

Pattern Detection And Recognition Using Over-Complete And Sparse  
Representations

Wumo Pan

A Thesis  
in  
The Department  
of  
Computer Science and Software Engineering

Presented in Partial Fullfillment of the Requirements  
for the Degree of Doctor of Philosophy (Computer Science) at  
Concordia University  
Montreal, Quebec, Canada

February, 2009

© Wumo Pan, 2009



Library and Archives  
Canada

Published Heritage  
Branch

395 Wellington Street  
Ottawa ON K1A 0N4  
Canada

Bibliothèque et  
Archives Canada

Direction du  
Patrimoine de l'édition

395, rue Wellington  
Ottawa ON K1A 0N4  
Canada

*Your file* *Votre référence*  
*ISBN:* 978-0-494-63351-9  
*Our file* *Notre référence*  
*ISBN:* 978-0-494-63351-9

#### NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

#### AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protègent cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

---

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

■ ■ ■  
**Canada**

# **ABSTRACT**

## **Pattern Detection And Recognition Using Over-Complete And Sparse Representations**

Wumo Pan, Ph.D.

Concordia University, 2009

Recent research in harmonic analysis and mammalian vision systems has revealed that over-complete and sparse representations play an important role in visual information processing. The research on applying such representations to pattern recognition and detection problems has become an interesting field of study. The main contribution of this thesis is to propose two feature extraction strategies - the global strategy and the local strategy - to make use of these representations.

In the global strategy, over-complete and sparse transformations are applied to the input pattern as a whole and features are extracted in the transformed domain. This strategy has been applied to the problems of rotation invariant texture classification and script identification, using the Ridgelet transform. Experimental results have shown that better performance has been achieved when compared with Gabor multi-channel filtering method and Wavelet based methods.

The local strategy is divided into two stages. The first one is to analyze the local over-complete and sparse structure, where the input 2-D patterns are divided into patches and the local over-complete and sparse structure is learned from these patches using sparse approximation techniques.

The second stage concerns the application of the local over-complete and sparse structure. For an object detection problem, we propose a *sparsity testing* technique,

where a local over-complete and sparse structure is built to give sparse representations to the text patterns and non-sparse representations to other patterns. Object detection is achieved by identifying patterns that can be sparsely represented by the learned structure. This technique has been applied to detect texts in scene images with a recall rate of 75.23% (about 6% improvement compared with other works) and a precision rate of 67.64% (about 12% improvement).

For applications like character or shape recognition, the learned over-complete and sparse structure is combined with a Convolutional Neural Network (CNN). A second text detection method is proposed based on such a combination to further improve (about 11% higher compared with our first method based on sparsity testing) the accuracy of text detection in scene images. Finally, this method has been applied to handwritten Farsi numeral recognition, which has obtained a 99.22% recognition rate on the CENPARMI Database and a 99.5% recognition rate on the HODA Database. Meanwhile, a SVM with gradient features achieves recognition rates of 98.98% and 99.22% on these databases respectively.



## ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Dr. Ching Y. Suen and Dr. Tien D. Bui. Both supervisors gave their support to my research without reservation. I have benefited a lot from their expertise, vision, ideas and personalities. Dr. Suen gave me unprecedented freedom in pursuing problems that would trigger my interests. His liberal supervision and emphasis on clarity were fundamental in shaping the research effort that finally led to this thesis. Dr. Bui provided me with many inspiring comments and heartfelt encouragement. Our discussions about Wavelet transforms and their variants laid the foundation of the methods applied in this thesis. The support from both professors went beyond pure academic supervision. They were generous in providing financial support. They made sure that I had every possible opportunity to improve my capabilities in different aspects. Without their support, this thesis would not have been possible.

My grateful thanks go to other members in the Center for Pattern Recognition and Machine Intelligence (CENPARMI). I enjoyed stimulating discussions and friendships with Yun Li, Chunlei He, Wu Ding, Xiaoxiao Niu and Ying Li. I also appreciate the help from previous CENPARMI students, including Dr. Qizhi Xu, Dr. Jianxiong Dong, Dr. Ping Zhang, Nenghong Fu and Yan Zhang.

Special thanks go to our Lab administrator, Nicola Nobile, for his excellent technical support and to our secretary, Marleah Blom, who cheers up every student in CENPARMI. I also wish to thank Shira Katz for her extreme patience in proofreading my thesis.

Finally, it is beyond words to express my appreciation to the support and happiness brought to me by my wife, Bo, my son, Winston, and my parents.

# Table of Contents

|  |           |
|--|-----------|
| <b>List of Figures</b>   | <b>x</b>  |
| <b>List of Tables</b>  | <b>xv</b> |
| <b>1 Introduction</b>  | <b>1</b>  |
| 1.1 Motivation . . . . .   | 1         |
| 1.2 Problem Statement . . . . .                                      | 8         |
| 1.3 Thesis Outline . . . . .   | 10        |
| <b>2 Background Material</b>   | <b>12</b> |
| 2.1 Continuous Ridgelet Transform . . . . .                          | 12        |
| 2.2 Sparse Approximation And Over-Complete Dictionary Learning . . . | 14        |
| 2.2.1 Notations . . . . .  | 14        |
| 2.2.2 Over-complete Dictionary Learning with K-SVD Algorithm . .     | 15        |
| 2.2.2.1 Sparse Coding . . . . .                                      | 16        |
| 2.2.2.2 Dictionary Updating . . . . .                                | 20        |
| 2.3 Convolutional Neural Network (CNN) . . . . .                     | 21        |
| <b>3 Text Detection from Scene Images Using Sparsity Testing</b>     | <b>23</b> |
| 3.1 Introduction . . . . .   | 23        |
| 3.2 Related Works . . . . .  | 25        |
| 3.3 Text Detection via Sparsity Testing . . . . .                    | 27        |
| 3.3.1 Learning Over-Complete Dictionary for Text . . . . .           | 28        |

|          |   |           |
|----------|---|-----------|
| 3.3.1.1  | Learning Procedure . . . . .  | 28        |
| 3.3.2    | Text Detection Algorithm . . . . .  | 31        |
| 3.3.2.1  | Preprocessing . . . . .   | 31        |
| 3.3.2.2  | Connected Components Labeling . . . . .   | 31        |
| 3.3.2.3  | Layout Analysis . . . . .   | 32        |
| 3.3.2.4  | Multi-scale Processing . . . . .  | 33        |
| 3.3.3    | Experimental Results . . . . .  | 33        |
| 3.4      | Conclusion . . . . .  | 37        |
| <b>4</b> | <b>Text Detection from Natural Scene Images Using Topographic Maps<br/>And Sparse Representations</b> | <b>38</b> |
| 4.1      | Introduction . . . . .  | 39        |
| 4.2      | Identification of Text Candidates Based on Topographic Maps . . . . .                                 | 40        |
| 4.2.1    | Topographic Maps - Background Information . . . . .   | 40        |
| 4.2.2    | Tree Representation of Level Lines . . . . .  | 41        |
| 4.2.3    | Identification of Text Candidates . . . . .   | 42        |
| 4.2.3.1  | Selecting $\varepsilon$ -Meaningful Boundaries . . . . .  | 43        |
| 4.2.3.2  | Adaptive Thresholding of Maximal Monotone Interval . . . . .  | 44        |
| 4.3      | Shape Analysis Using CNN And Sparse Representations . . . . .   | 47        |
| 4.3.1    | Learning Sparse Structure from the Data . . . . .   | 49        |
| 4.3.2    | The CNN Classifier . . . . .  | 50        |
| 4.4      | Text Line Growing And Text Segmentation . . . . .   | 51        |
| 4.5      | Experimental Results . . . . .  | 54        |
| 4.5.1    | Databases . . . . .   | 54        |
| 4.5.2    | Training the CNNs . . . . .   | 56        |
| 4.5.3    | Comparison between Regular CNN and Over-complete CNN . . . . .  | 57        |
| 4.5.4    | Comparison with Other Methods . . . . .   | 58        |
| 4.5.5    | Discussion . . . . .  | 59        |
| 4.6      | Conclusion . . . . .  | 61        |

|          |  |           |
|----------|--|-----------|
| <b>5</b> | <b>Isolated Handwritten Farsi Numeral Recognition Using Sparse and Over-Complete Representations</b> | <b>64</b> |
| 5.1      | Introduction . . . . .   | 64        |
| 5.2      | Related Works . . . . .  | 66        |
| 5.3      | Configuration of the CNN . . . . .   | 70        |
| 5.4      | Learning Over-Complete And Sparse Representations . . . . .  | 70        |
| 5.4.1    | Determining the $T_0$ in the K-SVD algorithm . . . . .   | 71        |
| 5.5      | Experiments . . . . .  | 73        |
| 5.5.1    | The Farsi Handwritten Numeral Databases . . . . .  | 74        |
| 5.5.2    | Preprocessing . . . . .  | 76        |
| 5.5.3    | Comparisons between CNN-Based Methods . . . . .  | 78        |
| 5.5.4    | Comparison with Other Classifiers . . . . .  | 83        |
| 5.6      | Conclusion . . . . .   | 86        |
| <b>6</b> | <b>Rotation Invariant Texture Classification by Ridgelet Transform</b>                               | <b>87</b> |
| 6.1      | Related Works . . . . .  | 88        |
| 6.1.1    | Model-Based Methods . . . . .  | 88        |
| 6.1.1.1  | Simultaneous Autoregressive (SAR)  |           |
|          | Model-Based Methods . . . . .  | 89        |
| 6.1.1.2  | Markov Model . . . . .   | 89        |
| 6.1.2    | Filtering-Based Methods . . . . .  | 90        |
| 6.1.2.1  | Gabor Multi-Channel Filtering . . . . .  | 91        |
| 6.1.2.2  | Wavelet Decompositions . . . . .   | 93        |
| 6.2      | Wavelet Selection In Ridgelet Transformation . . . . .   | 95        |
| 6.3      | Rotation Invariant Texture Feature Extraction in the Ridgelet Domain                                 | 96        |
| 6.3.1    | Calculating the Polar Fourier Transform from Radon Transform   | 98        |
| 6.3.2    | Rotation Invariant Feature Extraction . . . . .  | 98        |
| 6.3.3    | Relation of the Proposed Method to Multi-Channel Filtering .   | 100       |
| 6.4      | Experiments and Discussion . . . . .   | 101       |

|          |  |            |
|----------|--|------------|
| 6.4.1    | Experimental Results . . . . .                       | 101        |
| 6.4.2    | Discussion . . . . .                                 | 105        |
| 6.5      | Script Identification - A Case Study . . . . .       | 109        |
| 6.5.1    | Background . . . . .                                 | 109        |
| 6.5.1.1  | Local Methods . . . . .                              | 111        |
| 6.5.1.2  | Global Methods . . . . .                             | 113        |
| 6.5.2    | Experiments . . . . .                                | 114        |
| 6.5.2.1  | Data Preparation . . . . .                           | 114        |
| 6.5.2.2  | Selected Methods for Script Identification . . . . . | 115        |
| 6.5.3    | Experimental Results And Discussion . . . . .        | 117        |
| 6.6      | Conclusion . . . . .                                 | 118        |
| <b>7</b> | <b>Conclusion</b>                                    | <b>121</b> |
|          | <b>Bibliography</b>                                  | <b>124</b> |

# List of Figures

|     |  |    |
|-----|--|----|
| 1.1 | One level wavelet decomposition applied to two signals using Daubechie-6 wavelets: (a) Two input signals, one is the shifted version of another, (b) the scaling functions of the input signals, and (c) the wavelet coefficients of the input signals. . . . .  | 3  |
| 1.2 | One level un-decimated wavelet decomposition applied to two signals using the <i>algorithme à trous</i> : (a) Two input signals, one is the shifted version of another, (b) the scaling functions of the input signals, and (c) the wavelet coefficients of the input signals. . . . .   | 4  |
| 1.3 | Examples of the Curvelets. . . . .   | 5  |
| 1.4 | Sparse property of the coefficients of the wavelet, Curvelet and Contourlet transforms: (a) Amplitude distribu-tion of the wavelet coefficients in the finest horizontal sub-band, (b) amplitude distribu-tion of the Curvelet coefficients in one finest sub-band, and (c) amplitude distribu-tion of the Contourlet coefficients in one finest sub-band. . . . . | 7  |
| 1.5 | Experiment producing sparse components of the natural images in [91]. These sparse components are learned from 16 by 16 (in pixel) image patches, which are extracted from natural scenes with the goal of maximizing both preserved information and the sparsity of the expansions. . . . .   | 8  |
| 2.1 | An example of ridgelets in two orientations. The ridgelet on the right is a rotated version of the one on the left. . . . .  | 13 |

|     |  |    |
|-----|--|----|
| 2.2 | Adopted CNN architecture. . . . .  | 22 |
| 3.1 | Left: Examples of the observations taken from character edges for learning the dictionary. Right: The most frequently used atoms learned by K-SVD. . . . .   | 30 |
| 3.2 | Some examples generated by the proposed method. In each example, the top left gives the input grayscale image; the top right gives the edge map generated by the Canny operator; the bottom left gives the pixel-level labeling results; the bottom right gives the text LINEs detected after layout analysis. . . . . | 34 |
| 3.3 | (a) Recall rate of the proposed algorithm under different parameter settings. (b) Precision rate of the proposed algorithm under different parameter settings. . . . .   | 36 |
| 4.1 | Comparison between level lines and edges with two examples. . . . .  | 43 |
| 4.2 | Frequency maps of the images in Figure 4.1 shown as images. In these images, the darker the pixel is, the higher frequency it has. . . . .   | 45 |
| 4.3 | The curve that shows how the function $T$ would vary with local threshold $c_l$ . . . . .  | 47 |
| 4.4 | Detection of text candidates in some scene image samples. . . . .  | 48 |
| 4.5 | Atoms in the learned over-complete dictionary. . . . .   | 49 |
| 4.6 | Shape classification results using the proposed method. Only shapes classified as texts are shown in red. Left column: results from <i>regular</i> CNN. Right column: results from <i>over-complete</i> CNN. . . . .   | 52 |
| 4.7 | Examples of the final text detection results using the proposed method using <i>over-complete</i> CNN. . . . .   | 53 |
| 4.8 | Examples of the extracted texts using the proposed method. Left column: text line images cropped from the original images. Right column: extracted characters from their backgrounds. . . . .  | 55 |

|      |  |    |
|------|--|----|
| 4.9  | Examples of the samples used in CNN training. All samples were normalized to 47 by 47 pixels. . . . .  | 56 |
| 4.10 | Examples of failures of the proposed method. (a)~(c) show failures related to image contrast; (d)~(f) are examples of touching characters; (g) and (h) are examples of broken characters; (i) gives an example of the glare from flash reflections on a book cover; (j) demonstrates an example of character-like shapes that cause false alarms; (k) and (l) are examples of some special designs, which cannot be processed properly by the proposed method. . . . . | 62 |
| 5.1  | Numerals in Farsi that can be legally written in different shapes. . . .   | 65 |
| 5.2  | Demonstration of the feature extraction procedure in [113]: (a) Outer profiles, (b) projection histograms, and (c) crossing counts from two directions. . . . .  | 67 |
| 5.3  | Demonstration of the shadow coding feature extraction technique: (a) Mask definition, and (b) shadow codes for a sample of the Farsi numeral '8'. . . . .  | 68 |
| 5.4  | Error rate of the proposed method on the validation set of the CENPARMI Handwritten Farsi numeral database, using different values of the parameter $T_0$ . . . . .  | 72 |
| 5.5  | Over-complete and sparse dictionaries learned using different values of $T_0$ . . . . .  | 73 |
| 5.6  | Some samples in the CENPARMI Farsi handwritten numeral database.   | 75 |
| 5.7  | Some samples in the HODA Farsi handwritten numeral database. . .   | 77 |
| 5.8  | Examples of the preprocessing results from the CENPARMI Farsi handwritten numeral database: Left, original samples; Right, corresponding samples after preprocessing. . . . .  | 78 |
| 5.9  | Some distorted samples. Rows 1 and 3: samples after preprocessing. Rows 2 and 4: the same samples after distortion. . . . .  | 80 |



|      |   |     |
|------|---|-----|
| 5.10 | Misrecognized samples from the test set of the CENPARMI database<br>by the proposed method. . . . .   | 82  |
| 5.11 | Decomposition of a given gradient direction. . . . .  | 85  |
| 6.1  | A commonly adopted Gabor filter bank configuration. . . . .   | 92  |
| 6.2  | (a) Basis functions given by SVD decomposition. (b) Normalized<br>accumulative sum of the singular values. . . . .  | 93  |
| 6.3  | Plot of a B-spline wavelet in the time and frequency domain with<br>$f_c = 0.256$ and $f_b = 0.512$ radians per sample: (a) The real part of the<br>wavelet in the time domain, (b) the imaginary part of the wavelet in<br>the time domain, and (c) the Fourier spectrum of this wavelet (properly<br>scaled). . . . .   | 97  |
| 6.4  | Calculation of the polar Fourier transform using the Radon transform.   | 99  |
| 6.5  | Demonstration of the frequency-orientation decomposition achieved by<br>the proposed method. . . . .  | 101 |
| 6.6  | Demonstration of the discrimination capability of the proposed features:<br>(a) Ten texture samples from the Brodatz texture album (from left to<br>right and top to bottom: D15, D20, D23, D34, D37, D46, D57, D81,<br>D87, and D93), and (b) the clusters corresponding to the 10 textures<br>in the training set. Here, we use the two features with the highest<br>discrimination power as x and y axes. . . . .                          | 103 |
| 6.7  | The 35 Texture Samples from the VisTex Database[127]. . . . .   | 104 |
| 6.8  | Demonstration of the two issues related to histogram equalization: (a)<br>A synthetic texture generated by a sinusoid, (b) the same texture after<br>histogram equalization, (c) a horizontal scan line in texture image (a),<br>(d) a horizontal scan line in texture image (b), (e) texture D48 from<br>Brodatz texture data set, and (f) histogram equalized version of (e),<br>where background noise has been greatly amplified. . . . . | 106 |
| 6.9  | Textures with low classification rates. . . . .   | 108 |

|      |   |     |
|------|---|-----|
| 6.10 | Plots of the training and testing samples in the feature space. Each graph (a, b, c) corresponds to one feature extraction method. The features have been extracted from texture <i>Canvas023</i> in the Outex data set. For each figure, we used the two features with the highest discrimination power as x and y axes: (a) Features extracted by Method I, (b) features extracted by Method II, and (c) features extracted by the proposed method. . . . . | 110 |
| 6.11 | Image samples in different scripts with different fonts: (a) Chinese, (b) Japanese, (c) Korean, (d) Arabic, (e) English, and (f) Russian. . . . .   | 116 |

# List of Tables

|     |  |     |
|-----|--|-----|
| 3.1 | Experimental results and comparison. . . . .   | 37  |
| 4.1 | Comparison using the test database (%). . . . .  | 58  |
| 4.2 | Performance comparison using the same test database (%). . . . .   | 59  |
| 5.1 | Test error rates (%) of CNN based methods on the CENPARMI database.  | 80  |
| 5.2 | Confusion matrix of the proposed method on the CENPARMI database.  | 83  |
| 5.3 | Test error rates (%) of the classifiers investigated in this research. . .   | 85  |
| 6.1 | The Correct Classification Rate for Data Set 1 Using Different Feature<br>Extraction Methods and Different $k$ Values for $k$ -NN Classifier (%). .  | 102 |
| 6.2 | The Correct Classification Rate for Data Set 2 Using Different Feature<br>Extraction Methods and Different $k$ Values for $k$ -NN Classifier (%). .  | 103 |
| 6.3 | The Correct Classification Rate for Data Set 3 Using Different Feature<br>Extraction Methods and Different $k$ Values for $k$ -NN Classifier (%). .  | 104 |
| 6.4 | The confusion matrix of the proposed method on the Brodatz data<br>set using the 1-NN Classifier. Textures with classification rates higher<br>than 75% are not shown in this table. . . . . | 107 |
| 6.5 | Performance of the script identification algorithms on the basic database<br>(%). . . . .  | 119 |

# Chapter 1

## Introduction

In this introductory chapter, the research topic, namely, feature extraction using over-complete and sparse representations, is presented. We begin with a description of the motivation behind this research endeavor. The applications and the proposed methods are then briefly introduced. Finally, we conclude with an outline of this thesis.

