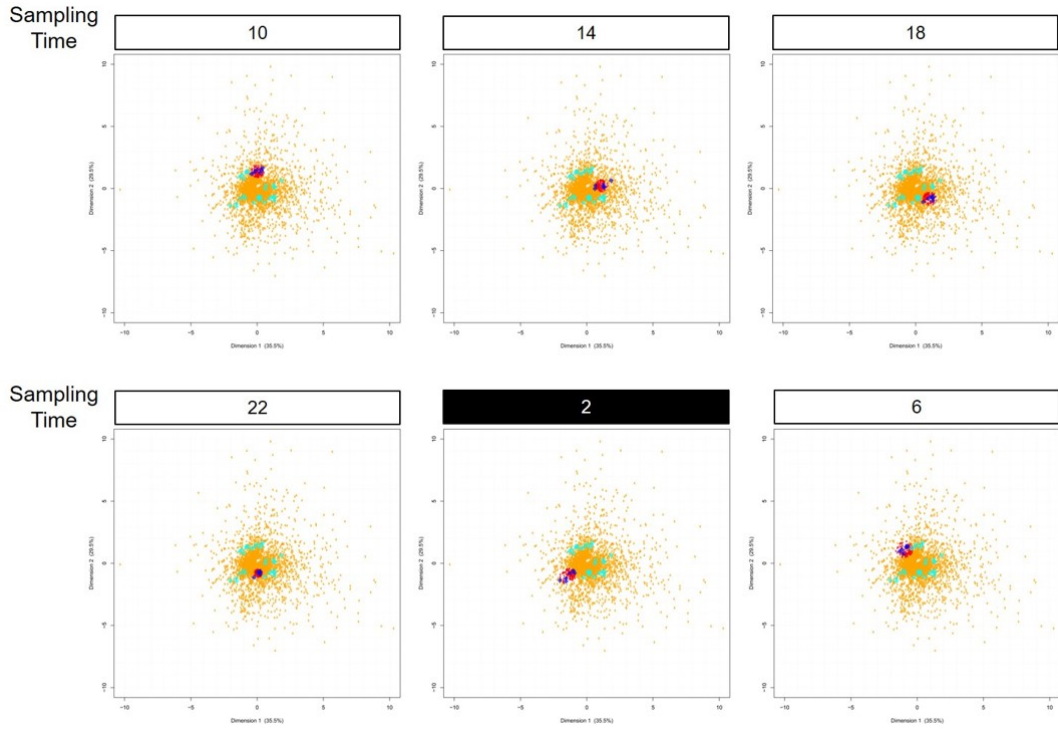


Figure 8.3: Results of a correspondence analysis between the putative periodically expressed genes and the sampling time. Dots (orange and red) and rectangles (green and blue) indicate genes and time periods, respectively. Blue rectangles and red dots represent relations between time periods and genes that show the top 100 genes ordered by correspondence contribution.

showed a rhythmic expression pattern through the diurnal transcriptome datasets (Fig.8.2(a)). To examine the adequacy of our wavelet screening, we verified the expression patterns of the homologs of genes that are involved in the circadian clock in plants such as rice (Matsuzaki et al., 2015) (Fig.8.2(b)). Through a correspondence analysis with the 3,621 periodically expressed genes, we also verified that these genes show corresponding relations with the time (Fig.8.3). The heatmap of expression patterns for the periodic genes clearly represented a diagonal striped pattern (Fig.8.2(c)), which is a typical representation of expression patterns of periodic genes, as reported in previous studies on model species such as *Arabidopsis* (Covington et al., 2008) and tomato (Higashi et al., 2016). These results indicate that we successfully identified the periodic genes through wavelet analysis with the diurnal transcriptome dataset in *B. distachyon*. The plant circadian clock plays a crucial role in photosynthesis and regulates carbohydrate and nitrogen metabolism (Adams and Carre, 2014). Moreover, it is involved in physiological responses to various environmental stresses in plants (Grundy et al., 2015). Therefore, the comprehensive

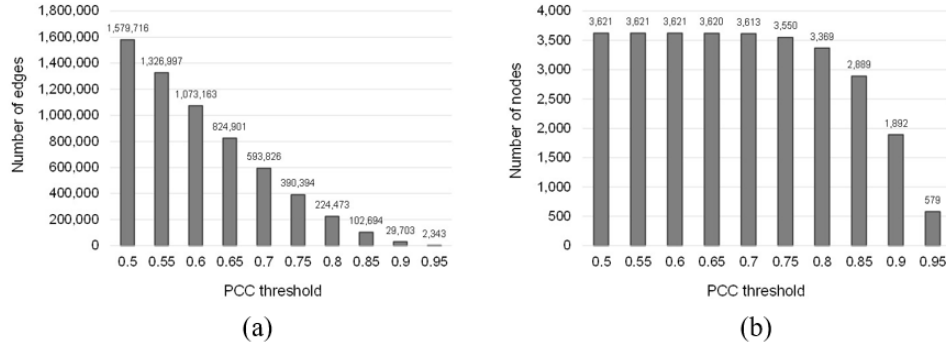


Figure 8.4: Summary of the co-expressed gene network of the periodic genes in *B. distachyon*. Distributions of number of edges (a) and nodes (b), along with the thresholds of Pearson's correlation coefficient.

list of rhythmically expressed genes will provide essential clues for understanding many circadian rhythm-regulated aspects of growth and development in the model grass. Our comprehensive list of the rhythmically expressed genes and their promoter sequences from *Brachypodium* is expected to compare *cis*-motifs with those from the other sequenced grass species across subfamilies such as rice and sorghum, which may allow us to understand diversification of GRNs involved in photoperiod responses evolved in Pooideae. Moreover, through comparative analyses with temperate grass crops such as wheat, barley, rye, and oats, functional analyses of the periodic genes in *Brachypodium* may suggest useful strategies for improving the productivity of those crops by modification of the photoperiodic response (Nakamichi, 2015).