### 1.1 Motivation

An important step in pattern recognition and pattern detection is to extract key information or features that highlight the difference among patterns (in pattern recognition), or the difference between the target patterns and the irrelevant backgrounds (in pattern detection). Generally speaking, extracting useful features from complex input patterns is not an easy task. A common practice then is to apply a certain transformation to a given pattern and then extract the features in the transformed domain. Among many possible transformations, linear ones have long been appealing due to their simplicity and efficiency. The most well-known linear transformations are Fourier transform and Wavelet transform.

Fourier transform treats a given signal as a sum of sine/cosine waves. This is amazing since even though the signal itself might be complex, its ingredients are

curves with very easy-to-understand properties. Furthermore, Fourier transform can be efficiently computed, due to the Fast Fourier Transform. Unfortunately, Fourier transform can only tell us which frequency components have occurred in the given signal. It can not tell us when those components start in the signal and when they stop. Therefore, Fourier transform is not a good analysis tool for time/space variant signals (such as images).

Wavelet transform is another very popular signal analysis tool. Several amazing properties of the wavelet transform have made it widely accepted in many applications [25, 79]: It is localized in both space and frequency domains and therefore very useful in analyzing time-variant signals. It naturally takes into consideration the multi-scale aspect of the signals. Furthermore, a very efficient algorithm to calculate the transformation is available.

Despite its great successes in many applications, Wavelet has three major limitations. One well-known fact is that the critically sampled Discrete Wavelet Transform (DWT) is not *shift invariant*. In other words, the DWT transform of a shifted signal is not a shifted version of the DWT transform of the original signal. This effect is shown in Figure 1.1. Two synthesized signals are given in Figure 1.1 (a), where one is the shifted version of the other. One level wavelet decomposition is applied to these signals using the Daubechie-6 wavelets. It can be observed that the wavelet coefficients (Figure 1.1 (b)) of the two signals are not shift invariant. This observation also applies to the scaling functions (Figure 1.1 (c)) of these two signals.

Another limitation of the DWT lies in its extension to higher dimensions. While efficient in representing point singularities in 1-dimension, separable DWT becomes inefficient in representing singularities in higher dimensions, e.g. lines or curves in 2-dimensions. Suppose we have an image  $f$  with discontinuity along a generic  $C^2$  smooth curve and the image is smooth elsewhere. The best  $m$ -term approximation

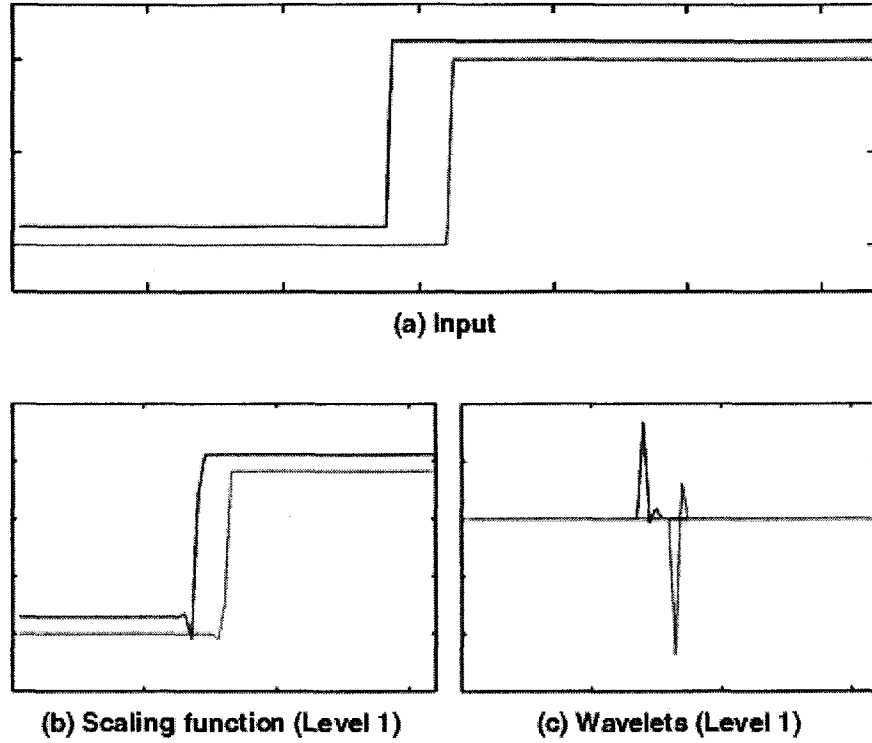


Figure 1.1: One level wavelet decomposition applied to two signals using Daubechies-6 wavelets: (a) Two input signals, one is the shifted version of another, (b) the scaling functions of the input signals, and (c) the wavelet coefficients of the input signals.

$\tilde{f}_m$  of this image would decay slowly [10]:

$$\|f - \tilde{f}_m\|_2^2 \asymp m^{-1}, m \rightarrow \infty.$$

Furthermore, separable DWT suffers from its poor directional selectivity. The usual orthogonal wavelet transforms have wavelets with only vertical, horizontal and diagonal orientations. This directional selectivity is, as we know, a very desirable property in pattern recognition applications.

To overcome these limitations of DWT, different variants of the wavelet methods have been proposed in recent years. One way of providing shift invariance is to use the un-decimated form of the dyadic filter tree, which is implemented most efficiently

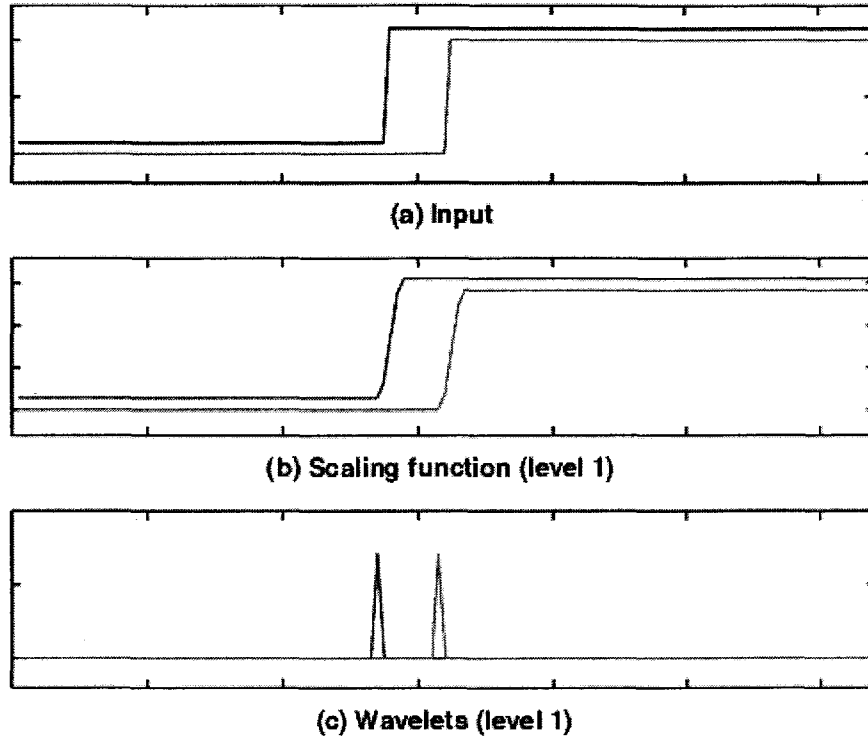


Figure 1.2: One level un-decimated wavelet decomposition applied to two signals using the *algorithm à trous*: (a) Two input signals, one is the shifted version of another, (b) the scaling functions of the input signals, and (c) the wavelet coefficients of the input signals.

by the *algorithm à trous*[82]. Results of this algorithm on the above-mentioned two signals (see 1.1 (a)) are shown in Figure 1.2. Another way to achieve shift invariance is to use the Dual-Tree Complex Wavelet Transform [61], which also provides better directional selectivity. One important point here is that both transformations are *redundant* in that the lengths of the coefficients of the transformation are larger than that of the input signals.

Extending the wavelet method to handle high-dimensional singularities and to provide sufficient directional selectivity is more involved. Many methods have been proposed, including steerable pyramids [111], bandlet [101], beamlet [34], Ridgelet

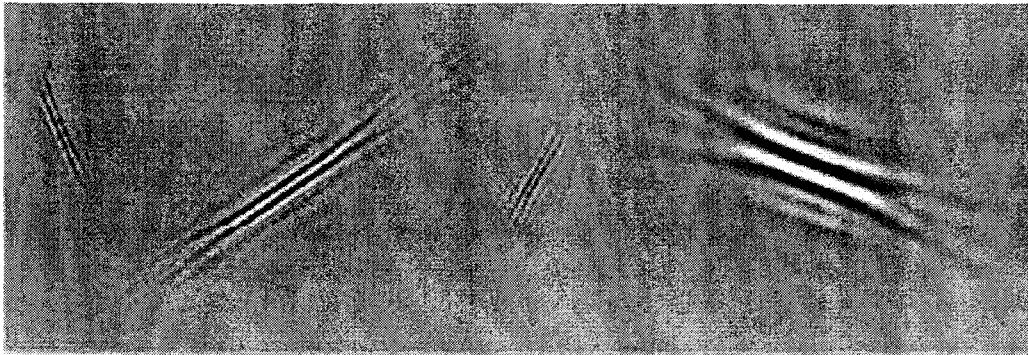


Figure 1.3: Examples of the Curvelets.

[9], Curvelet [11, 12] and Contourlet [32]. The latter two methods have attracted much attention. The construction of the Curvelet transform is based on windowing the subband images into blocks and applying the Ridgelet transform on these blocks. Figure 1.3 shows some examples of the Curvelet. Since the Ridgelet transform is defined by polar coordinates, the implementation of Curvelet transform for discrete images defined by the rectangular coordinates is a challenging task. On the other hand, the construction of the Contourlet transform works naturally on the discrete images. First, it applies the Laplacian pyramid introduced in [6] to achieve a multi-scale image decomposition. Each bandpass image resulting from the decomposition procedure is fed into a directional filter bank. Both methods are able to achieve optimal approximation behavior in a certain sense for 2-D piecewise smooth functions in  $\mathbf{R}^2$ , where the discontinuity curve is a  $\mathbf{C}^2$  function. Furthermore, they provide flexible schemes in that they allow for finer directional selectivity at finer scales.

In this evolution of transforms, two important properties are worth noting. First, these transforms are over-complete, or redundant. They represent a given signal with a set of atoms, the number of which is (much) larger than the dimension of the signal. For example, the version of the discrete Curvelet transform in [116] has a redundancy factor equal to  $16J+1$ , where  $J$  is the number of multi-scale levels, and the redundancy



factor of the Contourlet is 1.3. Second, these transforms are sparse in that most of the transformation coefficients are small in magnitude. Therefore, most information is concentrated in a few atoms. Some examples of the coefficient amplitude distribution of wavelet transform, Curvelet transform and Contourlet transform are shown in Figure 1.4. In each case, over 70 percent of the coefficients possess an amplitude that is only 10 percent of the maximum. This property explains why the simple thresholding method based on these transforms can achieve a state-of-the-art image denoising performance [116, 35, 76] and why the JPEG2000 image compression method has become a success [120].

The above two properties would become more interesting if we look at some recent results from the field of vision research. It is well known that the simple cells in the mammalian primary visual cortex have some distinct properties. That is, they are tuned to be spatially localized, oriented and bandpassed [8]. These properties of the simple cells are believed to make the visual system more efficient in processing visual information by finding the sparse structure available in the input and thus increasing the independence of the responses of the visual neurons [39]. Meanwhile, if we explicitly put sparsity as the goal, we can actually learn from natural scenes the building blocks that have all of the above-mentioned properties [91], as shown in Figure 1.5. Strikingly, these sparse components resemble closely the aforementioned Curvelets, as shown in Figure 1.3. Therefore, it has been suggested that the human visual system is more likely working in an over-complete and sparse way [92]. On the one hand, it has a huge amount of visual neurons (mathematically, this corresponds to an over-complete dictionary). On the other hand, only a few neurons (this means sparsity) are excited in order to capture essential information from the scene.

In summary, all research results from computational harmonic analysis, signal processing and mammalian vision systems suggest that over-completeness and sparsity

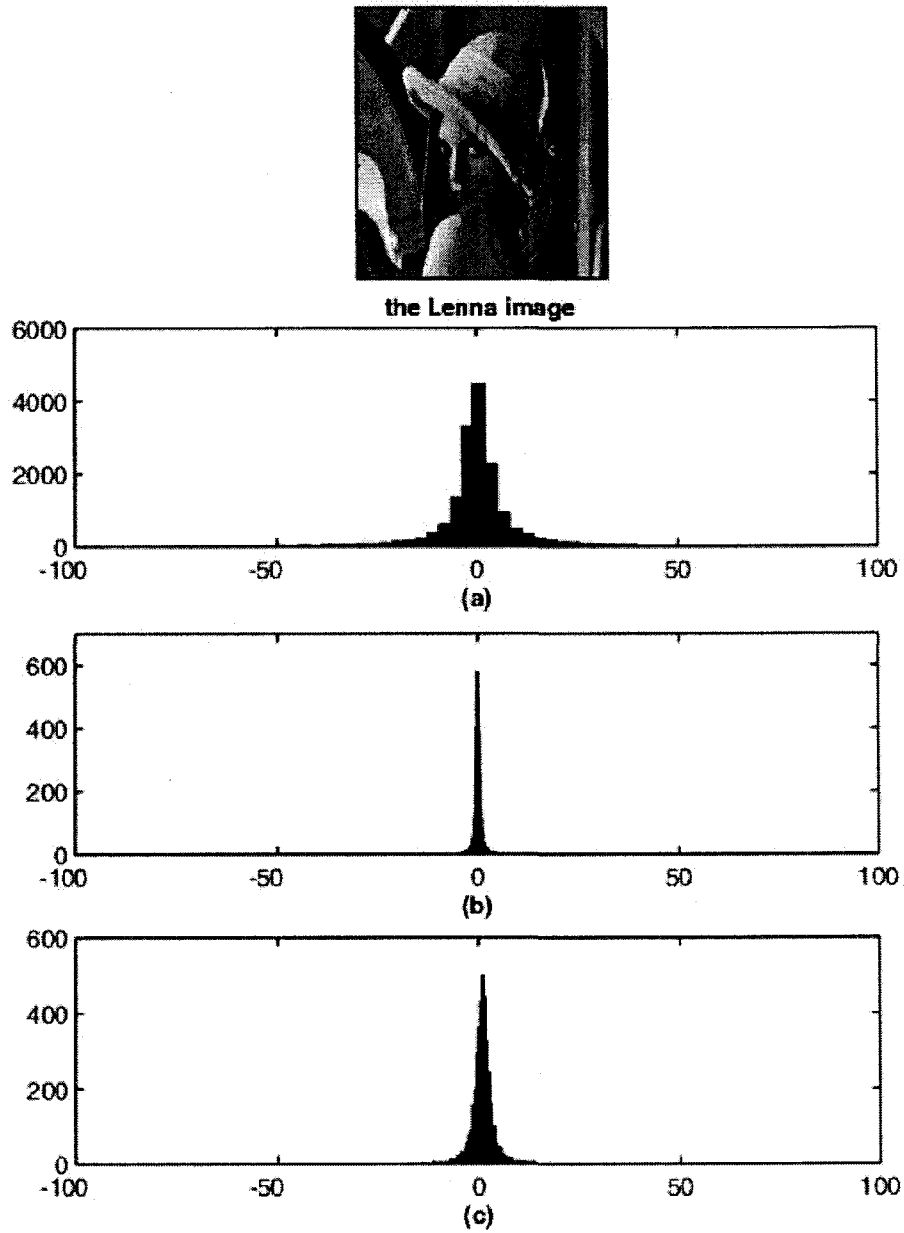


Figure 1.4: Sparse property of the coefficients of the wavelet, Curvelet and Contourlet transforms: (a) Amplitude distribution of the wavelet coefficients in the finest horizontal sub-band, (b) amplitude distribution of the Curvelet coefficients in one finest sub-band, and (c) amplitude distribution of the Contourlet coefficients in one finest sub-band.

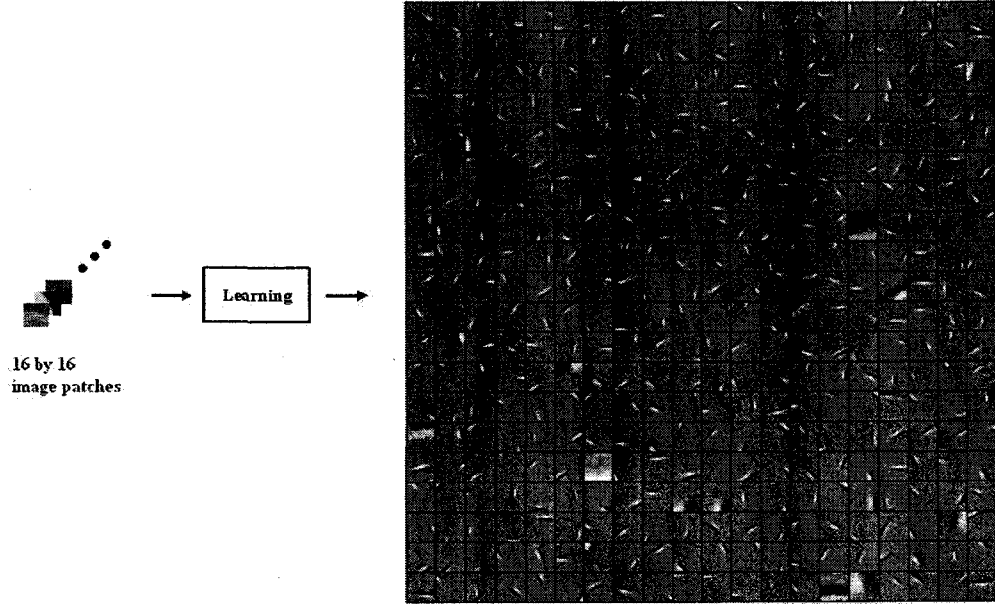


Figure 1.5: Experiment producing sparse components of the natural images in [91]. These sparse components are learned from 16 by 16 (in pixel) image patches, which are extracted from natural scenes with the goal of maximizing both preserved information and the sparsity of the expansions.

are innate properties in signals and are important for visual information processing. It can be worthwhile to investigate how to apply these two properties to pattern analysis, especially pattern detection and recognition. This is the problem to be addressed in this thesis.

## 1.2 Problem Statement

In this thesis, two feature extraction techniques are proposed that exploit the advantage of over-complete and sparse representations. First, a rotation invariant feature extraction method in the Ridgelet transform domain is proposed and applied to the problem of texture classification, where transformation-based methods have become an important class of feature extraction algorithms. A typical scenario here is to transform an image into its frequency domain, where features are extracted by generating a proper tiling in that domain. However, two problems exist. First, when

an image is rotated, its Fourier transform also rotates. Therefore, features extracted from spectrum information on a rectangular grid can be unstable. Second, in the frequency tiling, dyadic sub-band decomposition is usually adopted (such as Gabor multi-channel filtering and DWT). Therefore, this might not be optimal to capture most classification-related information. Solutions to these problems will be provided in this thesis.

As a case study, we also apply some recent texture classification methods, including the proposed one, to a problem from document image processing field, the problem of script identification. The evaluation results show that the proposed method performs very well in this case.

The second feature extraction technique is based on the analysis of local sparse structures. This feature extraction technique is then applied to the problem of text detection from scene images and handwritten Farsi numeral recognition. Existing redundant sparse transforms do not work well in these problems, where local features play a more important role. Furthermore, it is not obvious how suitable a given transformation is for sparsely describing the signals in question. In the proposed method, a local sparse structure is analyzed using small image patches in the input pattern using a similar strategy to that in [91]. A recently proposed method, the K-SVD algorithm [3], is also adopted to learn the sparse structure from the data.

One way to apply the learned sparse structure is to identify those signals/patterns that can be effectively represented by this structure. This procedure is useful in pattern detection problems, of which text detection in scene images is an example. Specifically, sparse structure is learned from the curve segments on the boundaries of the characters and text detection is achieved from the edge map of a given scene image. This idea has been inspired by the fact that, by only looking at the edge map of a scene image, human beings are able to tell which edge point is, or is not part

of a text. To simulate the over-complete and sparse way in which the human vision system is working, we have built an over-complete dictionary that gives sparse (or efficient) representation to text signals and non-sparse (or inefficient) representation to non-text signals. Thus, text detection is achieved by sparsity testing of the edge segments in the edge map.

While the above method shows promising results, it has a high time complexity and a high false alarm rate. To overcome these limitations, we propose a different method of using the sparse structure lying inside the data. Instead of using the edge map, we switch to the topographic map [87, 14] in the second proposed method. Thus, we convert the text detection problem to the shape classification problem. The latter is then solved by the combination of the sparse structure analysis and a machine learning technique, namely, the Convolutional Neural Network (CNN) [66].

Lastly, we show that the combination of sparse structure analysis technique and CNN is readily extended to other pattern recognition problems, such as character recognition. Specifically, the problem of handwritten Farsi numeral recognition is chosen as an example to show the merit of such a combination.

### 1.3 Thesis Outline

This thesis is organized as follows. Chapter 2 provides background information for the chapters that follow. Specifically, we introduce the material related to sparse structure analysis, including pursuit methods and the K-SVD algorithm. We also present the convolutional neural network and the Ridgelet transform.

Chapter 3 and Chapter 4 then discuss the problem of text detection from scene images. In Chapter 3, we first review the related literature and then we present the text detection method via sparsity testing. We move on to present the second text detection method in Chapter 4. This method identifies the candidate characters

from the topographic map [87, 14], which are actually shapes in the image that satisfy certain properties. Real characters are selected from the candidate characters through a shape classification technique that combines the sparse structure learned from shape boundaries and the CNN.

Chapter 5 applies the combination of sparse structure analysis technique and CNN to the problem of handwritten Farsi numeral recognition. We also compare this method with SVM, MQDF and CNN by itself.

Chapter 6 presents the application of Ridgelet transform to the problem of rotational invariant texture classification. It will be shown that Ridgelet transform is an ideal tool for rotation invariant texture classification since it is able to compute the spectrum information on a polar grid, instead of a rectangular grid. However, the wavelet transform as the building block of the Ridgelet transform has to be chosen properly in order to extract most classification-related information from the samples. Dyadic wavelets usually do not work well in this respect.

Finally, Chapter 7 concludes this thesis.

# Chapter 2

## Background Material

In this chapter, we present the background material on which the research endeavors are based. First, we present the Ridgelet transform, which is applied as a feature extraction tool for texture classification. Then, we introduce algorithms related to the analysis of the over-complete and sparse structure inside the data. This structure is able to give sparse representations of the data with an over-complete set of atoms. The application of some of these techniques in text detection and handwritten Farsi numeral recognition will be described in the following chapters. Finally, we briefly introduce a machine learning technique: the Convolutional Neural Network, which plays an important role in our selected shape recognition and handwritten numeral recognition problems.

### 2.1 Continuous Ridgelet Transform

The two-dimensional continuous Ridgelet transform in  $\mathbf{R}^2$  can be defined as follows [9]: Let  $\psi : \mathbf{R} \rightarrow \mathbf{R}$  with sufficient decay that satisfies the admissibility condition:

$$\int |\hat{\psi}(\xi)|^2 / |\xi|^2 d\xi < \infty, \quad (2.1)$$

which holds if  $\psi$  has a vanishing mean  $\int \psi(t)dt = 0$ . Here,  $\hat{\psi}$  stands for the Fourier transform of the function  $\psi$ .

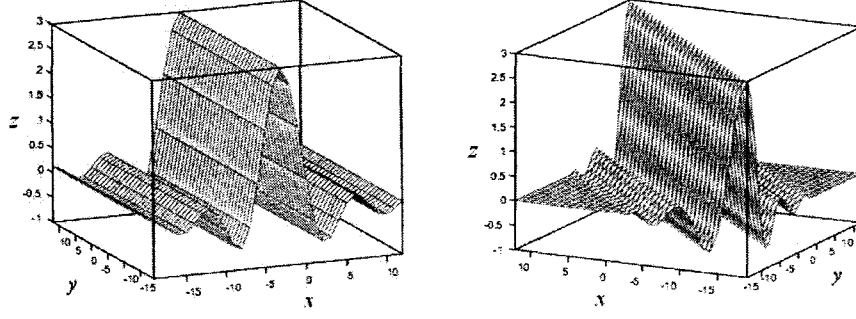


Figure 2.1: An example of ridgelets in two orientations. The ridgelet on the right is a rotated version of the one on the left.

For each  $a > 0$ , each  $b \in \mathbf{R}$  and each  $\theta \in [0, 2\pi)$ , we define the bivariate *Ridgelet*  $\psi_{a,b,\theta} : \mathbf{R}^2 \rightarrow \mathbf{R}$  as:

$$\psi_{a,b,\theta}(x_1, x_2) = a^{-1/2} \cdot \psi((x_1 \cos \theta + x_2 \sin \theta - b)/a). \quad (2.2)$$

A ridgelet is constant along the lines  $x_1 \cos \theta + x_2 \sin \theta = \text{const.}$  Transverse to the ridgelet is a wavelet. Figure 2.1 gives an example of ridgelets in two orientations.

Given an integrable bivariate function  $f(\mathbf{x})$ , we define its *Ridgelet transform* as:

$$\mathfrak{R}_f(a, b, \theta) = \int_{\mathbf{R}^2} \bar{\psi}_{a,b,\theta} f(\mathbf{x}) d\mathbf{x}. \quad (2.3)$$

Here,  $\bar{\psi}$  stands for the complex conjugate of  $\psi$ .

Ridgelet analysis may be viewed as wavelet analysis in the Radon domain. We denote the Radon transform as:

$$R_f(\theta, t) = \int_{\mathbf{R}^2} f(\mathbf{x}) \delta(x_1 \cos \theta + x_2 \sin \theta - t) d\mathbf{x}, \quad (2.4)$$

with  $\delta$  being the Dirac function. Then, the Ridgelet transform is the application of a 1-D wavelet transform to the slices of the Radon transform, as follows:

$$\mathfrak{R}_f(a, b, \theta) = \int_{\mathbf{R}} \psi_{a,b}(t) R_f(\theta, t) dt, \quad (2.5)$$



where  $\psi_{a,b}(t) = \psi((t-b)/a)/\sqrt{a}$  is a one-dimensional wavelet.

The Ridgelet transform will become the basic tool used in the rotational invariant texture feature extraction technique proposed in Chapter 6.

## 2.2 Sparse Approximation And Over-Complete Dictionary Learning

In this section, we mainly explain the K-SVD algorithm [3], which will be applied extensively to learn the over-complete and sparse structure from the data. Before delving into the details of the K-SVD algorithm, some useful notations are given first. Then, two major steps of the K-SVD algorithm, the sparse approximation step and the dictionary update step, are presented in detail.

### 2.2.1 Notations

Usually, the objects we deal with in pattern detection and recognition problems fit into the  $d$ -dimensional, real inner-product space  $\mathbf{R}^d$ , which is called the *signal space*, with  $d$  finite. Elements of the signal space are generally referred to as *signals*. The inner product is written as  $\langle \cdot \rangle$ , and we denote the corresponding Euclidean norm by  $\|\cdot\|_2$ . The distance between two signals is the Euclidean norm of their difference.

A *dictionary* for the signal space is a finite collection  $\mathbf{D}$  of unit-norm elementary signals. The elementary signals in a dictionary  $\mathbf{D}$  are called *atoms*, and each atom is denoted by  $\phi_i$ , where the parameter  $i$  is drawn from some index set  $\mathbf{I}$ . The indices may have an interpretation, such as the time-frequency or time-scale or time-scale-orientation localization of an atom, or they may simply be labels without any underlying metaphysics. The whole dictionary structure is thus:

$$\mathbf{D} = \{\phi_i : i \in \mathbf{I}\}.$$

The number of atoms in a dictionary is denoted as  $N = |\mathbf{D}| = |\mathbf{I}|$ , where  $|\cdot|$  denotes the cardinality of a set.

If a dictionary spans the whole signal space, we say that the dictionary is *complete*. In this thesis, we are interested in a *redundant* complete, or over-complete, dictionary, with  $N > d$ . In this case, each signal has an infinite number of representations due to the redundancy of the dictionary. A possibility is to choose the sparsest one from these representations.

Suppose we have a signal  $s$ , which is represented as a linear combination of the atoms in the dictionary  $\mathbf{D}$ :

$$s = \sum_{i \in \mathbf{I}} x_i \phi_i.$$

The *coefficient vector*  $\mathbf{x}$  is a vector whose  $i$ th element is  $x_i$ . The *sparsity* of the coefficient vector  $\mathbf{x}$  is measured by the  $l_0$  *quasi-norm*  $\|\cdot\|_0$ , which counts the number of non-zero components in the vector. The smaller this  $l_0$  norm, the higher degree of the sparsity in the coefficient vector  $\mathbf{x}$ .

### 2.2.2 Over-complete Dictionary Learning with K-SVD Algorithm

Suppose we have a set of data that contains  $L$  samples of the signal in question. These samples can be represented by a  $d \times L$  matrix  $\mathbf{Y}$ , where each column of  $\mathbf{Y}$  stands for one sample. We want to find an over-complete dictionary  $\mathbf{D}$  ( $\mathbf{D} \in \mathbf{R}^{d \times K}$  with  $K > d$ ) so that:

$$\mathbf{Y} = \mathbf{D}\mathbf{X} \tag{2.6}$$

and each column of  $\mathbf{X}$  (a coefficient vector) is as sparse as possible. In effect, a relaxed version of this problem is considered as follows:

$$\min_{\mathbf{D}, \mathbf{X}} \{\|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2\} \text{ subject to } \forall i, \|x_i\|_0 \leq T_0, \tag{2.7}$$

where  $T_0$  is a threshold specifying the maximum number of non-zero coefficients needed for the representation, and the notation  $\|\cdot\|_F$  stands for the Frobenius norm.  $T_0$  is a parameter that needs to be specified by the user. Another parameter we need to determine before applying this algorithm is the *redundant factor*  $R_f = K/d$ , which determines how redundant the dictionary should be. Note that  $R_f > 1$ . To solve this problem, the K-SVD algorithm has been proposed in [3]. Here, the “SVD” stands for the Singular Value Decomposition [117].

The K-SVD algorithm works iteratively. First, the target  $\mathbf{D}$  is assumed to be fixed and the algorithm tries to find the best coefficient matrix  $\mathbf{X}$ . This step is also called *sparse coding*, which is described in the next subsection. Second, once the sparse coding task is done, a second stage is performed to search for a better dictionary using singular value decomposition. This iteration continues until the algorithm converges.

### 2.2.2.1 Sparse Coding

Since the penalty term in the problem (2.7) above can be rewritten as:

$$\|\mathbf{Y} - \mathbf{DX}\|_F^2 = \sum_{i=1}^L \|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i\|_2^2, \quad (2.8)$$

when the dictionary  $\mathbf{D}$  is assumed to be known, problem (2.7) can then be decoupled into  $L$  distinct problems of the following form:

$$\min_{\mathbf{x}_i} \{\|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i\|_2^2\} \text{ subject to } \|\mathbf{x}_i\|_0 \leq T_0, \text{ for } i = 1, 2, \dots, L. \quad (2.9)$$

Each of these problems is a *sparsity-constrained approximation* problem.

When the dictionary  $\mathbf{D}$  is orthonormal, the sparsity-constrained approximation problem is easy to solve. Suppose  $\mathbf{y}$  is a sample signal, consider the orthogonal series:

$$\sum_{i \in \mathbf{I}} \langle \mathbf{y}, \phi_i \rangle \phi_i.$$

If we sort the terms so that the inner products are decreasing in magnitude, then we may truncate the series after  $T_0$  terms to obtain an optimal  $T_0$ -term approximation of the input signal. The coefficients in the representation of this approximation are just the inner products that appear in these  $T_0$  terms.

On the other hand, the sparsity-constrained approximation problem is NP-hard in general [27, 89]. Therefore, we have to live with some algorithms that give sub-optimal results. The two most common approaches are: greedy methods and convex relaxation methods. We focus only on greedy methods, due to their simplicity and efficiency.

A greedy method for sparse approximation constructs a solution one step at a time by selecting the atom most strongly correlated with the residual part of the signal and using it to update the current approximation. We are going to present two of the most prevalent greedy techniques, *Matching Pursuit* and *Orthogonal Matching Pursuit*. In both techniques, suppose we have an input signal  $\mathbf{y}$  and we want to find the coefficient vector  $\mathbf{x}$  that solves the sparsity-constrained approximation problem.

#### A. Matching Pursuit(MP) [80]

The Matching Pursuit algorithm works as follows:

1. Initialize the coefficient vector  $\mathbf{x} \leftarrow 0$ , the residual  $\mathbf{r}_0 \leftarrow \mathbf{y}$ , and the loop index  $t = 1$ .
2. Determine an index  $i_t$  such that:

$$i_t \in \arg \max_i |\langle \mathbf{r}_{t-1}, \phi_i \rangle|.$$

3. Update the coefficient vector:

$$\mathbf{x}(i_t) = \mathbf{x}(i_t) + \langle \mathbf{r}_{t-1}, \phi_{i_t} \rangle.$$

4. Compute the new residual:

$$\mathbf{r}_t = \mathbf{r}_{t-1} - \langle \mathbf{r}_{t-1}, \phi_{i_t} \rangle \phi_{i_t}.$$

5. Increment the loop index:  $t = t + 1$ .

6. If the stopping criterion has not been met, return to Step 2.

Several points are worth mentioning in this algorithm. Step 2 is the greedy selection, which chooses an atom that is most strongly correlated with the residual part of the signal. Note that the MP algorithm may select the same index many times over when the dictionary is not orthogonal. This repetition occurs because the inner product between an atom and the residual does not account for the contributions of other atoms to the residual. Step 4 computes a new residual by subtracting a component in the direction of the atom  $\phi_{i_t}$ . If the dictionary is complete, it can be shown that the norm of the residual converges to zero as  $t$  approaches infinity [80].

The computational cost of this algorithm can be estimated as follows: the greedy selection in Step 2 nominally involves computing the inner products between the residual and all the atoms in the dictionary, which generally requires  $O(dN)$  floating-point operations. Steps 3 and 4 require only  $O(d)$  floating-point operations. If the loop is executed  $T_0$  times, then the cost of the algorithm is  $O(dT_0N)$ .

#### B. Orthogonal Matching Pursuit(OMP)[124]

OMP adds a least-squares minimization to MP to obtain the best approximation over the atoms that have already been selected. The algorithm works as follows:

1. Initialize the index set  $I_0 = \emptyset$ , the residual  $\mathbf{r}_0 \leftarrow \mathbf{y}$ , and the loop index  $t = 1$ .

2. Determine an index  $i_t$  such that:

$$i_t \in \arg \max_i |\langle \mathbf{r}_{t-1}, \phi_i \rangle|.$$

3. Update the index set  $I_t = I_{t-1} \cup \{i_t\}$ .

4. Find the solution  $\mathbf{x}$  of the least-squares problem:

$$\min_{\mathbf{x} \in \mathbb{R}^{I_t}} \|\mathbf{y} - \sum_{j=1}^t \mathbf{x}(i_j) \phi_{i_j}\|_2.$$

5. Compute the new residual:

$$\mathbf{r}_t = \mathbf{r}_{t-1} - \langle \mathbf{r}_{t-1}, \phi_{i_t} \rangle \phi_{i_t}.$$

6. Increment the loop index:  $t = t + 1$ .

7. If the stopping criterion has not been met, return to Step 2.

The most essential difference between MP and OMP lies in step 4 above, where the coefficient vector is the solution to a least-squares problem. Thus, OMP maintains a loop invariant:

$$\langle \mathbf{r}_t, \phi_{i_j} \rangle = 0, \text{ for } j = 1, \dots, t.$$

It follows that Step 2 always selects an atom that is linearly independent from the atoms that have already been chosen. In consequence, the residual must be equal to zero after  $d$  steps.