### 8.3.3 GCN for Periodic Genes in *B. distachyon*

To build a framework of a regulatory network for the periodic genes in *B. distachyon*, we constructed a co-expression gene network based on Pearson's correlation coefficient, and represented subnetworks at each time point composed of the co-expressed genes. To construct the co-expression gene network, we calculated Pearson's correlation coefficients (PCC) for all pair-wise combinations of the periodic genes in *B. distachyon*. We found that a GCN contains 3,613 periodic genes linked by 593,826 edges with a  $PCC \geq 0.7$  for a cutoff threshold of co-expression, and that the number of genes and edges are reduced to 3,369 and 224,473, respectively, with a  $PCC \geq 0.8$  as its threshold (Fig.8.4). With a color gradient corresponding to the peak time of each gene estimated

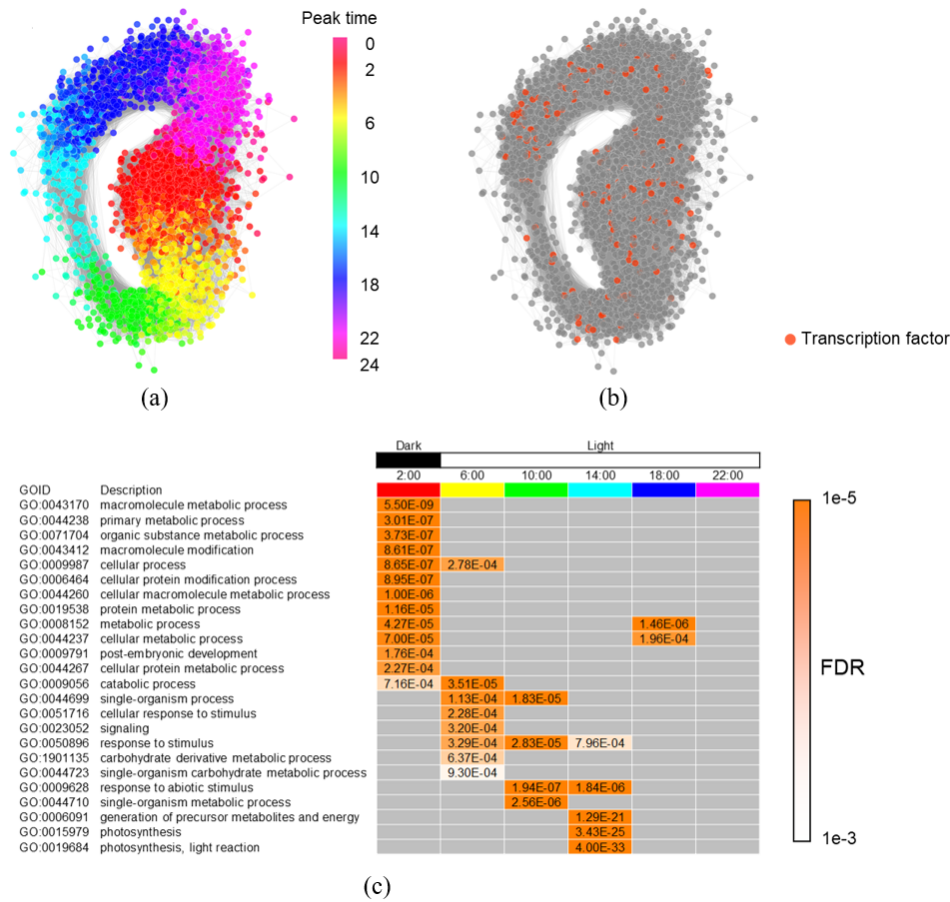


Figure 8.5: Co-expressed gene network of the diurnal transcriptome in *B. distachyon*. (a) A circular co-expressed gene network with nodes of the rhythmic genes colored according to their peak time in their diurnal expressions. (b) Genes encoding putative transcription factors found among the rhythmically expressed genes in *B. distachyon*. (c) Enriched biological functions in each time period based on the process ontology.

by our wavelet analysis, we demonstrated that the periodic gene network represents a circular form that is composed of chronologically connected genes (Fig.8.5(a)). The result is consistent with the diagonal stripe expression patterns (Fig.8.2(c)). Periodic gene expression is usually regulated by periodically expressed transcription factors. In our list of *Brachypodium* rhythmic genes, we found 192 genes that putatively encode transcription factors (Fig.8.5(b)). Some of the clock genes encode transcription factors that regulate the periodicity of the expression of many downstream genes in plants by binding to particular *cis*-elements (De Caluwe et al., 2016; Romanowski and Yanovsky, 2015). The comprehensive identification of genes co-expressed with those encoding transcription factors at each peak time, as well as *cis*-motifs in the promoters of rhythmic genes will enable us

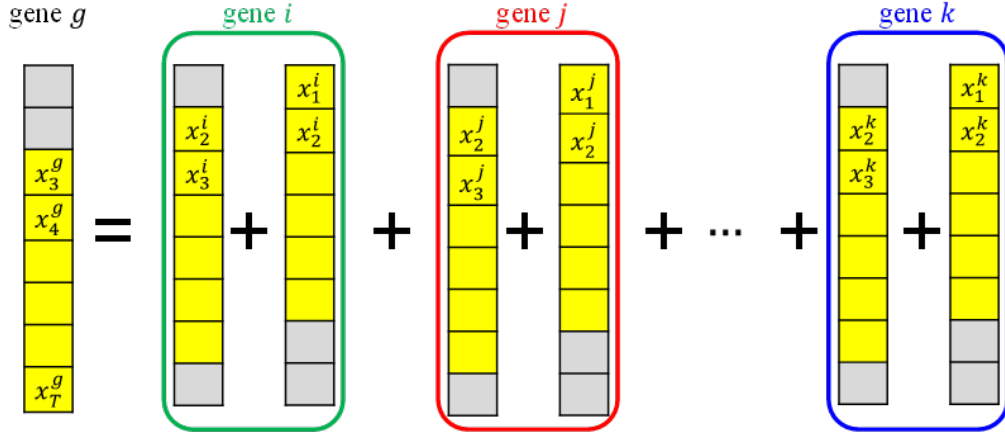


Figure 8.6: Schematic representation of an ARX model explaining relations of gene  $g$  with all the other genes. The colored circles represent grouped variables through  $t - 1$  and  $t - 2$  in the case of ARX(2).

to infer period-specific transcriptional regulatory networks, which coordinate to particular events in each period. For summarizing gene functions overrepresented in each period, we examined enriched functions of genes based on their peak time. We analyzed peak-time distribution of each of the rhythmically expressed genes and found that 36% of the genes peak at night. Thereafter, assessment of the biological functions of the periodic genes enriched at each peak time revealed that genes putatively involved in various cellular processes, primarily including metabolic processes, are enriched under dark conditions. Moreover, in addition to the processes of genes involved in photosynthesis, those related to responses to stimuli were enriched under daylight conditions, suggesting temporal compartmentalization of biological processes through diurnal changes and providing a landscape of biological events associated with circadian control in the *Brachypodium* transcriptome (Fig.8.5(c)).

### 8.3.4 GRN Inference Based on Group SCAD Analysis

To infer possible dependencies between the rhythmically expressed genes, we applied the ARX( $p$ ) model with a sparse regularization by group SCAD to the time-series expression dataset of the periodic genes. Here,  $p$  denotes a time order, i.e., the target variable at time  $t$  is regressed by the target values observed at  $t - 1, \dots, t - p$  as well as exogenous variables observed at the same time points. As schematically shown in Fig.8.6, the ARX( $p$ ) model expresses the univariate time-series

| Past time periods<br>to estimate gene<br>expression at $t$ | ARX(1) | ARX(2)     | ARX(3)          | Total  |
|------------------------------------------------------------|--------|------------|-----------------|--------|
|                                                            | $t-1$  | $t-1, t-2$ | $t-1, t-2, t-3$ |        |
| Number of edges<br>estimated                               | 2,151  | 811        | 145             | 3,107  |
| Number of genes<br>of parent node                          | 20     | 59         | 4               | 76*    |
| Number of genes<br>of child node                           | 1,888  | 634        | 137             | 2,171* |

\* The numbers are non-redundant numbers of genes identified across three models.

(a)

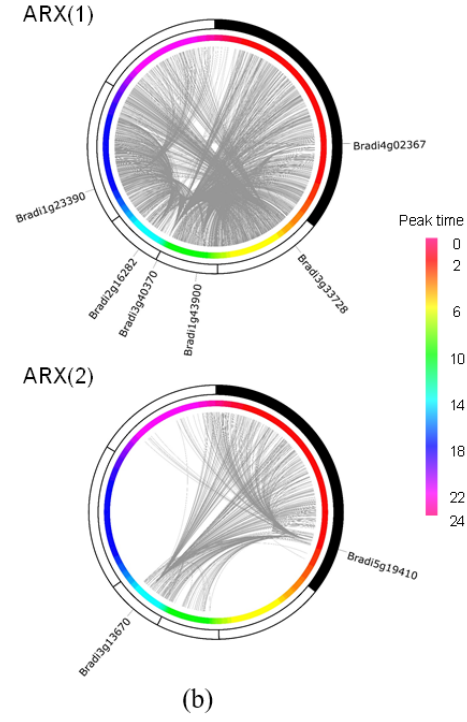


Figure 8.7: A gene regulatory network inferred by the group SCAD method in *B. distachyon*. (a) Summarized network structure of the group SCAD networks based on the ARX(1), ARX(2), and ARX(3) models. (b) Circular representations of the group SCAD networks with the rhythmic genes and their expression peak time from ARX(1) and ARX(2) models. The nodes of the rhythmic genes are sorted based on their peak time of gene expression. The gray lines represent edges of the networks. Identifiers of genes with more than 50 output edges are shown.

expression of gene  $g$  based on a combination of past observations of expressions of genes that are selected from all the rhythmic genes (gene  $i$ , gene  $j$ ,  $\dots$ , gene  $k$ ), including gene  $g$  itself, through a variable selection process. We considered three models, ARX(1), ARX(2), and ARX(3), to examine the expression of gene  $g$  at time  $t$  from the expression patterns of other genes with time lags 1, 2, and 3, respectively. We randomly produced 30 data matrices with our randomization procedure described in Appendices 1 and 2, and estimated regulatory relations for each of the data matrices. Through our multiple network estimation analysis, we quantitatively assessed regulatory relations for all pair-wise combinations, “gene  $i \rightarrow$  gene  $j$ ”, to infer the candidate directed edges from an adjacency matrix  $B$  based on  $B^m$  ( $m = 1, \dots, 30$ ). From the result, we inferred the possible regulatory interactions between the genes, which exceed the confidence threshold set to 20.