The computational cost of OMP can be analyzed in a similar way to MP. The only difference is in Step 4. Since the solution of the least-squares problem at iteration  $t$  can be built based on the solution to iteration  $t - 1$ , the time complexity at this step is  $O(td)$  and the total cost after  $T$  iterations would be  $O(dT(T + N))$ .

Another point we need to mention for MP and OMP is the stopping criteria. In the case of K-SVD algorithm, these algorithms will stop after  $T_0$  iterations.

### 2.2.2.2 Dictionary Updating

The update of the dictionary  $\mathbf{D}$  is achieved one column at a time. Assume that both coefficients  $\mathbf{X}$  and the dictionary  $\mathbf{D}$  are fixed and we put in question only one column  $\mathbf{d}_k$  in the dictionary and the coefficients that correspond to it, at the  $k$ th row in  $\mathbf{X}$ , are denoted as  $\mathbf{x}_T^k$ . Then the penalty term in (2.7) can be rewritten as:

$$\begin{aligned}\|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 &= \left\| \mathbf{Y} - \sum_{j=1}^K \mathbf{d}_j \mathbf{x}_T^j \right\|_F^2 \\ &= \left\| \left( \mathbf{Y} - \sum_{j \neq k} \mathbf{d}_j \mathbf{x}_T^j \right) - \mathbf{d}_k \mathbf{x}_T^k \right\|_F^2 \\ &= \|\mathbf{E}_k - \mathbf{d}_k \mathbf{x}_T^k\|_F^2.\end{aligned}\tag{2.10}$$

The multiplication of  $\mathbf{D}\mathbf{X}$  has been decomposed into the sum of  $K$  rank-1 matrices. Each  $K - 1$  of these matrices is assumed to be fixed. The matrix  $\mathbf{E}_k$  stands for the error for all  $N$  samples when the  $k$ th atom is removed.

Note that the SVD algorithm can not be directly applied here to solve  $\mathbf{d}_k$ , since the sparsity constraint can not be guaranteed this way. A remedy has been proposed in [3] which works as follows: Define  $\omega_k$  as the group of indices pointing to examples in  $\mathbf{Y}$  that use the atom  $\mathbf{d}_k$ , i.e., those where  $\mathbf{x}_T^k$  is nonzero. Thus:

$$\omega_k = \{i | 1 \leq i \leq K, \mathbf{x}_T^k(i) \neq 0\}.$$

Define matrix  $\mathbf{\Omega}_k$  of size  $L \times |\omega_k|$ , with ones on the  $(\omega_k(i), i)$ th entries and zeros otherwise. By multiplying  $\mathbf{x}_R^k = \mathbf{x}_T^k \mathbf{\Omega}_k$ , the row vector  $\mathbf{x}_T^k$  is shrunk and all zero components are discarded. The multiplication  $\mathbf{Y}_k^R = \mathbf{Y} \mathbf{\Omega}_k$  creates a matrix of size  $d \times |\omega_k|$  that includes only sample signals currently using the  $\mathbf{d}_k$  atom. A similar effect happens with  $\mathbf{E}_k^R = \mathbf{E}_k \mathbf{\Omega}_k$ , where error is computed only from samples that use the  $\mathbf{d}_k$  atom.

Now, by using SVD to minimize:

$$\|\mathbf{E}_k \boldsymbol{\Omega}_k - \mathbf{d}_k \mathbf{x}_T^k \boldsymbol{\Omega}_k\|_F^2 = \|\mathbf{E}_k^R - \mathbf{d}_k \mathbf{x}_R^k\|_F^2, \quad (2.11)$$

the sparsity constraint of the coefficient vector  $\mathbf{x}_T^k$  is maintained. Furthermore, while updating one atom in the dictionary, this algorithm simultaneously updates the affected entry in all the coefficient vectors.

## 2.3 Convolutional Neural Network (CNN)

A Convolutional Neural Network is a neural network with a “deep” supervised learning architecture that has been shown to perform well in visual tasks [66, 110]. A CNN can be divided into two parts: automatic feature extractor and a trainable classifier. The automatic feature extractor contains feature map layers that extract discriminating features from the input patterns via two operations: linear filtering and down-sampling. In the linear filtering process, the size of the filter kernels in the feature maps is set to 5 by 5 and the down-sampling ratio is set to 2. A back-propagation algorithm is used to train the classifier and learn the weights in the filter kernels.

Instead of using the CNN with a more complicated architecture like LeNet-5[66], we chose a simplified implementation as proposed in [110]. The network architecture is shown in Figure 2.2. The input layer is an  $S_1 \times S_1$  matrix containing an input pattern. The second (with  $N_1$  feature maps and  $S_2 = (S_1 + 1)/2$ ) and the third (with  $N_2$  feature maps and  $S_3 = (S_2 + 1)/2$ ) layers are two feature map layers, which are trained to do feature extraction at two different resolutions. Each neuron in these two layers is connected to a 5 by 5 window in the previous layer, with the strength of the connection defined by the weights in the filter kernel. Neurons in one feature map share the same filter kernel. The remaining part of the architecture consists of



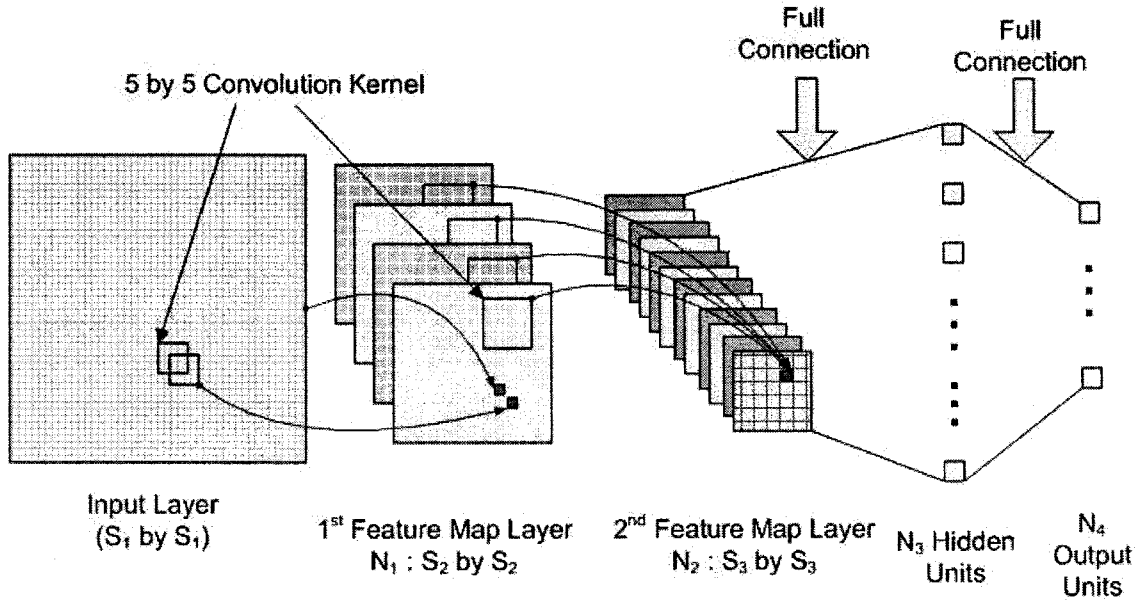


Figure 2.2: Adopted CNN architecture.

a fully connected multi-layer perceptron with  $N_3$  neurons in the hidden layer and  $N_4$  neurons in the output layer. Here,  $S_1$ ,  $N_1$ ,  $N_2$ ,  $N_3$  and  $N_4$  are the parameters we need to determine before using the CNN.

The structure of the CNN provides an ideal platform to incorporate the sparse and over-complete structure, which is learned from the data using the K-SVD algorithm. As will be presented later, the atoms in the learned structure are applied to initialize the weights of the filters in the first feature map layer of the CNN. Such a combination is a good feature analysis technique and it will be applied in Chapters 4 and 5.

## Chapter 3

# Text Detection from Scene Images Using Sparsity Testing

In this chapter, we present our first method for text detection from scene images, which is based on sparsity testing using the edge map generated by the Canny edge detector. First, we briefly introduce the background of this problem and its possible applications. Then we present related work around this problem. After that, we present the proposed method and finally, the experimental results.

### 3.1 Introduction

As mobile digital imaging devices, such as digital cameras, cell phones, camcorders etc., become more and more ubiquitous, researchers in the field of document image processing and recognition (DIPR) are getting increasingly interested in studying the images/videos captured by these devices. Naturally, one of their top concerns is to automatically extract, if there is any, text information from these documents, since text is a valuable source of high-level semantics and is useful in many applications.

In the literature, texts in images are usually classified into the following two categories:

- *Artificial text*, or superimposed text, is superimposed on the image after the

image/video has been taken with the camera. It is designed to show some important information. Therefore, it is rendered with high a contrast against the background so that it can be read easily. To increase the readability of the text against the complex background, the designer of the screen text systems usually uses artificial shadows to enhance the contrast of the text against its background. While making text detection easier, this also makes the text segmentation more difficult for automatic text reading systems. Examples of artificial text are subtitles in movies or dates on photos, etc.

- *Scene text*, on the other hand, is the text printed on the surface of some objects in the image. Due to different recording conditions, the text can be partly occluded, unevenly illuminated, distorted by perspective projection or by the surface properties of the objects. The text can also have varying sizes, styles and colors. To save space which is limited on handheld devices, the captured images usually are encoded in a compressed format, such as JPG. Such a compression procedure would introduce further degradation to the captured images. Examples of scene text include text on a road-sign, a car plate, or a book cover, etc.

In this research, we will focus on scene text detection in camera document images. This is a very challenging problem due to the above-mentioned properties.

Possible applications of detecting text in scene images include, but are not limited to, image and video indexing/retrieval, video surveillance, and license plate recognition. Furthermore, when connected with a city map database, it maybe possible to build a system that provides location information via the use of text clues extracted from the camera captured images. Such a system can be very useful for people in a foreign city. It may also be possible to help tourists to overcome the language barrel

by translating the detected text into their native languages.

## 3.2 Related Works

Many text detection methods can be found in the literature. Generally, three types of information in the image have been exploited for text detection: edge, texture and color.

### Edge-based methods

Methods falling into this category assume that text can be characterized as regions of high contrast and high frequencies. In [129], edges are grouped into strokes, and then strokes are aggregated to form character strings. A complex-valued edge orientation map in a small window is calculated and then fed into a neural network for classification in [70]. Chen et al. [18] use the morphological dilation operation to connect horizontal and vertical edges into text region candidates, which are further verified by machine learning techniques. Garcia et al. [43] proposed a new feature referred to as variance of edge orientation. This method relies on the fact that text strings contain edges in many orientations. However, some background areas, such as trees, may also have this property. In [74], edge detection is conducted directly in the RGB color space. Limitations of these methods include: edges are sensitive to the imaging conditions and it is usually difficult to differentiate between edges from characters and edges from background details.

### Texture-based methods

Methods in this category treat text as a type of texture. They usually divide the whole image into blocks and extract texture features in each block. Commonly used features include: wavelet features [48, 104, 44, 106] and autoregressive

features [59]. Then proper classifiers, e.g., neural network [48], support vector machine (SVM) [59] or  $K$ -means classifier [44, 106], are employed to classify image blocks into text or non-text. The limitation of texture-based methods in detecting text in scene image is that texture property in the text region is usually weak and it is difficult to differentiate it from background textures. Another problem with such methods is that handling texts in different sizes is not a straightforward task.

### Color-based methods

Methods using color information usually assume that a text string contains a uniform color. For example, in [51], Jain and Yu propose a method called connected color component analysis (CCA) for text location. CCA uses spatial structure analysis of the color connected components and can work well in situations like the characters on book covers, news titles, and video captions. In [86], clustering is performed in Lab color space and then connected component analysis is applied to detect text regions. Low level features including color continuity, gray-level variation and color variance are exploited to detect text in scene images [58]. The detected text regions are verified by multi-scale wavelet features and SVM. In scene images, these methods usually do not work well since the uniform color assumption is no longer valid.

Some methods also use the combination of the three above-mentioned information. In [23], a hierarchical detection framework that embeds multi-resolution, multi-scale edge detection and Gaussian mixture model based color analysis has been proposed. In [42], Gao and Yang combine the edge information with the color layout analysis to detect scene text. In [131], color, texture and OCR feedback are combined to locate the text regions in the image. Chen and Yuille [22] extract three sets of

features, including mean and standard deviation properties of the image intensity within predefined box patterns, histogram features of the image intensity and gradient and edge-based features. These features are used to build weak classifiers, which are then combined into one strong classifier to classify an image region into text or non-text, using the AdaBoost technique. The method proposed in [57] is a combination of two methods. The first one extracts candidate text regions using edge information extracted by the Laplacian operator. The second one is a split-merge algorithm using image intensity information.

### 3.3 Text Detection via Sparsity Testing

The first text detection method we propose is to build an over-complete dictionary that gives sparse representations to text signals and non-sparse representations to non-text signals. In other words, we are trying to build a dictionary that can represent text signals more efficiently and use that dictionary for text detection. The rationale behind the proposed method lies in the following facts and assumptions:

- We notice that, in most cases, advertisers tend to make the text somehow distinct from the background, so that the text can be easily noticed and read by human beings. This means that analyzing edge information in an image might be a good choice for text detection. Actually, many methods in the literature [129, 70, 18, 43, 23, 42] did make use of edge information.
- Except for some extreme cases (such as text in very small sizes), humans can tell where the text lies by only looking at the edge map of an image. As explained in Chapter 1, the human vision system is very likely working in a sparse-overcomplete way. We want to simulate this strategy and apply it to the text detection problem.

Our proposed method first calculates the edge map of the input image using Canny operator [13]. Then, it scans the edge map with a small window. Each edge segment falling into the small window undergoes a sparsity test. If it passes the test, which means it can be efficiently represented by the dictionary, we label those edge pixels on the segment as text. Those text edge pixels are then grouped into text lines.

### **3.3.1 Learning Over-Complete Dictionary for Text**

The core of our method is to create an over-complete dictionary that gives sparse representations to text or characters. Many transforms, such as Gabor transform, Wavelet transform, Ridgelet transform and Curvelet transform, etc., can be used to give sparse representations for different types of signals. However, these transforms are not readily applicable in our situation, where we try to identify text connected components from the edge map of a given image. Actually, to search for sparse representation of given signals is a difficult task itself and is still an active research topic.

Instead of applying those transforms mentioned above, we turn to methods that can learn the over-complete representation from data. There are already several methods available for this task, including probabilistic methods [69, 63], the Method of Optimal Directions (MOD) [37] and the union of orthonormal bases [68], etc. Recently, a new method named K-SVD [3] has been proposed, which will be adopted in our experiments for learning the over-complete dictionary from data.

#### **3.3.1.1 Learning Procedure**

To train an over-complete dictionary for text signals using the K-SVD algorithm, we need to first collect some data for training. It would be extremely tedious to manually collect data from real scene images. Therefore, we have used images of isolated machine-printed characters, which include 10 digits and lowercase and uppercase

letters. There are 12 typefaces (such as Times New Roman, Arial, etc.), 4 styles (normal, bold, italic and bold italic), and 2 size variations (11-point and 8-point) in these images. The data has been collected by using the following procedure:

1. Generate the edge map for each character image using the Canny operator.
2. Scan each edge map with a small window of size  $16 \times 16$  pixels. If a long enough edge segment (with a length longer than 16 pixels) falls into the small scanning window, we generate a new observation by doing the following: initialize with zero a new image of the same size as the scanning window; center the edge segment into this image; convert this image to a vector by concatenating its rows.
3. The newly generated observation is added into the observation set if it is significantly different from other observations already in the set. The difference between observations is measured by the Euclidean distance between vectors.

In total, we have collected 12278 observations from the edge maps of these isolated character images. On the left of Figure 3.1, we give some examples of the observations used for dictionary learning.

It would be tempting to use the whole character edge map as one observation. This idea does not work well since in a real edge map of a real scene image, it is very difficult to get a complete character edge map. A dictionary learned in this way would find it difficult to be applied to real images. Furthermore, using the whole edge map of a character usually results in long observation vectors and would take the K-SVD too much time to learn. Therefore, instead of working on the whole edge map, we only look at the curve segments on the edge of characters.

We also need to pay attention to the size of the sliding window. If it is too large, again it will generate very long observation vectors and thus increase the difficulty in



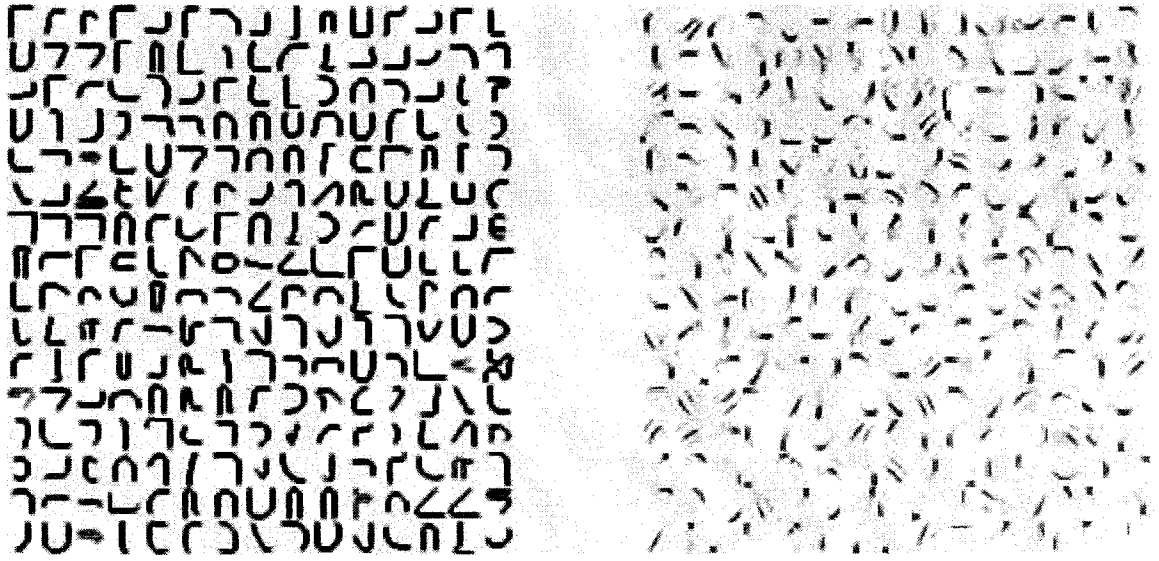


Figure 3.1: Left: Examples of the observations taken from character edges for learning the dictionary. Right: The most frequently used atoms learned by K-SVD.

learning. If it is too small, the curve segments will not bear too much character shape information and therefore will not be so useful in text detection. In our experiments, we found that size  $16 \times 16$  is the best tradeoff.