With the inferred network of the *Brachypodium* rhythmic genes using group SCAD (i.e., the group SCAD network), we figured a regulatory network of its leaf transcriptome. This network represented a typical sparse structure of a transcriptional regulatory network, with a small number of parental nodes and a large number of child nodes (Fig.8.7(a)), consistent with the findings in several other organisms (Blais and Dynlacht, 2005; Bock et al., 2012; Liu, 2015). We figured a network that contains 3,107 directed edges from 762,171 genes; in total, 2,187 genes (i.e., 60% of the rhythmically expressed genes). To illustrate the distribution of each of the parent and child nodes along with the time periods of their diurnal expression, we superimposed the group SCAD network onto the estimated peak-time of their expression, and found that several genes of the parent nodes have a remarkable number of edges in particular time periods (Fig.8.7(b)). The networks inferred by the ARX(1) model were composed of 20 parent nodes and 1,888 child nodes, suggesting that 60% of the expression patterns of the rhythmically expressed genes are explained by the rhythmical expression of 20 genes 4 h earlier. Specifically, each of the six genes of the parent nodes in the ARX(1) model was linked to more than 50 child nodes, suggesting that these genes could show representative gene expression patterns, which might be spatiotemporal decisive factors for the expression patterns of many genes 4 h later in the *Brachypodium* diurnal transcriptome. Interestingly, we found that the network estimated by the ARX(2) model showed the highest number of genes in parent nodes, indicating that a significant number of possible transcriptional relations exist from 8 h earlier. However, the ARX(3) model showed significantly reduced numbers of such relations between genes. Specifically, we found that Bradi3g13670 and Bradi5g19410 were linked to 220 and 254 child genes, respectively, in the ARX(2) model. The deduced protein sequence of Bradi5g19410 encodes a putative type-2C protein phosphatase (PP2C), which is known to negatively regulate ABA responses and MAPK cascade pathways, thereby playing important roles in stress signal transduction in plants (Hirayama and Umezawa, 2010; Umezawa et al., 2010). A study of genome-wide identification and evolutionary analyses of a putative PP2C gene family recently reported that 86 PP2C genes are present in *B. distachyon* (Cao et al., 2016). Although there is no evidence associated with molecular functions of Bradi5g19410, its rhythmic expression patterns in the diurnal transcriptome and possible spatiotemporal regulatory relations with a number of genes suggest that its crucial roles in the diurnal transcriptome might be related to the circadian gating of physiological responses in *Brachypodium*. The deduced protein sequence

of Bradi3g13670 encodes putative beta-carotene isomerase D27, which participates in a pathway leading to the biosynthesis of strigolactones (SLs) (Alder et al., 2012) and functions as endogenous and exogenous signaling molecules in response to various environmental cues (Brewer et al., 2013; Pandey et al., 2016). In addition, previous studies on FAR-RED ELONGATED HYPOCOTYL3 (FHY3), a negative regulator of RBOH genes, demonstrated that SL signaling is associated with responses to reactive oxygen species, which is mediated by phytochrome A signaling and the circadian clock, involved in the far-red light response (Lin et al., 2007). Therefore, phytochrome A signaling homeostasis controlled by the orthologous genes should presumably be involved in transcriptional regulation in *Brachypodium* and regulate a number of subordinate genes in its diurnal growth.

## 8.4 Summary

The present study demonstrates the gene networks in the model grass *B. distachyon* on the basis of diurnal transcriptome analyses. Two different approaches, co-expressed gene network and group SCAD network, enabled us to infer the transcriptomic relations of the rhythmic genes. The findings suggested that the combinatorial use of these methods should facilitate the understanding of gene networks that are hierarchically and spatiotemporally organized. Because hierarchical organization and spatiotemporal continuity are the essential features of biological molecular networks, our approach will provide a promising analytical strategy to elucidate a regulatory gene network. Moreover, it will be useful for exploring the candidates of key genes that might influence a number of subordinate genes, according to a series of chronological transcriptome datasets from laboratory and field conditions. These studies are necessary for modeling crop growth associated with agronomic traits in crops.

## Appendix 1: Details of ARX Modeling with Group SCAD

The GRN estimation described here used autoregressive with exogenous variables (ARX) models along with a sparse estimation approach based on group smoothly clipped absolute deviation (SCAD). This appendix describes the details of this modeling.

Let  $x_t^g$  ( $g \in \{1, \dots, N\}, t \in \{1, \dots, T\}$ ) denote the gene expression value of gene  $g$  at time  $t$ , where  $N$  and  $T$  represent the numbers of genes and observation times, respectively. Suppose that the expression time-series of  $g$  is represented by the following ARX( $p$ ) model:

$$x_t^g = \sum_{i=1}^N \sum_{j=1}^p \beta_j^i x_{t-j}^i + \varepsilon_t \quad (t \in \{p+1, \dots, T\}), \quad (8.1)$$

where the noise  $\varepsilon_t$  follows a normal distribution  $N(0, \sigma^2)$ . Each coefficient  $\beta_j^i$  represents the interaction from gene  $i$  to gene  $g$  at a time delay  $j$  ( $\in \{1, \dots, p\}$ ), i.e.,  $\beta_j^{i \rightarrow g}$ . For the sake of simplicity, we omit the notation  $g$  in the coefficient  $\beta_j^i$ . Let  $\mathbf{x}_{1:T}^g = (x_1^g, \dots, x_T^g)^T$ ,  $X_{1:T} = (\mathbf{x}_{1:T}^1, \dots, \mathbf{x}_{1:T}^N)$ , and  $\boldsymbol{\beta}_j = (\beta_j^1, \dots, \beta_j^N)^T$ . Then, the ARX( $p$ ) model in matrix form is

$$\mathbf{x}_{p+1:T}^g = \sum_{j=1}^p X_{p+1-j:T-j} \boldsymbol{\beta}_j + \boldsymbol{\varepsilon}, \quad (8.2)$$

where  $\boldsymbol{\varepsilon} \sim N(\mathbf{0}, \sigma^2 \mathbf{I}_{T-p})$ .

The weight parameter  $\boldsymbol{\beta}_j$  can be estimated by the least squares method. This work, however, adopted a sparse estimation scheme based on the SCAD penalty, because gene networks are typically sparse. The penalty is given by

$$p_\lambda(\beta) = \begin{cases} \lambda|\beta| & \text{if } |\beta| \leq \lambda, \\ -\frac{(|\beta|^2 - 2a\lambda|\beta| + \lambda^2)}{2(a-1)} & \text{if } \lambda < |\beta| \leq a\lambda, \\ \frac{(a+1)\lambda^2}{2} & \text{if } |\beta| > a\lambda, \end{cases} \quad (8.3)$$

where  $a$  and  $\lambda$  are positive constants. The penalty can make irrelevant coefficients of  $\boldsymbol{\beta}_j$  shrink to exactly zero in the training process. That is, we can identify interactive genes related to gene  $g$ . In our experiments, we fixed  $a = 3.7$  according to the literature (Fan and Li, 2001), while the tuning parameter  $\lambda$  was estimated by cross-validation.

Moreover, we incorporate group effects into the estimation. Let  $\boldsymbol{\beta}^i = (\beta_1^i, \dots, \beta_p^i)^T$  and  $\boldsymbol{\beta} = (\boldsymbol{\beta}^1, \dots, \boldsymbol{\beta}^N)^T$  (i.e.,  $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \dots, \boldsymbol{\beta}_p) : N \times p$ ). The group SCAD estimator  $\hat{\boldsymbol{\beta}}$  can be

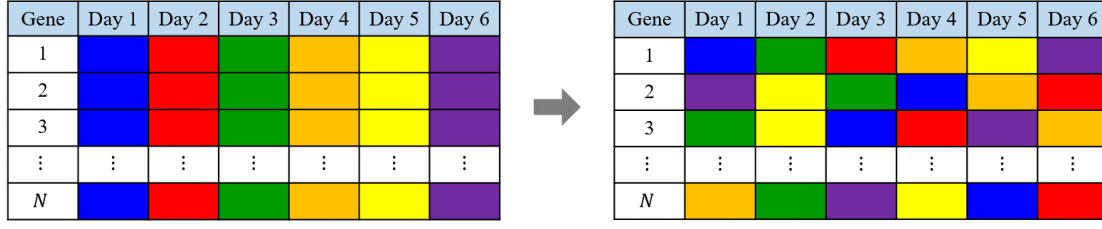


Figure 8.8: Schematic illustration of the data randomization process.

derived as

$$\hat{\beta} = \arg \min_{\beta} \left\{ \left\| \mathbf{x}_{p+1:T}^g - \sum_{j=1}^p X_{p+1-j:T-j} \beta_j \right\|_2^2 + \sum_{i=1}^N p_{\lambda}(\|\beta^i\|_2) \right\}, \quad (8.4)$$

where  $\|\cdot\|_2$  denotes the  $l_2$  norm of a vector. When  $\beta^i \neq \mathbf{0}$ , all the elements of  $\beta^i$  are nonzero because of the group effect, and we conclude that gene  $i$  regulates gene  $g$ . When  $\beta^i = \mathbf{0}$ , on the other hand, the corresponding edge is not drawn in the network. By applying the estimator to every gene, we obtain an  $N \times N$  adjacency matrix  $B$  as the output, where an element  $B_{ij}$  is 1 only if gene  $j$  regulates gene  $i$  and 0 otherwise.

## Appendix 2: Randomization

To acquire reliable outcomes as consistently as possible, multiple datasets were artificially created by random permutation, and networks were then estimated on each of the randomized datasets. The data randomization process was as follows.

The analysis included three replicates of gene expression time-series data, which were given as matrices of size  $N \times 12$ . The data were obtained over two consecutive days at 4-h intervals. Combining the three replicates produced an  $N \times 36$  data matrix, which could be virtually viewed as observations over six consecutive days. The six-day observations were then permuted for individual genes while maintaining time order within each day. Figure 8.8 outlines this process, which resulted in multiple data matrices.