Another important point is that we need to center the edge segment into the newly generated image. This is because the learned dictionary  $D$  is not shift-invariant: it will give quite different representations to shifted versions of the same observation. Therefore, in both data collecting and text detection, we explicitly center the edge segment at the center of the small image window.

In applying the K-SVD algorithm, there are two important parameters we need to choose: the redundancy factor  $R_f$  and the number of elements in each linear combination, or  $T_0$ . These values are chosen empirically so that a good text detection performance can be achieved. Details of the parameter selection process will be described in section 3.3.3.

### 3.3.2 Text Detection Algorithm

In this section, we present in detail how to detect text through a sparsity test using the learned dictionary. This method consists of four procedures: Preprocessing, Connected Component Labeling, Layout Analysis and Multi-scale Processing.

#### 3.3.2.1 Preprocessing

Given an input image, we first apply the Canny operator to extract the edge map. We adopt the implementation of this operator in the Matlab<sup>®</sup> image processing toolbox and use the default parameters. Then, connected component analysis is applied to remove from the edge map those edge pixels that belong to lines or large connected components.

#### 3.3.2.2 Connected Components Labeling

The CC labeling procedure actually consists of two stages. The first stage is called pixel-level labeling. The edge map is scanned using a  $16 \times 16$  pixels window in a similar way as we scan the character edge map in data collecting. The scanning step is set to 4 pixels in both horizontal and vertical directions. If a long enough edge segment (with a length longer than 16 pixels) falls into the small window, we generate an observation vector  $\mathbf{y}$ , as we have done in data collecting, and we solve for  $\mathbf{x}$  in the following equation using OMP (see section 2.2.2):

$$\mathbf{y} = \mathbf{D}\mathbf{x} \tag{3.1}$$

Here,  $\mathbf{D}$  is the over-complete dictionary we have learned. If  $\mathbf{x}$  is sparse enough, we label pixels on this edge segment as text. By saying sparse enough, we mean that  $\mathbf{x}$  has at most  $T_1 = 16$  non-zero components. This number has been empirically chosen based on a dataset of 120 real scene images. The results of this pixel-level labeling on three sample images are shown in Figure 3.2 in section 3.3.3.

After the pixel-level labeling stage, we group the edge points into connected components and label each connected component as text if more than 25 percent of its edge points have been labeled as text at the pixel labeling stage. Here, we intentionally choose a low threshold to avoid missing any true text components.

### 3.3.2.3 Layout Analysis

Layout is defined as the way characters are arranged when they are printed. In the scene images, we can still take advantage of the layout information to filter out some of the non-text components. In this work, we mainly exploit the following layout knowledge:

- Characters are usually arranged into horizontal or vertical lines. Since the image can be captured at various angles, the text lines can be skewed.
- Within one text line, characters are roughly of the same size.
- The distance between neighboring characters in a text line is within a certain threshold.

The layout analysis procedure can be divided into two stages: Connected Component (CC) merging stage and the verification stage. In the first stage, we differentiate two sets of connected components. The first set contains connected components that are labeled as text by the connected component labeling procedure in section 3.3.2.2, and we call it set A. The second set, B, contains the remaining connected components in the image. Two connected components are merged if the following two conditions are satisfied:

1. They are of the same height, or,

$$\frac{\text{minimum height of the two CCs}}{\text{maximum height of the two CCs}} \geq 0.5 \quad (3.2)$$

2. Their horizontal distance is shorter than their average height.

The merging process starts by picking out one CC from set A and trying to merge it with other CCs in set A and set B according to the above merging conditions, until no more merging could happen. These merged CCs are put into a data structure, called a “LINE”. This merging process continues until set A has been exhausted.

The above merging algorithm works for horizontal text lines only. However, it is easy to extend it to handle vertical cases too.

The verification stage mainly focuses on those “short” LINEs, or, LINEs with no more than 3 CCs. We will discard a short LINE if, among all the edge points it has, the percentage of the edge points that are labeled as text is less than 80.

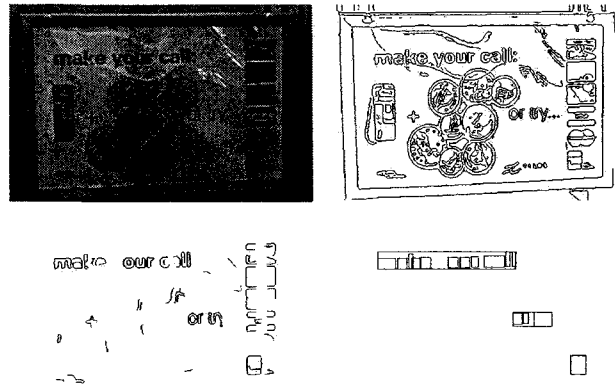
The detected text LINEs are shown in Figure 3.2 in section 3.3.3. In these examples, we draw the bounding box of each LINE along with the bounding boxes of the connected components inside that LINE.

#### **3.3.2.4 Multi-scale Processing**

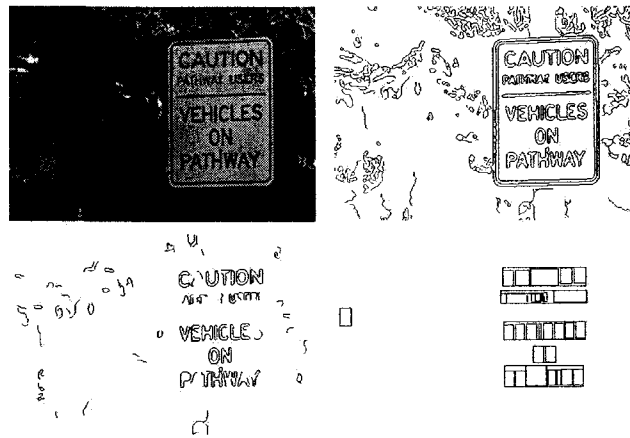
The text detection algorithm described so far works fine on texts with heights between  $20 \sim 40$  pixels. One possible method to extend this algorithm to handle texts with larger sizes is to work on different scales of the input image. In our experiments, three scales are considered. Each scale is generated from a previous scale by down-sampling with a factor of two in both directions. The results of detection at different scales are combined by merging overlapped LINEs.

### **3.3.3 Experimental Results**

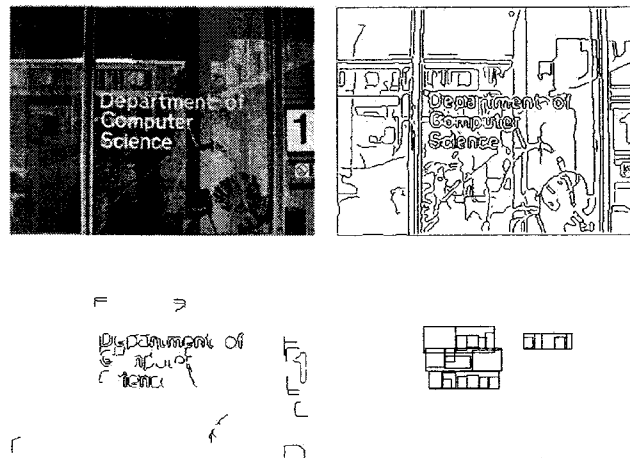
A database of 120 scene images has been collected to help us in selecting the parameters used in the proposed method. To evaluate the proposed method, we use the 2003



(a) Example A.



(b) Example B.



(c) Example C.

Figure 3.2: Some examples generated by the proposed method. In each example, the top left gives the input grayscale image; the top right gives the edge map generated by the Canny operator; the bottom left gives the pixel-level labeling results; the bottom right gives the text LINES detected after layout analysis.

ICDAR (International Conference on Document Analysis and Recognition) Text Location Contest trial test database, which is publicly available and can be downloaded at <http://algoval.essex.ac.uk/icdar/Datasets.html> [122]. Included in this database are 251 images and the ground truth of the word bounding boxes of all the target texts in these images.

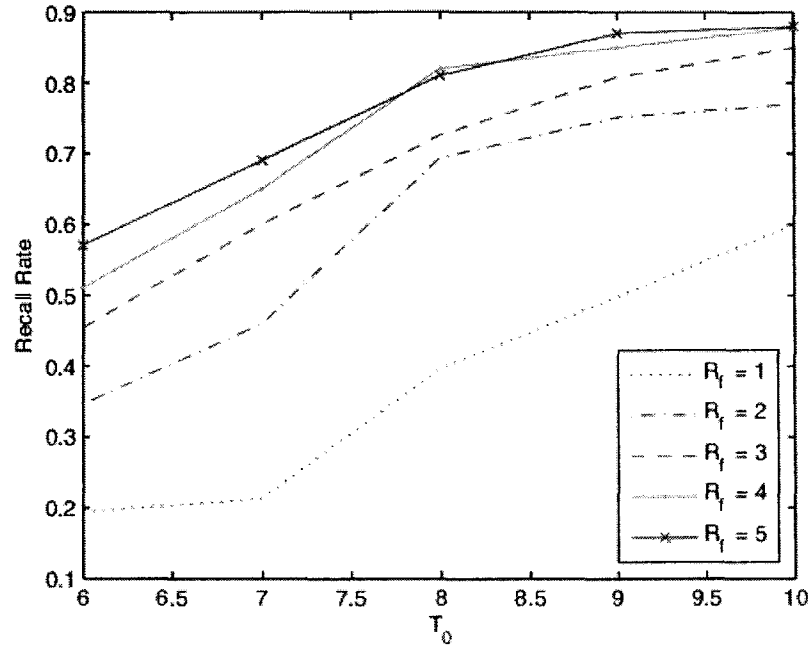
The evaluation is based on the notions of *precision* and *recall*. Precision,  $p$ , is defined as the number of words correctly detected divided by the total number of words detected. Recall,  $r$ , is defined as the number of words correctly detected divided by the total number of words in the ground truth.

As mentioned above, we have experimentally selected two main parameters in the K-SVD algorithm: the redundant factor  $R_f$  and the sparse threshold  $T_0$ . Let  $R_f \in \{1, 2, 3, 4, 5\}$  and  $T_0 \in \{6, 7, 8, 9, 10\}$ . We have exhaustively tried all combinations of these two parameters on the training dataset and the performance of these experiments are shown in Figure 3.3.

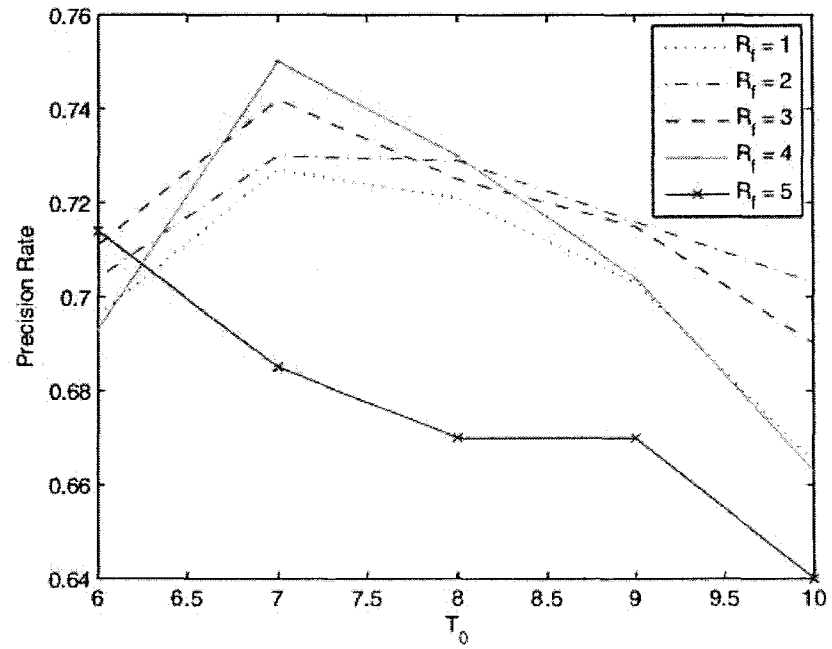
From these experiments, we can see that as the redundant factor  $R_f$  grows, the recall rate also grows. However, the difference between  $R_f = 4$  and  $R_f = 5$  is not significant. If we fix the parameter  $R_f$  and let  $T_0$  grow, the recall rate also increases, since more edge points can be labeled as text. The situation is different when it comes to the precision of text detection. When  $R_f$  starts to grow, the precision grows too, but very slowly. While  $R_f$  grows to 5, the precision starts to fall. For each fixed  $R_f$ , the precision gradually starts to fall while  $T_0$  grows.

As the tradeoff between recall rate and detection precision, we choose  $T_0 = 8$  and  $R_f = 4$ , which means we have 1024 atoms in the dictionary. On the right of Figure 3.1 in section 3.3.1.1, we show some of the most frequently used learned atoms in reconstructing the observations.

Even though many text detection methods can be found in the literature, most



(a)



(b)

Figure 3.3: (a) Recall rate of the proposed algorithm under different parameter settings. (b) Precision rate of the proposed algorithm under different parameter settings.

Table 3.1: Experimental results and comparison.

| Method          | $p$ (precision) | $r$ (recall) |
|-----------------|-----------------|--------------|
| [57]            | 56.3%           | 64.3%        |
| [58]            | 55.8%           | 68.9%        |
| Proposed method | 67.64%          | 75.23%       |

authors have reported their results on private databases. To the best of our knowledge, only two authors in [57, 58] have reported their results on the same public database adopted in this work. We include these results in Table 3.1, where we can see that both the recall rate and the precision rate of the proposed method are much better than the other two methods [57, 58].

### 3.4 Conclusion

In this chapter, a text detection method based on sparse representation is proposed. This idea has been inspired by the results from vision research. Experimental results show that such a mechanism could be an interesting computation model for text detection in images. It may also be possible to apply this model to other object detection problems.

The limitation of the proposed method mainly lies in that only very local information is considered in the dictionary learning and text detection. This results in a low precision of detection. Furthermore, the time complexity of the proposed method is also high, due to the calculation of the coefficient vectors for each edge segment. In the next chapter, we are going to present a different way to make use of the sparse structure inside the data. Using that way, we can significantly alleviate these two limitations.



## Chapter 4

# Text Detection from Natural Scene Images Using Topographic Maps And Sparse Representations

In Chapter 3, text detection is achieved by sparsity testing. This idea has been inspired by recent research results in vision research, which have suggested that sparse structure is an innate property of natural scene images and that mammalian visual systems have evolved, enabling them to process the spatial information by finding the sparse structure available in the input [39, 91, 92].

While showing promising results, sparsity testing has some limitations. First, it works on the edges generated by the Canny operator [13], which gives very local information. This results in a high rate of false alarms in text detection. Second, the computational complexity of this method is high since the size of the adopted over-complete dictionary is large and it takes time to do the sparsity testing.

In this chapter, a more sophisticated text detection method is proposed. This method makes use of an image representation technique named *Topographic Maps* [87, 14], to convert the text detection problem into a shape classification problem. This conversion can significantly improve the time efficiency issue and reduce the number of false alarms.

## 4.1 Introduction

The text detection method proposed in this chapter:

- 1) Converts the text detection problem to a shape classification problem, and
- 2) Performs shape classification by exploiting the sparse structure in the shape data.

The proposed method first calculates the topographic map from a given image. The topographic map contains the level lines that are organized into a tree. These level lines are capable of capturing the boundaries of characters in the images and thus can provide a good starting point for shape analysis. The non-text level lines are then efficiently removed by first analyzing the  $\varepsilon$ -meaningful boundaries and then applying adaptive thresholding to the maximal monotone intervals in the level line tree. Afterwards, the sparse structure is learned from the shape data using the K-SVD algorithm and integrated with a well-known machine learning technique - Convolutional Neural Network (CNN) - to classify the remaining level lines into text or non-text shapes. Finally, layout analysis is applied to complete the text line detection.

To justify the selection of the topographic map representation of an image in this research, we mention several important advantages that this representation possesses:

- It is invariant to global contrast change.
- It is a hierarchical representation since level sets are ordered by the inclusion relation.
- Object contours (especially character contours in our case) locally coincide very well with level lines. This is a very important property that makes level lines a better choice than edges to the text detection problem, since edge detectors usually fail near  $T$ -junctions and can not give complete character boundaries.

Furthermore, using a topographic map representation to convert the text detection problem to a shape classification problem improves our capability to handle texts in different sizes via size normalization before shape classification. Multi-resolution processing is not required by the proposed method.

## 4.2 Identification of Text Candidates Based on Topographic Maps

In this section, we introduce the topographic maps [14] and compare them with the edge maps given by the Canny operator. We use the topographic maps to efficiently identify shapes as text candidates. Afterwards, the shape classification procedure works on these candidate shapes and picks out true text shapes.

### 4.2.1 Topographic Maps - Background Information

Let  $\Omega$  be a domain in  $\mathbf{R}^2$  and  $u : \Omega \rightarrow \mathbf{R}$  be an image (a bounded measurable function). Given any  $\lambda \in \mathbf{R}$ , the *upper level set* of  $u$  is given by:

$$X_\lambda u = \{\mathbf{x} \in \mathbf{R}^2 : u(\mathbf{x}) \geq \lambda\}. \quad (4.1)$$

Note that the family of upper level sets is decreasing:

$$\forall \lambda \leq \mu, X_\lambda u \supset X_\mu u. \quad (4.2)$$

The *upper topographic map* of an image is a family of the connected components of the upper level sets of  $u$ . It is obvious that the upper topographic map is contrast invariant. Furthermore, the upper topographic map provides a complete image representation, since we can reconstruct the image from it [107]:

$$u(\mathbf{x}) = \sup\{\lambda, u(\mathbf{x}) \geq \lambda\} = \sup\{\lambda, \mathbf{x} \in X_\lambda u\}. \quad (4.3)$$

The *level lines*[14] of  $u$  are the boundaries of the upper level sets of  $u$ . If image  $u$  is such that, for each level set  $X_\lambda u, \lambda \in \mathbf{R}$ , the boundary  $\partial X_\lambda u$  is made of a finite or countable union of closed Jordan curves<sup>1</sup>, then the oriented level lines will perfectly define the level sets and hence the image  $u$ . As mentioned in [14], the discrete images always satisfy the above condition and therefore, we can use the terms “topographic map” and “level line” equivalently in this method.

### 4.2.2 Tree Representation of Level Lines

According to equation (4.2), the upper level sets are nested. This property is still valid when going from the whole level sets to their connected components. These inclusions can be represented by a tree. Monasse et al. [87] have shown that, however, this representation sometimes leads to a non-intuitive description of the inclusion relationship. Instead of working directly on level sets, they propose to focus on the level lines of these level sets.