With 30 replicates generated through this randomization process, the network estimation was applied to each of them, yielding 30 adjacency matrices  $B^1, \dots, B^{30}$ . The final network was derived by extracting gene interactions satisfying  $B_{ij} \geq 20$ , where  $B = \sum_{i=1}^{30} B^i$ .

# Bibliography

- Adams, S., and Carre, I. A. Downstream of the plant circadian clock: output pathways for the control of physiology and development. *Essays Biochem.* 49(1): 53-69, 2011.
- Akman, O. E., Watterson, S., Parton, A., Binns, N., Millar, A. J., and Ghazal, P., Digital clocks: simple Boolean models can quantitatively describe circadian systems. *J. R. Soc. Interface* 9(74): 2365-2382, 2012.
- Alder, A., Jamil, M., Marzorati, M., Bruno, M., Vermathen, M., Bigler, P., et al., The path from  $\beta$ -carotene to carlactone, a strigolactone-like plant hormone. *Science* 335(6074): 1348-1351, 2012.
- An, T., Cai, Y., Zhao, S., Zhou, J., Song, B., Bux, H., et al., Brachypodium distachyon T-DNA insertion lines: a model pathosystem to study nonhost resistance to wheat stripe rust. *Sci. Rep.* 6: 255, 2016.
- Araujo, K., Mahajan, D., Kerr, R., and Silva, M., Global biofuels at the crossroads: An overview of technical, policy, and investment complexities in the sustainability of biofuel Development, *Agriculture* 7(4): 32, 2017.
- Barabasi, A. L., and Oltvai, Z. N., Network biology: Understanding the cell's functional organization. *Nat. Rev. Genet.* 5: 101-113, 2004.
- Blais, A., and Dynlacht, B. D., Constructing transcriptional regulatory networks. *Genes Dev.* 19: 1499-1511, 2005.
- Bock, M., Ogishima, S., Tanaka, H., Kramer, S., and Kaderali, L., Hub-centered gene network reconstruction using automatic relevance determination. *PLoS One* 7(5): e35077, 2012.
- Boeva, V., Analysis of genomic sequence motifs for deciphering transcription factor binding and transcriptional regulation in Eukaryotic cells. *Front. Genet.* 7: 24, 2016.
- Bolger, A. M., Lohse, M., and Usadel, B., Trimmomatic: A flexible trimmer for Illumina sequence data. *Bioinformatics* 30(15): 2114-2120, 2014.
- Braun, P., Aubourg, S., Van Leene, J., De Jaeger, G., and Lurin, C., Plant protein interactomes. *Annu. Rev. Plant Biol.* 64: 161-187, 2013.
- Bredow, M., Vanderbeld, B., and Walker, V. K., Knockdown of ice-binding proteins in Brachypodium distachyon demonstrates their role in freeze protection. *PLoS One* 11: e0167941, 2016.

- Brewer, P. B., Koltai, H., and Beveridge, C. A., Diverse roles of strigolactones in plant development. *Mol. Plant* 6(1): 1828, 2013.
- Brkljacic, J., Grotewold, E., Scholl, R., Mockler, T., Garvin, D. F., Vain, P., et al., Brachypodium as a model for the grasses: today and the future. *Plant Physiol.* 157(1): 3-13, 2011.
- Cao, J., Jiang, M., Li, P., and Chu, Z., Genome-wide identification and evolutionary analyses of the PP2C gene family with their expression profiling in response to multiple stresses in *Brachypodium distachyon*. *BMC Genomics* 17: 175, 2016.
- Covington, M. F., Maloof, J. N., Straume, M., Kay, S. A., and Harmer, S. L., Global transcriptome analysis reveals circadian regulation of key pathways in plant growth and development. *Genome Biol.* 9: R130, 2008.
- Davidson, E., and Levin, M., Gene regulatory networks. *Proc. Natl. Acad. Sci. U. S. A.* 102(14): 4935, 2005.
- De Caluwe, J., Xiao, Q., Hermans, C., Verbruggen, N., Leloup, J.-C., and Gonze, D., A compact model for the complex plant circadian clock. *Front. Plant Sci.* 7: 74, 2016.
- Emmert-Streib, F., Dehmer, M., and Haibe-Kains, B., Gene regulatory networks and their applications: understanding biological and medical problems in terms of networks. *Front. Cell Dev. Biol.* 2: 38, 2014a.
- Emmert-Streib, F., Simoes, R. de M., Mullan, P., Haibe-Kains, B., and Dehmer, M., The gene regulatory network for breast cancer: Integrated regulatory landscape of cancer hallmarks. *Front. Genet.* 5: 15, 2014b.
- Erwin, D. H., and Davidson, E. H., The evolution of hierarchical gene regulatory networks. *Nat. Rev. Genet.* 10: 141-148, 2009.
- Espinosa-Soto, C., Padilla-Longoria, P., and Alvarez-Buylla, E. R., A gene regulatory network model for cell-fate determination during *Arabidopsis thaliana* flower development that is robust and recovers experimental gene expression profiles. *The Plant Cell* 16(11): 2923-2939, 2004.
- Fan, J., and Li, R., Variable selection via nonconcave penalized likelihood and its oracle properties. *J. Am. Stat. Assoc.* 96(456): 1348-1360, 2001.
- Fujita, A., Sato, J. R., Garay-Malpartida, H. M., Yamaguchi, R., Miyano, S., Sogayar, M. C., et al., Modeling gene expression regulatory networks with the sparse vector autoregressive model. *BMC Syst. Biol.* 1: 39, 2007.
- Fukushima, A., Kusano, M., Redestig, H., Arita, M., and Saito, K., Integrated omics approaches in plant systems biology. *Curr. Opin. Chem. Biol.* 13(5-6): 532-538, 2009.
- Grundy, J., Stoker, C., and Carr, I. A., Circadian regulation of abiotic stress tolerance in plants. *Front. Plant Sci.* 6: 648, 2015.
- Havens, K. A., Guseman, J. M., Jang, S. S., Pierre-Jerome, E., Bolten, N., Klavins, E., et al., A synthetic approach reveals extensive tunability of auxin signaling. *Plant Physiol* 160(1): 135-142, 2012.

- Higashi, T., Tanigaki, Y., Takayama, K., Nagano, A. J., Honjo, M. N., and Fukuda, H., Detection of diurnal variation of tomato transcriptome through the molecular timetable method in a sunlight-type plant factory. *Front. Plant Sci.* 7: 87, 2016.
- Hirayama, T., and Shinozaki, K., Research on plant abiotic stress responses in the post-genome era: Past, present and future. *Plant J.* 61(6): 1041-1052, 2010.
- Hirayama, T., and Umezawa, T., The PP2C-SnRK2 complex *The. Plant Signal. Behav.* 5(2): 160-163, 2010.
- Jaeger, K. E., Pullen, N., Lamzin, S., Morris, R. J., and Wigge, P. A., Interlocking feedback loops govern the dynamic behavior of the floral transition in Arabidopsis. *Plant Cell* 25(3): 820-833, 2013.
- Karlebach, G., and Shamir, R., Modelling and analysis of gene regulatory networks. *Nat. Rev. Mol. Cell. Biol.* 9(10): 770-780, 2008.
- Kellogg, E. A., Brachypodium distachyon as a genetic model system. *Annu. Rev. Genet.* 49: 1-20, 2015.
- Koda S., Onda Y., Matsui H., Takahagi K., Uehara-Yamaguchi Y., Shimizu M., Inoue K., Yoshida T., Sakurai T., Honda H., Eguchi S., Nishii R., and Mochida K., Diurnal transcriptome and gene network represented through sparse modeling in *Brachypodium distachyon*. *Frontiers in Plant Science* 8, article no. 2055, 2017.
- Krouk, G., Lingeman, J., Colon, A. M., Coruzzi, G., and Shasha, D., Gene regulatory networks in plants: learning causality from time and perturbation. *Genome Biol.* 14: 123, 2013.
- La Rota, C., Chopard, J., Das, P., Paindavoine, S., Rozier, F., Farcot, E., et al., A data-driven integrative model of sepal primordium polarity in Arabidopsis. *Plant Cell* 23(12): 4318-4333, 2011.
- Lachmann, A., Giorgi, F. M., Lopez, G., and Califano, A., ARACNe-AP: Gene network reverse engineering through adaptive partitioning inference of mutual information. *Bioinformatics* 32(14): 2233-2235, 2016.
- Lebre, S., Inferring dynamic genetic networks with low order independencies. *Stat. Appl. Genet. Mol. Biol.* 8(1): 1-38, 2009.
- Lee, W. P., and Tzou, W. S., Computational methods for discovering gene networks from expression data. *Brief. Bioinform.* 10(4): 408-423, 2009.
- Lin, R., Ding, L., Casola, C., Ripoll, D. R., Feschotte, C., and Wang, H., Transposase-derived transcription factors regulate light signaling in Arabidopsis. *Science* 318(5854): 1302-1305, 2007.
- Liu, Z.-P., Reverse engineering of genome-wide gene regulatory networks from gene expression data. *Curr. Genomics* 16(1): 3-22, 2015.
- Lopez-Maury, L., Marguerat, S., and Bahler, J., Tuning gene expression to changing environments: from rapid responses to evolutionary adaptation. *Nat. Rev. Genet.* 9(8): 583-593, 2009.