In our case, only discrete images are considered. Any connected component  $\mathcal{C}$  of a level set is bounded and its border  $\partial\mathcal{C}$  is a union of Jordan curves:

$$\partial\mathcal{C} = \bigcup_i J_i(\mathcal{C}). \quad (4.4)$$

According to Jordan theorem<sup>2</sup>, each closed Jordan curve  $J$  has an interior and an exterior and the interior is, by definition, bounded. Furthermore, from all the  $J_i$ s in (4.4), only one Jordan curve has an interior that contains  $\mathcal{C}$ . This Jordan curve is denoted as  $J(\mathcal{C})$  and its interior is called a “shape”[87].

Based on these “shapes”, a tree structure is proposed in [87] as follows: each node corresponds to a shape, its descendants are the shapes that are included inside it and

---

<sup>1</sup>A Jordan curve is a plane curve which is topologically equivalent to (a homeomorphic image of) the unit circle, i.e., it is simple and closed.

<sup>2</sup>the Jordan curve theorem states that, if  $J$  is a simple closed curve in  $\mathbf{R}^2$ , then  $\mathbf{R}^2 - J$  has two components (an “inside” and an “outside”), with  $J$  the boundary of each.

its parent is the smallest shape that contains it. The boundaries of these shapes are most interesting to us, since they define the character boundaries clearly.

### 4.2.3 Identification of Text Candidates

Before we continue, let us look at two examples as shown in Figure 4.1, and discuss the differences between the traditional edges (the output of the Canny edge detector [13]) and the level lines. In Figure 4.1(a), we show a rather simple example, where the almost uniform gray background is divided by a diagonal white strip. The resulting two diagonal edges would cut the boundaries from any foreground character that intersect with them into broken parts and thus the complete character shape information would be lost, as shown in Figure 4.1(c). On the contrary, we can see that in Figure 4.1(b), with properly chosen  $\lambda$  ( $\lambda = 80$  for this sample), the character shape information is perfectly retained.

In Figure 4.1(e), we show a difficult sample with a very complex background. After applying the Canny edge detector, we find that the edges from foreground character boundaries are buried in the edges from background details and, in some cases, they are broken again (see Figure 4.1(g)). On the other hand, level lines still work fine on this sample (see Figure 4.1(f)) with  $\lambda = 194$ .

The above two examples show that level lines work better in preserving the shape information of the characters in the image when compared with traditional edge detectors. However, one question left unanswered is this: what about the parameter  $\lambda$ ? Generally speaking, we do not know which  $\lambda$  to use in order to get those nice shapes of characters from a given image. The grayscale values of the characters in the scene images can range from black to white. This property requires us to take into consideration level lines on all possible grayscale values ( $\lambda = 0, \dots, 255$  with scene images). However, this requirement brings another problem: there could be too

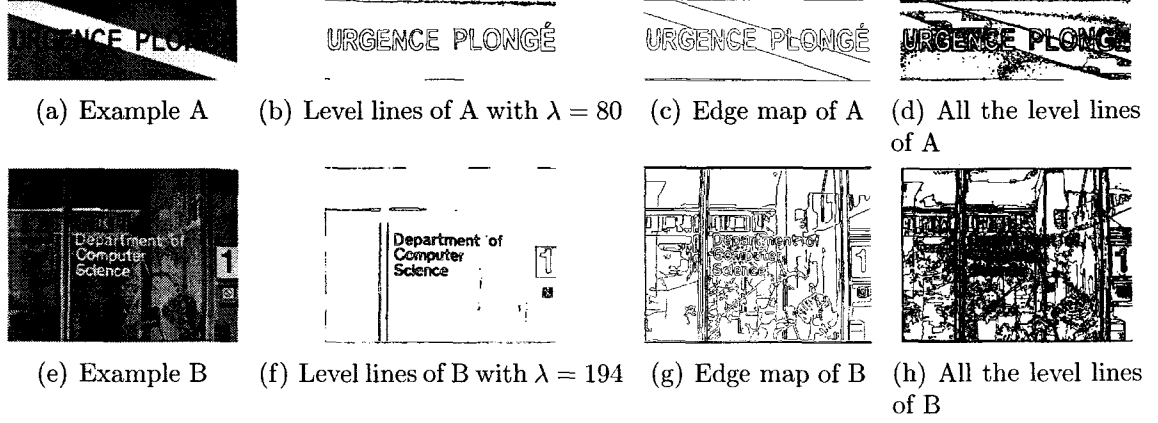


Figure 4.1: Comparison between level lines and edges with two examples.

many level lines in a scene image, as shown in Figures 4.1(d) and 4.1(h). Therefore, to make the level line representation useful in the text detection problem, one has to come up with a method that efficiently identifies candidate text shapes and filters out other non-text shapes.

In the proposed method, the identification of text candidates is achieved via the following two steps: Selecting  $\varepsilon$ -meaningful boundaries and adaptive thresholding of maximal monotone interval.

#### 4.2.3.1 Selecting $\varepsilon$ -Meaningful Boundaries

This is actually a step in the edge detection method proposed in [29], which is an application of a basic principle of perception described by Helmholtz. The key point of this principle says that perceptually “meaningful” structures may be viewed as exceptions to randomness.

Suppose  $C$  is a level line of the image  $u$  with length  $l$  and there are  $N_l$  level lines in the image. Then  $C$  is called an  $\varepsilon$ -meaningful boundary if:

$$NFA(C) \equiv N_l H \left( \min_{\mathbf{x} \in C} |Du(\mathbf{x})| \right)^{1/2} < \varepsilon, \quad (4.5)$$

where  $D$  is the gradient operator and  $H(\mu) = P(|Du| > \mu)$  is the distribution of

the gradient norm (a measure of contrast) of image  $u$ , which is empirically estimated using:

$$\forall \mu > 0, P(X > \mu) = \frac{\#\{\mathbf{x}, |Du(\mathbf{x})| > \mu\}}{\#\{\mathbf{x}, |Du(\mathbf{x})| > 0\}}$$

where the symbol  $\#$  gives the cardinality of a set. As discussed in [29], the  $\varepsilon$  is set equal to 1 in practice.

The number  $NFA$  is called Number of False Alarms and it measures the meaningfulness of a level line. Smaller NFAs indicate perceptually more meaningful level lines. After this step, we only keep those contrasted level lines that satisfy equation (4.5).

#### 4.2.3.2 Adaptive Thresholding of Maximal Monotone Interval

The  $\varepsilon$ -meaningful boundaries are still redundant in that there are many “fat” edges, which are made of well contrasted parallel level lines. These “fat” edges happen more often at the boundaries of characters in the scene images since, for easy readability, characters in real scenes (advertise, signs, etc.) are generated in a way that makes them well contrasted from their backgrounds. The consequence of this property is that similar contrasted level lines delineating the character shapes tend to occur at consecutive gray levels. This phenomenon has been made obvious in Figure 4.2, which shows the *frequency maps* of the two example images in Figure 4.1. The frequency maps are created by accumulating, at each pixel, the number of instances when that pixel becomes a point on some level line in each image and then normalizing these numbers into  $[0,1]$  by dividing them with the maximal possible number of occurrences at any pixel. As we expected, pixels located along the character boundaries usually achieve higher frequencies.

The above observation suggests that the occurrence of those “fat” edges gives a useful clue for text detection. The problem here is how to measure the “fatness” of



Figure 4.2: Frequency maps of the images in Figure 4.1 shown as images. In these images, the darker the pixel is, the higher frequency it has.

the edges. This can be done by thresholding the *Maximal Monotone Interval* in the level line tree.

Note that the above level line selecting procedure maintains the inclusion relationship among the selected  $\varepsilon$ -meaningful boundaries. Therefore, these boundaries can still be represented as a tree and furthermore, the *maximal monotone level line interval*[29] can be defined in this tree. This level line interval is monotone in that any level line in this interval has a unique descendant. It is maximal since no more level lines can be added. Furthermore, the gray levels on which the level lines in this interval are detected are either increasing or decreasing.

Based on the maximal monotone level line interval, we impose a similarity constraint and define the *maximal similar level line interval*. Due to the time efficiency consideration, the similarity measure here is a very simple one, which is described as follows. Let  $C_1$  and  $C_2$  be two level lines. We say that  $C_1$  and  $C_2$  are similar if:

$$|area(C_1) - area(C_2)| < 2 \max(l_1, l_2),$$

where  $area(C)$  gives the area of the shape defined by level line  $C$  and  $l_1, l_2$  are the lengths of  $C_1$  and  $C_2$ , respectively. Thus, a *maximal similar level line interval* contains a consecutive sequence of level lines in the maximal monotone level line interval and



each neighboring pair of level lines satisfy the above similarity constraint.

If the number of level lines in the maximal similar level line interval is large, these level lines are more likely to reside at the boundary of some character. Therefore, we pick those maximal similar level line intervals with the number of level lines larger than a threshold  $t$ , and in each of these picked out level line intervals, only the level line with the smallest  $NFA$  is retained as the character candidate.

The threshold  $t$  has to be chosen carefully so that it will adapt to the local contrast variation in the image. Naturally, when the contrast between foreground characters and the background is low, we expect a small threshold. On the other hand, when the contrast is large, we need a higher threshold. In the proposed method, we introduce a nonlinear function  $T$  as:

$$T = 0.5 * (1 + \tanh(\alpha * (c_l - \beta))), \quad (4.6)$$

where  $c_l$  is the local contrast calculated as the variance of the grayscale values of the pixels around each level line.  $\alpha$  is a parameter which controls how fast  $T$  should increase when  $c_l$  increases.  $\beta$  is selected so that when  $c_l$  is close to zero,  $T$  will also be close to zero. In our experiments,  $\alpha = 0.25$  and  $\beta = 12$ . Figure 4.3 shows how the function  $T$  would vary with local threshold  $c_l$ . Finally, we calculate the adaptive threshold  $t = kT$ , where  $k = 12$  is a constant.

The text candidates detected by the proposed method on some typical sample images in our database are shown in Figure 4.4. The number of identified text candidates is much smaller than that of the level lines. At the same time, most characters have been retained. Since we use an adaptive threshold selection technique, the local contrast variation in the image (as in Figure 4.4(e)) can be handled satisfactorily. Therefore, these text candidates provide us with a good starting point for future shape analysis.

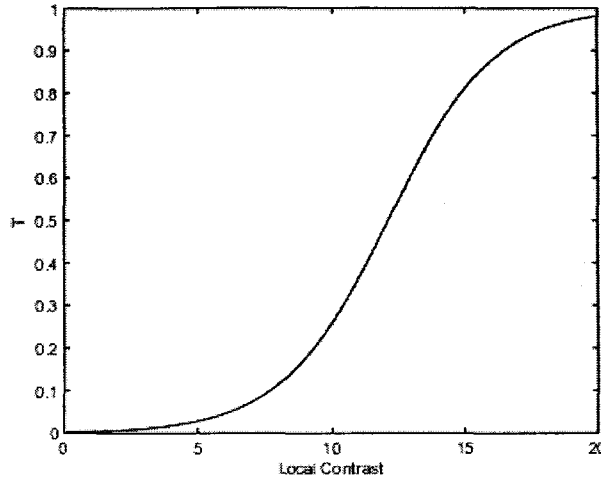


Figure 4.3: The curve that shows how the function  $T$  would vary with local threshold  $c_l$ .

### 4.3 Shape Analysis Using CNN And Sparse Representations

After text candidate identification procedure, the next task is to find the real character shapes from those text candidates. This in fact is a pattern recognition problem - classifying the text candidates into character shapes and non-character shapes. In this research, we exploit the sparse structure in the shape data and apply it to this specific shape classification problem. We adopt a well-known classification technique named Convolutional Neural Network (CNN) [66, 110]. One advantage of CNN is that it can automatically learn important features from the training samples. Furthermore, the structure of the CNN allows us to easily integrate the sparse structure learned from the data, into the training procedure and improve the performance of the classification.

In the following, we will discuss how the over-complete and sparse structure is learned from the shape data and how the CNN classifier is configured in this method.



Figure 4.4: Detection of text candidates in some scene image samples.

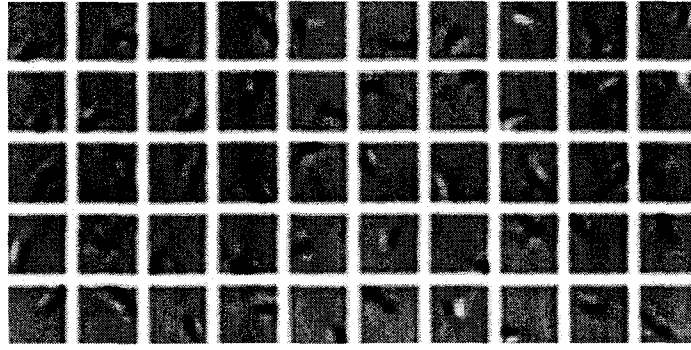


Figure 4.5: Atoms in the learned over-complete dictionary.

### 4.3.1 Learning Sparse Structure from the Data

As mentioned earlier, the sparse structure of the shape data is learned using the K-SVD algorithm[3]. To collect some shape data for sparse structure learning, we first apply the above text candidate identification procedure on the image samples in our training dataset, then extract the identified candidate shapes and save each shape boundary into a black-and-white bitmap image. From these bitmap images, we randomly select 3000 images and in each of these selected images, we sample along the shape boundary 5 by 5 image patches at a sampling step of 3 pixels. The size of the image patch is chosen to be the same as the filter kernel in the first feature map layer of the CNN, which we will discuss in the following paragraphs. In total, we have chosen 31,833 image patches. The number of atoms in the dictionary to be learned is set as  $K = 50$ , which means that the dictionary  $D$  has a redundancy factor of 2. Another parameter we need to specify is  $T_0$ , which has been chosen to be 7 according to our previous experience in handwritten numeral recognition [99]. The atoms in the learned dictionary are shown in Figure 4.5. These atoms show strong properties of locality and orientation selectivity.

### 4.3.2 The CNN Classifier

CNN is a well-known machine learning technique. Different from regular neural networks, CNN is a “deep” learning structure in that it has more layers of neurons. The first two layers of this neural network can be viewed as a trainable feature extractor, where simple features are extracted at a higher resolution, and then converted into more complex features at a coarser resolution through down-sampling. The remaining part can be viewed as a trainable classifier, usually a neural network.

The special structure of the CNN provides an ideal platform for integrating the over-complete and sparse structure learned from the shape data. Therefore, we have selected the CNN for the shape classification task. The structure of the CNN classifier used in this thesis has been shown in Figure 2.2, Chapter 2. We have considered two sets of parameters. In the first set, or the *regular* set, the network parameters are  $S_1 = 49, N_1 = 5, N_2 = 10, N_3 = 50$  and  $N_4 = 2$ . The CNN with this parameter set will be initialized randomly in the training process. In the second parameter set, the *over-complete* set, we have  $S_1 = 49, N_1 = 50, N_2 = 10, N_3 = 50$  and  $N_4 = 2$ . The CNN with this parameter set is going to be trained with the weights of the first feature map layer initialized with atoms in the learned sparse and over-complete dictionary and other weights will be initialized randomly. The details of training procedure of these two CNNs will be discussed in section 4.5.

Shape classification results of these two types of CNNs on some sample images are shown in Figure 4.6. Results generated by *regular* CNN are shown in the left column whereas results generated by *over-complete* CNN are shown in the right column. Only text shapes are shown in red. It can be observed that most non-text shapes in these samples can be effectively removed, while better performances can be achieved by the over-complete CNN. Also, note that some character shapes are not correctly classified, such as the letters ‘i’ and ‘l’ in Figure 4.6(e). The shapes of these letters

in certain fonts are difficult to differentiate since they look very similar to the shapes of short vertical lines.

## 4.4 Text Line Growing And Text Segmentation

The shape classification method proposed in the previous section efficiently eliminates those non-text shapes from the shape set generated by the text candidate identification procedure. After that, we group those text shapes into text lines using the following layout analysis rules:

- Within one text line, text shapes are roughly of the same size.
- The distance between neighboring text shapes in a text line is within a certain threshold.
- The neighboring text shapes should have a significant overlap in the y-direction.

As shown in Figure 4.6, some characters were not detected for the following reasons:

- 1) Since the local contrasts in the scene images are so complex, the adaptive threshold selection technique in the text identification procedure could not handle some extreme cases;
- 2) The CNN classifiers do not work well on certain special fonts or touched characters.

To overcome the above limitations, an extension procedure is applied to each text line by looking at those neighboring shapes that have been missed during the previous procedures. For those shapes which meet the above mentioned layout analysis rules, we put them back into the text line. Some examples of the final text detection results are shown in Figure 4.7. Images are samples in our training database. The results are generated using over-complete CNN.



Figure 4.6: Shape classification results using the proposed method. Only shapes classified as texts are shown in red. Left column: results from *regular* CNN. Right column: results from *over-complete* CNN.

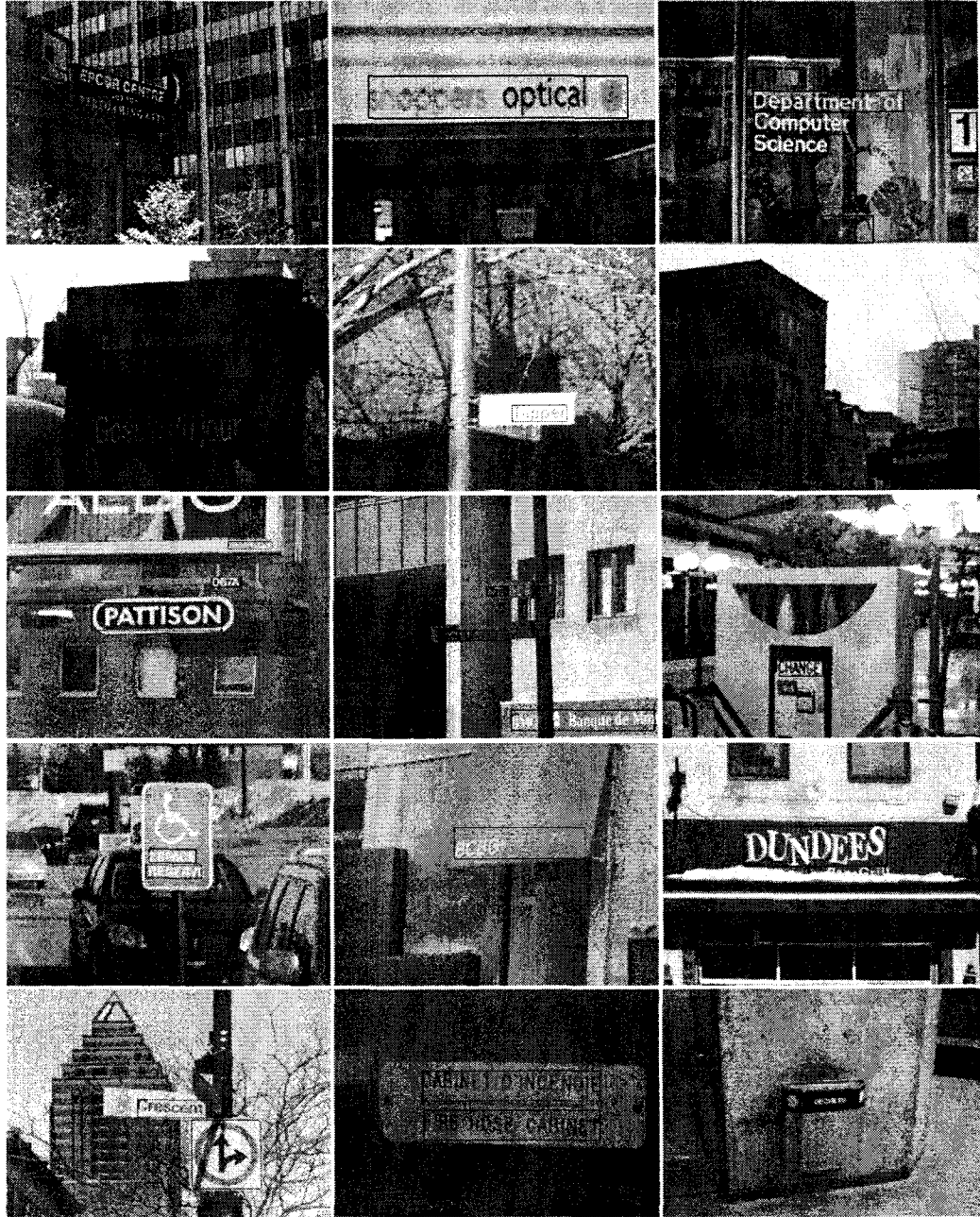


Figure 4.7: Examples of the final text detection results using the proposed method using over-complete CNN.