- Love, M. I., Anders, S., and Huber, W., Differential analysis of count data - the DESeq2 package. doi:110.1186/s13059-014-0550-8, 2014.
- Matsuzaki, J., Kawahara, Y., and Izawa, T., Punctual transcriptional regulation by the rice circadian clock under fluctuating field conditions. *Plant Cell* 27(3): 633-648, 2015.
- Michailidis, G., and D'Alche-Buc, F., Autoregressive models for gene regulatory network inference: Sparsity, stability and causality issues. *Math. Biosci.* 246(2): 326-334, 2013.
- Mochida, K., and Shinozaki, K., Unlocking triticeae genomics to sustainably feed the future. *Plant Cell Physiol.* 54(12): 1931-1950, 2013.
- Moreno-Risueno, M. A., Busch, W., Benfey, P. N., Cook, D. R., and Varshney, R. K., Omics meet networks — using systems approaches to infer regulatory networks in plants. *Curr. Opin. Plant Biol.* 13(2): 126-131, 2009.
- Nakamichi, N., Adaptation to the local environment by modifications of the photoperiod response in crops. *Plant Cell Physiol.* 56(4): 594-604, 2015.
- Pandey, A., Sharma, M., and Pandey, G. K., Emerging roles of strigolactones in plant responses to stress and development. *Front. Plant Sci.* 7: 434, 2016.
- Pattanaik, S., Patra, B., Singh, S. K., and Yuan, L., An overview of the gene regulatory network controlling trichome development in the model plant, Arabidopsis. *Front Plant Sci.* 5: 259, 2014.
- Pokhilko, A., Fernandez, A. P., Edwards, K. D., Southern, M. M., Halliday, K. J., and Millar, A. J., The clock gene circuit in Arabidopsis includes a repressilator with additional feedback loops. *Mol. Syst. Biol.* 8: 574, 2012.
- Provart, N., Correlation networks visualization. *Front. Plant Sci.* 3: 240, 2012.
- Rancour, D. M., Hatfield, R. D., Marita, J. M., Rohr, N. A., and Schmitz, R. J., Cell wall composition and digestibility alterations in *Brachypodium distachyon* achieved through reduced expression of the UDP-arabinopyranose mutase. *Front. Plant Sci.* 6: 446, 2015.
- Reece-Hoyes, J. S., Diallo, A., Lajoie, B., Kent, A., Shrestha, S., Kadreppa, S., et al., Enhanced yeast one-hybrid assays for high-throughput gene-centered regulatory network mapping. *Nat. Methods* 8(12): 1059-1064, 2011.
- Romanowski, A., and Yanovsky, M. J., Circadian rhythms and post-transcriptional regulation in higher plants. *Front. Plant Sci.* 6: 437, 2015.
- Salazar, J. D., Saithong, T., Brown, P. E., Foreman, J., Locke, J. C. W., Halliday, K. J., et al., Prediction of photoperiodic regulators from quantitative gene circuit models. *Cell* 139(6): 1170-1179, 2009.
- Sankar, M., Osmont, K. S., Rolcik, J., Gujas, B., Tarkowska, D., Strnad, M., et al., A qualitative continuous model of cellular auxin and brassinosteroid signaling and their crosstalk. *Bioinformatics* 27(10): 1404-1412, 2011.

- Scheres, B., and van der Putten, W. H., The plant perceptron connects environment to development. *Nature* 543(7645): 337-345, 2017.
- Shimamura, T., Imoto, S., Yamaguchi, R., Fujita, A., Nagasaki, M., and Miyano, S., Recursive regularization for inferring gene networks from time-course gene expression profiles. *BMC Syst. Biol.* 3: 41, 2009.
- Shojaie, A., and Michailidis, G., Discovering graphical Granger causality using the truncating lasso penalty. *Bioinformatics* 26(18): i517-i523, 2011.
- Sibout, R., Proost, S., Hansen, B. O., Vaid, N., Giorgi, F. M., Ho-Yue-Kuang, S., et al., Expression atlas and comparative coexpression network analyses reveal important genes involved in the formation of lignified cell wall in *Brachypodium distachyon*. *New Phytol.* 215(3): 1009-1025, 2017.
- Steuer, R., Kurths, J., Daub, C. O., Weise, J., and Selbig, J., The mutual information: Detecting and evaluating dependencies between variables. *Bioinformatics* 18 Suppl 2: S231-S240, 2002.
- Thompson, D., Regev, A., and Roy, S., Comparative analysis of gene regulatory networks: from network reconstruction to evolution. *Annu. Rev. Cell Dev. Biol.* 31: 399-428, 2015.
- Tohge, T., and Fernie, A. R., Co-expression and co-responses: within and beyond transcription. *Front. Plant Sci.* 3: 248, 2012.
- Tzfadia, O., Diels, T., De Meyer, S., Vandepoele, K., Aharoni, A., and Van de Peer, Y., CoExpNetViz: Comparative co-expression networks construction and visualization tool. *Front. Plant Sci.* 6: 1194, 2016.
- Umezawa, T., Nakashima, K., Miyakawa, T., Kuromori, T., Tanokura, M., Shinozaki, K., et al., Molecular basis of the core regulatory network in ABA responses: Sensing, signaling and transport. *Plant Cell Physiol.* 51(11): 1821-1839, 2010.
- Urano, K., Kurihara, Y., Seki, M., and Shinozaki, K., "Omics" analyses of regulatory networks in plant abiotic stress responses. *Curr. Opin. Plant Biol.* 13(2): 132-138, 2010.
- van Dam, S., Vosa, U., van der Graaf, A., Franke, L., and de Magalhães, J. P., Gene co-expression analysis for functional classification and gene-disease predictions. *Brief. Bioinform.* pii: bbw139, 2017.
- Vernoux, T., Brunoud, G., Farcot, E., Morin, V., Van den Daele, H., Legrand, J., et al., The auxin signalling network translates dynamic input into robust patterning at the shoot apex. *Mol. Syst. Biol.* 7: 508, 2014.
- Vogel, J., and Bragg, J., *Brachypodium distachyon*, a new model for the Triticeae," in *Genetics and Genomics of the Triticeae* (New York, NY: Springer US), pp. 427-449, 2009.
- Vogel, J. P., *Genetics and Genomics of Brachypodium*, doi:10.1007/978-3-319-26944-3, 2016.
- Wang, L., Chen, G., and Li, H., Group SCAD regression analysis for microarray time course gene expression data. *Bioinformatics* 23(12): 1486-1494, 2007.

Wu, X., Li, P., Wang, N., Gong, P., Perkins, E. J., Deng, Y., et al., State space model with hidden variables for reconstruction of gene regulatory networks. *BMC Syst. Biol.* 5(Suppl 3): S3, 2011.