After text line detection, it is usually desirable to extract the characters from their backgrounds. Another advantage of using level lines is that the text segmentation task becomes straightforward: since these level lines delineate the boundaries of characters, we first sample along the character boundary using a sampling step of 5 pixels. For each sampled point on the boundary, we collect the grayscale value of the pixels in a 5 pixel by 5 pixel window centered at that point. This collection contains samples from background gray values and foreground gray values, so we use Otsu’s method [93] to find the optimal threshold and extract the character by image thresholding. Examples of the extracted texts are shown in Figure 4.8.

## 4.5 Experimental Results

In this section, first we briefly present the databases used for training and testing. Then, we describe how our CNNs are trained. Finally, evaluation results and discussion are presented.

### 4.5.1 Databases

For training purposes, we have collected 350 scene images using digital cameras. During the image capture process, the camera was set to the default automatic setting for focus, flashlight and contrast gain control. These training images have been captured on streets, in parks, subway stations and airports, including examples of homogeneous, multicolored, textured and imaged backgrounds. The texts appearing in these images vary in color, orientation, style and size. The resolution of these images ranges from  $640 \times 480$  pixels to  $2,048 \times 1,536$  pixels while text size varies from about 6 by 11 pixels to 135 by 170 pixels.

For testing purposes, we choose the 2003 ICDAR (International Conference on Document Analysis and Recognition) Text Location Contest trial test database, the



Figure 4.8: Examples of the extracted texts using the proposed method. Left column: text line images cropped from the original images. Right column: extracted characters from their backgrounds.

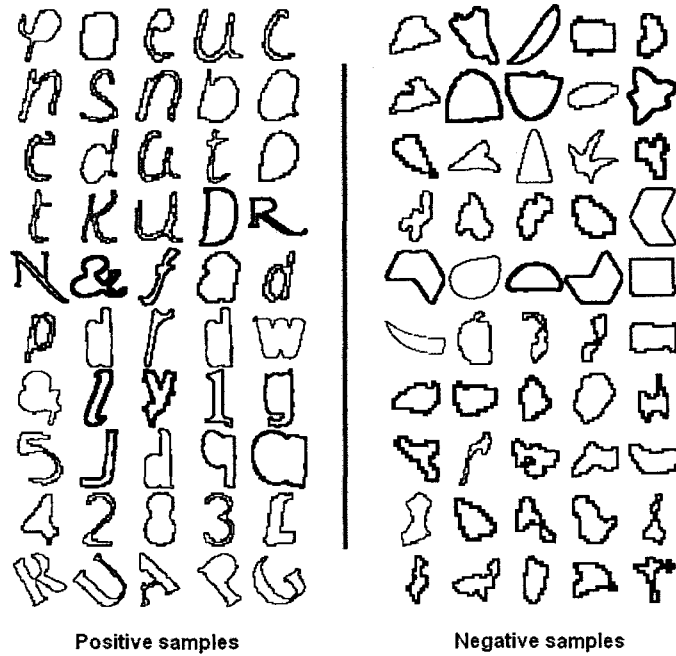


Figure 4.9: Examples of the samples used in CNN training. All samples were normalized to 47 by 47 pixels.

same public database we use in Chapter 3. Included in this database are 251 images and the ground truth of the word bounding boxes of all the target texts in these images.

#### 4.5.2 Training the CNNs

To train the CNNs for shape classification, we needed to prepare some data. We first applied the proposed text candidate identification technique to the images in our training set and manually divided those identified text candidates into text shapes and non-text shapes. In total, we collected 19,858 non-text shape samples and 19,369 text shape samples. All shapes were normalized to 47 by 47 pixels and then padded with 0 to 49 by 49 pixels before sending them to the CNN for training. Figure 4.9 shows some of these positive and negative samples. To accommodate possible distortions

that the characters in the image may have undergone during image capturing and encoding, we applied a combination of scaling transformation, rotation transformation and elastic transformation [110] to each sample before it was propagated through the CNN in each training epoch. The scaling factor was selected uniformly from  $[-0.15, 0.15]$ , with the negative scaling factor standing for shrinkage while the positive scaling factor standing for enlargement. The rotation angle was also picked uniformly from  $[-5^\circ, 5^\circ]$ , with the negative angle standing for counter-clockwise rotation and the positive angle standing for clockwise rotation.

### 4.5.3 Comparison between Regular CNN and Over-complete CNN

As shown in Figure 4.6 in section 4.3, using over-complete CNN for shape classification can achieve better results. Now let us look at the difference between these two CNNs in terms of their performance of text detection. This comparison has been performed on the test database.

To match the text detection results to the ground truth, we have adopted the algorithm in [77]. The *match*  $m_p$  between two rectangles is defined as the area of intersection divided by the area of the minimum bounding box containing both rectangles. Hence, the best match  $m(r, R)$  for a rectangle  $r$  in a set of rectangles  $R$  is defined as:

$$m(r, R) = \max m_p(r, r') | r' \in R.$$

Then, the *precision* and *recall* are defined as:

$$\begin{aligned} p' &= \frac{\sum_{r_e \in E} m(r_e, T)}{|E|} \\ r' &= \frac{\sum_{r_t \in T} m(r_t, E)}{|T|}, \end{aligned}$$

where  $T$  is the set of ground-truth rectangles and  $E$  is the set of estimated rectangles.

Table 4.1: Comparison using the test database (%).

|                   | Precision | Recall | f    | t(s) |
|-------------------|-----------|--------|------|------|
| Regular CNN       | 71.1      | 67.2   | 68.1 | 2.5  |
| Over-complete CNN | 77.2      | 66.3   | 67.5 | 7.4  |

The precision and recall measures are then combined into an  $f$  measure using the following formula:

$$f = \frac{1}{\alpha/p' + (1 - \alpha)/r'}$$

with  $\alpha$  set to 0.5.

The results of the proposed method on the test database using regular CNN and over-complete CNN are shown in Table 4.1. In this table,  $t$  stands for the average processing time in seconds per image. We can see from this table that the recall rates of the two CNNs are almost the same. The main differences between the two CNNs are the precision of the text detection and the processing time. While the over-complete CNN can detect texts with a higher accuracy, it also requires more processing time.

#### 4.5.4 Comparison with Other Methods

Even though many text detection methods have been proposed in the literature, most of them have used private databases for evaluation. In the *ICDAR* text location competitions [78, 77], several methods have been evaluated on a common database, to which we did not have access. To the best of our knowledge, our previous work [100] and two other methods [58, 57] have reported evaluation results on the same test database as the one used in this chapter. The way the *precision* and *recall* rates were calculated in these methods is different from that in [77]. In order to compare the proposed method with the methods in [58, 57], we also adopted their precision and recall measures described in the following paragraphs.

Table 4.2: Performance comparison using the same test database (%).

|  | Precision | Recall |
|--|-----------|--------|
| [58]                                   | 55.8      | 68.9   |
| [57]                                   | 56.3      | 64.3   |
| [100]                                  | 67.6      | 75.2   |
| Proposed method with regular CNN       | 70.3      | 74.1   |
| Proposed method with Over-complete CNN | 78.7      | 73.4   |

Let  $Sum$  be the total number of text regions in the ground truth,  $True$  be the number of correctly detected text regions,  $Part$  be the number of partially detected text regions, and  $False$  be the number of false alarms. Then,  $Precision$  and  $Recall$  are defined as:

$$Precision = \frac{True}{True + Part + False}$$

$$Recall = \frac{True}{Sum}.$$

The results of the proposed method compared with other methods are shown in Table 4.2. We can see that while maintaining a high text detection rate, the proposed method can achieve a significant improvement in text detection accuracy.

#### 4.5.5 Discussion

Since the proposed method makes use of the complete character boundaries for classification, it becomes advantageous in enhancing the precision of text detection when compared with other methods, which mainly use local information for feature extraction and thus are prone to false alarms. Another benefit of the proposed method is its capability to handle varying text sizes, since shapes with different sizes will be normalized to the same scale during classification. No multi-resolution analysis is needed.

As we can see in Table 4.1, the introduction of sparse representations into the shape classification procedure can further improve the precision of text detection, as

it improves shape classification accuracy. However, it does not help much in improving the text recall rate. This is natural because all text lines in an image are possibly detected during the text candidate identification process. The shape classification procedure mainly focuses on eliminating the false alarms.

Though the proposed method has made a good progress, we notice that the performance of the proposed method is still not high enough. Therefore, it is necessary to investigate the failure cases of the proposed method. In total, we identified six typical failure cases, as shown in Figure 4.10. We elaborate on them here:

- 1) *Failure due to contrast-related issues.* Even based on the topographic map representation of an image, which is invariant to global contrast variation, the proposed method still suffers from local contrast variations. First, selection of the  $\varepsilon$ -meaningful boundaries is based on a global threshold. Second, since the thresholding of the maximum monotone interval takes into consideration the local contrast variation, it is empirical in nature and is not optimal. Therefore, the proposed method could miss some target text lines where contrast between text and background is very low, as shown in Figures 4.10(a)~4.10(c). We need to point out that, due to the shading in the image, the upper part of the text line in Figure 4.10(c) is well contrasted while its lower part is poorly contrasted. This makes the situation more complicated.
- 2) *Touching characters.* As the proposed method relies on shapes that correspond to the boundaries of characters, it fails to detect touching characters. Examples of such cases are shown in Figures 4.10(d)~4.10(f). The method proposed in [22] might work better in these cases.
- 3) *Broken characters.* When characters are broken, as shown in Figure 4.10(g) and Figure 4.10(h), the proposed method can not detect them.

- 4) *Flash reflection.* A typical problem with camera-based images is the glare from flash reflections, especially on smooth surfaces like book covers (as shown in Figure 4.10(i)) and glass surfaces. The highlighted region barely contains any image information. Any text which falls into that region can not be detected by the proposed method.
- 5) *Character-like non-text shapes.* In most cases, shapes contain enough information for the CNN to classify them as text or non-text. However, there are still some ambiguous shapes that the proposed method can not handle properly. Figure 4.10(j) gives one such example where the boundaries of those dark dots are similar to that of an ‘0’ or ‘o’. Such cases are the key sources of the false detections using the proposed method.
- 6) *Special designs.* Finally, Figure 4.10(k) and Figure 4.10(l) give two examples of specially designed texts, which are difficult to be detected.

From the above identified failure cases, we can see that the text detection problem is still a very challenging one. It is an “open” problem in that, we have not imposed any limitation on the content of the image, or on the imaging conditions. Such an “openness” is the main difficulty in the text detection problem. Continuous research efforts towards this problem are needed.

## 4.6 Conclusion

A new text detection method is proposed in this chapter to find text lines in scene images. This method is based on topographic maps and sparse and over-complete representations, and it converts the text detection problem into a shape classification problem. The layout analysis technique is applied as a postprocessing stage. The benefits of this method include improved text detection accuracy, better capability of





Figure 4.10: Examples of failures of the proposed method. (a)~(c) show failures related to image contrast; (d)~(f) are examples of touching characters; (g) and (h) are examples of broken characters; (i) gives an example of the glare from flash reflections on a book cover; (j) demonstrates an example of character-like shapes that cause false alarms; (k) and (l) are examples of some special designs, which cannot be processed properly by the proposed method.

handling texts with different sizes and more robustness to varying imaging conditions. Experimental results show that a good performance has been achieved.

Future research directions include:

- a) Improving the text segmentation algorithm. Our experiments show that the adopted simple segmentation algorithm does not work well when the contrast between the background and the foreground is too small or when the color on the character boundary and the color inside the character boundary are different. More sophisticated methods are needed in these cases.
- b) Recognition of characters with projection distortions, some of these distortions might be very severe, and recognition of characters with background noise.

## Chapter 5

# Isolated Handwritten Farsi Numeral Recognition Using Sparse and Over-Complete Representations

The goal of this chapter is to show that the shape classification method, proposed in Chapter 4, can be easily applied to a classical pattern recognition problem: handwritten numeral recognition. Specifically, it will be applied on handwritten Farsi numerals. The sparse structure is represented as an over-complete dictionary which is learned from 5 pixels by 5 pixels image patches randomly selected from the Farsi numeral database. The atoms in this dictionary have been adopted again to initialize the first feature map layer of the CNN, and the CNN is then trained to do the classification task.

### 5.1 Introduction

Recognition of handwritten Latin numerals has been extensively studied in the past few decades. Many methods have been proposed with very high recognition rates [118, 64, 71, 73, 133, 110, 66, 84, 55]. Some of these techniques have been applied in real systems, such as mail sorting [115] and bank check recognition [30, 65].









|                      |   |   |   |   |
|----------------------|---|---|---|---|
| <b>Farsi Shapes:</b> |  |  |  |  |
|                      |  |  |  |  |
| <b>Latin Digit:</b>  | 2   | 3   | 4   | 6   |

Figure 5.1: Numerals in Farsi that can be legally written in different shapes.

On the other hand, research progress has been very limited towards automatic recognition of numerals written in scripts other than Latin. In this paper, we look at the recognition of handwritten numerals in one important cursive script: Farsi (Persian). Farsi is the main language used in Iran and Afghanistan, and it is spoken by more than 110 million people, including some people in Tajikistan, and Pakistan. Due to its wide usage, the problem of automatic recognition of handwritten Farsi script has attracted increasing interest in the research community.

Similar to Latin script, handwritten Farsi numerals have large variations in writing styles, sizes, and orientations. What makes the recognition of the handwritten Farsi numerals more challenging is that some of the numerals can be legally written in different shapes. Figure 5.1 demonstrates this situation, where numerals ‘2’, ‘3’, ‘4’ and ‘6’ are written in two different shapes, respectively.

In the following paragraphs, we first present related works in the problem of handwritten Farsi numeral recognition and then move on to introduce the proposed method.

## 5.2 Related Works

Several recognition techniques for handwritten Farsi characters and numerals have already been published. Different types of features have been investigated. In this section, we discuss some of these proposed algorithms, with emphasis on the feature extraction methods.

In a study on Farsi/Arabic handwritten isolated digit recognition, Soltanzadeh et al. [113] use features extracted from outer profiles, crossing counts and projection histograms at multiple orientations. Figure 5.2 demonstrates the calculation of these quantities at horizontal and vertical orientations. Since their lengths at different orientations are different, the outer profiles are normalized to a fixed length using down-sampling or up-sampling followed by a smoothing procedure. The normalized outer profiles are then used as features. The same procedure is applied to crossing counts and projection histograms. Finally, a size feature is introduced to help classify the digit ‘0’, which is usually written as a small dot. A Support Vector Machine is trained using these features for classification. A private database of handwritten Farsi numerals has been collected by the authors, which has 5000 training samples and 4000 testing samples, written by 90 writers. The best results reported in this paper is a 99.57% recognition rate with 257 extracted features, using SVM classifiers with RBF kernels.

Mowlaei et al. [88] propose a system for recognition of isolated Farsi numerals and characters. In this system, each input pattern undergoes three preprocessing procedures. First, it is aligned properly by finding the bounding box of the character. Then, the pattern is normalized to 64 by 64 pixels in size. Lastly, stroke width normalization by skeletonization is applied. The Haar wavelet is applied to decompose the preprocessed pattern into three resolution levels. The approximate image at the

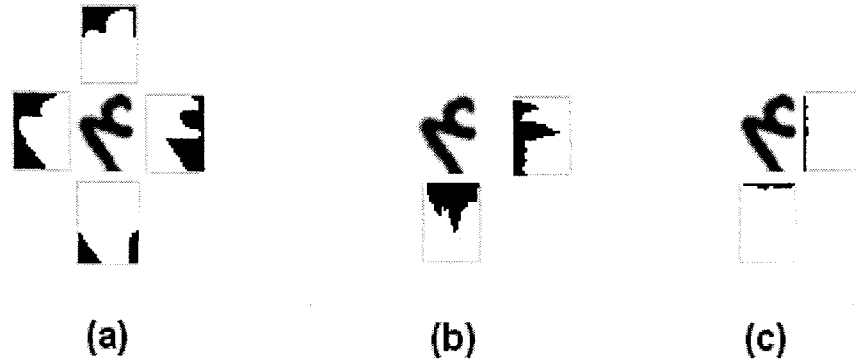


Figure 5.2: Demonstration of the feature extraction procedure in [113]: (a) Outer profiles, (b) projection histograms, and (c) crossing counts from two directions.

3rd level is converted to a feature vector after applying a low-pass filter. This feature vector is then fed to a Support Vector Machine for training. This method has been applied to recognize city names and zip numbers in handwritten postal addresses.

Ziaratban et al. [134] propose a template-based feature extraction method. In this method, twenty templates that are believed to be able to capture the most significant information from handwritten Farsi/Arabic numerals are selected heuristically. Feature extraction is carried out via the following template matching method: For each template, find the best match in an input image and record the location and the match score of the best match as two features. These features are then fed into a multi-layer perceptron for training. The proposed method has been evaluated on a private database containing 10,000 samples of handwritten Farsi numerals. These samples have been collected from 200 people of different ages and literacy levels. Six thousand of these samples have been used for training and the rest for testing. A recognition rate of 97.65% on the testing set has been achieved.

An interesting feature extraction technique, called shadow coding, has been applied in [109]. The main idea is to encode the digit in such a way that information

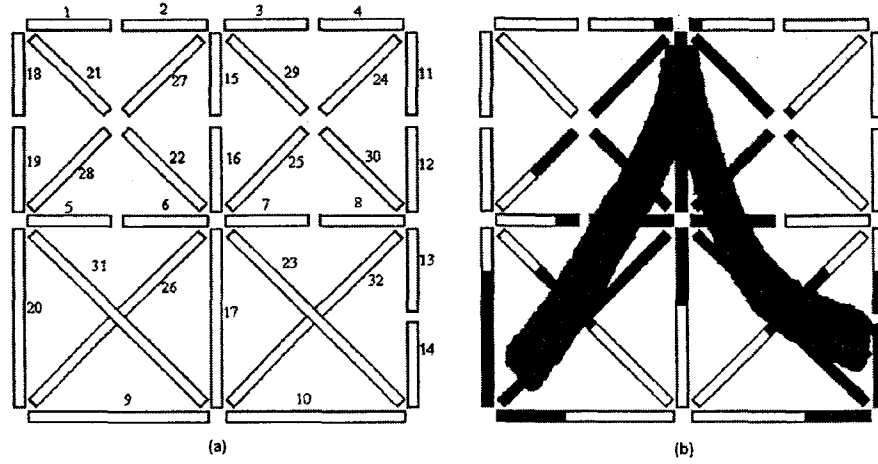


Figure 5.3: Demonstration of the shadow coding feature extraction technique: (a) Mask definition, and (b) shadow codes for a sample of the Farsi numeral ‘8’.

regarding its shape structure is embedded in the resulting code. This is achieved by defining a mask that consists of several line segments in the principal directions (e.g., vertical, horizontal, diagonal, and off-diagonal), covering the strategic locations within the image. The line segments in each principal direction are grouped into a single category. The shadow coding operation is carried out by projecting the shape in the image onto the line segments in each category. The projection is applied by assigning each shape pixel to the closest line segment in the considered category. The extracted features (shadow codes) are defined as the total number of shape pixels assigned to each line segment divided by the length of the respective line segment. A specific mask has been defined to represent the shape characteristics of the handwritten Farsi numerals, as shown in Figure 5.3 (a). An example demonstrating the calculated shadow codes for Farsi digit ‘8’ is shown in Figure 5.3 (b). The extracted shadow coding features are fed to a probabilistic neural network for training and testing. A recognition rate of 97.8% has been achieved on a private handwritten Farsi numeral database.

Another type of features, called structural features, have been applied to handwritten Farsi character recognition in [2], where a two-stage recognition process has been proposed. The first is the preprocessing stage. In this stage, a skeletonization algorithm is first applied to capture the structural relationships between the character components, which include straight line segments, loops, and dots. Then, the character skeleton is converted to a tree structure suitable for recognition. In the second stage, a set of Fuzzy Constrained Character Graph Models (FCCGMs) has been built based on those basic character components. These models are graphs, with fuzzily labeled arcs, and are used as prototypes of the character set. The recognition control applies a set of rules in sequence to match a character tree to an FCCGM. One major limitation of this method is its high time complexity.

In addition to the above mentioned Farsi script related features, moment-based features, which are script independent, are also applied to the recognition of handwritten Farsi numeral recognition. Orthogonal moments that are invariant to the basic image transformation, such as translation, scaling, and rotation, are of great importance in pattern recognition [123, 4]. In [28], Dehghan and Faez have evaluated the effectiveness of utilizing various orthogonal moments as features in the recognition of the handwritten Farsi characters. Zernike moments, pseudo Zernike moments and Legendre moments have been investigated in this evaluation. These moments are invariant to translation and scaling transformations, and are robust to noise. These moment features are used to train an unsupervised neural network, called Adaptive Resonance Network [38]. The experimental results on a database with 20 classes and 3,200 samples show that the Pseudo Zernike moments, with an order of 5, perform the best. They have achieved an error rate of 3.06%.

In the following sections, we present the proposed method and the experimental



results. Since the numeral recognition methodology is the same as the shape classification method in Chapter 4, we only explain how the CNNs are configured and how the K-SVD algorithm has been carried out in this work.

### 5.3 Configuration of the CNN

We used two sets of parameters in this work. In the first set, or the *over-complete* set, the network parameters are  $N_1 = 50$ ,  $N_2 = 50$ ,  $N_3 = 150$  and  $N_4 = 10$ . With this parameter set, we train two CNNs in our experiments for comparison, as follows:

- The first CNN is trained with the weights of the first feature map layer initialized with the learned sparse and over-complete atoms, and other weights are initialized randomly. We call this CNN the Sparse-Over-Complete (SP-OC) network.
- The second CNN is trained with all its weights initialized randomly. We call this CNN the Random-Over-Complete (RD-OC) network.

In the second parameter set, the *regular* set, we have  $N_1 = 5$ ,  $N_2 = 50$ ,  $N_3 = 100$  and  $N_4 = 10$ , which have been applied in [110]. We call this CNN architecture the *Regular* network. This network is trained with all its weights initialized randomly. The same size parameter,  $S_1 = 35$ , is used in both parameter sets.

### 5.4 Learning Over-Complete And Sparse Representations

To learn the over-complete dictionary for the Farsi handwritten numerals, we randomly selected 26,156 image patches from the CENPARMI database after preprocessing. Each of these image patches has a size of 5 by 5 pixels. The CENPARMI database and the preprocessing procedure will be discussed later in Section 5.5. The

size of each image patch has been chosen to be the same as the size of the filter kernels in the first feature map layer of the CNN. We have chosen  $K = 50$ , which means that the dictionary  $\mathbf{D}$  has a redundancy factor of 2. Another parameter we need to specify is  $T_0$ , which has been chosen empirically. We will discuss the selection of  $T_0$  in the following section.

#### 5.4.1 Determining the $T_0$ in the K-SVD algorithm

Before using the K-SVD algorithm, one has to specify the parameter  $T_0$ , which determines how sparse the representations given by the learned dictionary should be. If we use a very small  $T_0$  and ask the learned dictionary to give a very sparse representation, then the dictionary can not give faithful representations of the data. On the other hand, if we use a large  $T_0$ , we can achieve a good data representation. But we may lose some desired properties in the learned dictionary, such as locality and orientation selectivity.

In this work, the parameter  $T_0$  has been chosen experimentally. That is, several different values of  $T_0$  have been selected, and the dictionaries corresponding to these  $T_0$ s have been built using the K-SVD algorithm. Then, these learned dictionaries have been applied to initialize the weights of the first feature map layer of the CNNs to be trained. The value that gives the CNN that achieves the best performance is chosen as the parameter  $T_0$ . The data used in this procedure is the validation set of the CENPARMI Handwritten Farsi numeral database [112]. We will give more details about this database in Section 5.5.

We choose  $T_0$  to be one of the following numbers:  $\{5, 7, \dots, 17, 19\}$  and train the corresponding CNN classifier via the proposed method. The performance of these classifiers is shown in Figure 5.4, from which we can see that the best performance is achieved when  $T_0$  is set to 7 and after that, the performance of the classifier tends to

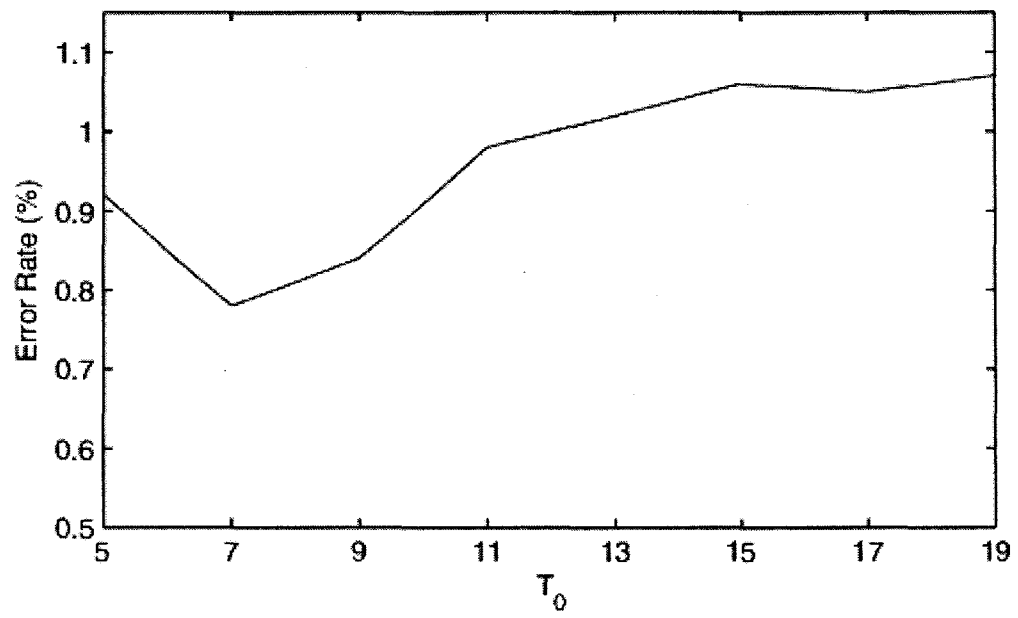


Figure 5.4: Error rate of the proposed method on the validation set of the CENPARMI Handwritten Farsi numeral database, using different values of the parameter  $T_0$ .

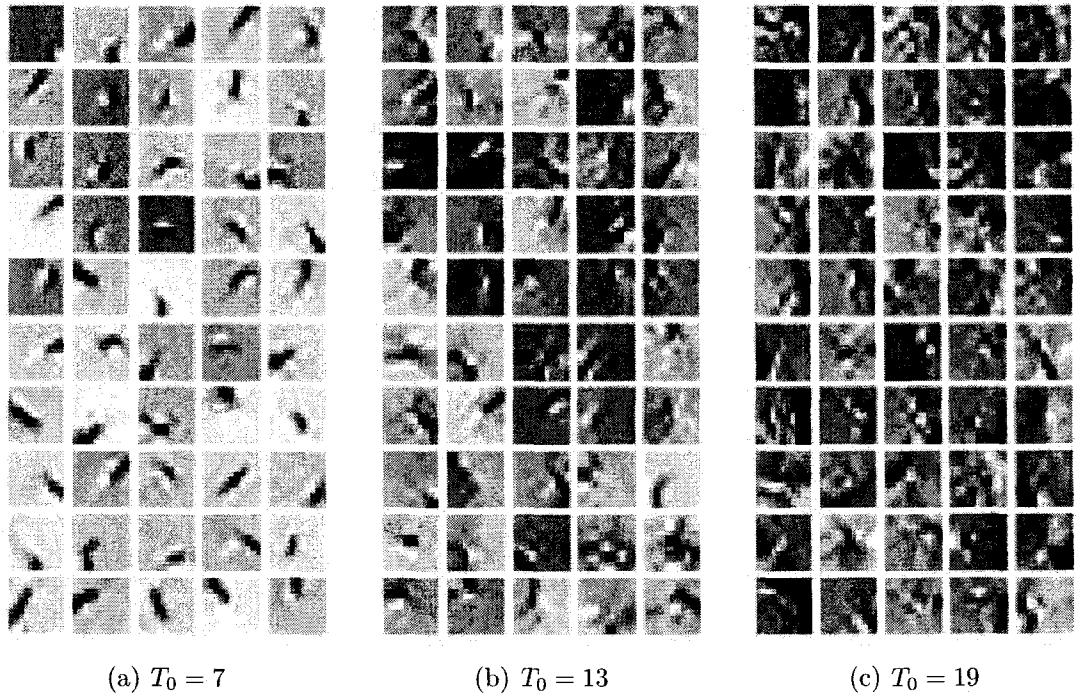


Figure 5.5: Over-complete and sparse dictionaries learned using different values of  $T_0$ .

drop while  $T_0$  increases. In Figure 5.5 we show the over-complete dictionaries learned via the K-SVD algorithm when  $T_0$  is set to 7, 13 and 19. When  $T_0 = 7$ , the atoms in the learned dictionary possess significant orientation and scale selectivity, which are the desired properties for feature extraction. However, as  $T_0$  increases, these properties are gradually lost. Therefore, we use  $T_0 = 7$  in all of our experiments.

## 5.5 Experiments

To evaluate the effectiveness of the proposed method, we have conducted experiments under different conditions. We first introduce the databases used in these experiments. Two publicly available isolated Farsi numeral databases are adopted: the CENPARMI Handwritten Farsi numeral database [112] (CENPARMI database in short), and the

HODA Handwritten Farsi numeral database [56] (HODA database in short). We briefly describe the properties of these databases and show some of their samples. We also explain the preprocessing procedures that have been applied to these samples.

As an evaluation, it would be interesting to first compare the proposed method with the “traditional” CNN, or the CNN without using over-complete and sparse representations. This evaluation procedure has been conducted using the CENPARMI database. Similar results and conclusions could be achieved on the other databases. Another interesting point is to compare the proposed method with some other well-known classification techniques. In our case, we have chosen SVM and the Modified Quadratic Discriminant Function (MQDF) classifiers for comparison. Since these classifiers require explicitly extracted features from the input pattern, we also describe the features we have considered in our experiments.

### **5.5.1 The Farsi Handwritten Numeral Databases**

The first Farsi handwritten numeral database we have used in our experiments was built at CENPARMI[112]. These samples were collected from 175 writers of different ages, education level and genders. All of these samples were scanned as 300dpi color images and then converted into grayscale images. These samples were further divided into non-overlapping training, verification and testing sets. There are 11,000 samples in the training set, with 1100 samples for each class; 2000 samples in the verification set, with 200 samples for each class and 5000 samples in the testing set, with 500 samples for each class. Some samples from this database are shown in Figure 5.6.

The second database used is the HODA database [56], which contains samples collected from both high school students and undergraduate students in Iran. The sample images were scanned at 200 dpi in 24 bit color and then converted into binary

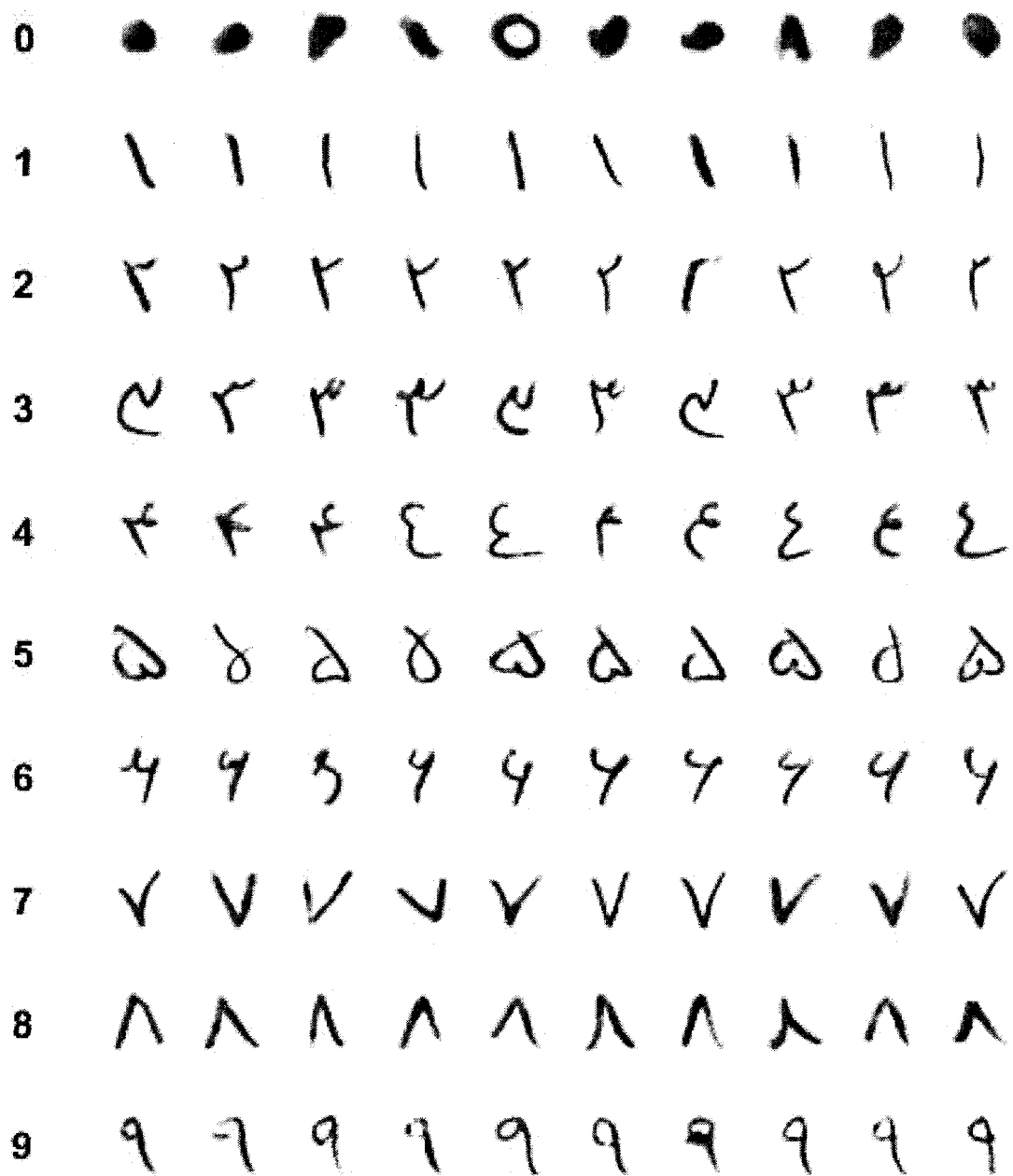


Figure 5.6: Some samples in the CENPARMI Farsi handwritten numeral database.

format. In the training set of this database, there are 6000 samples for each class and in total, there are 60,000 samples. In the testing set of this database, there are 2000 samples for each class and in total, there are 20,000 samples. Some samples from this database are shown in Figure 5.7.

Compared with the CENPARMI database, the HODA database is different in database size and scan resolution. The samples in the CENPARMI database are in grayscale images, while the samples in the Hoda database are in binarized images. The shape variations in some of the patterns (such as ‘0’, ‘2’, ‘3’, etc.) are also different.

### 5.5.2 Preprocessing

As shown in Figure 5.6, there are large variations in both the grayscale values and the sizes of the samples in the CENPARMI database. Therefore, grayscale normalization and size normalization techniques are needed for preprocessing. Since the samples in the HODA database contains are binary, we have also converted these samples into pseudo-grayscale images using a Gaussian blur before preprocessing.

For grayscale normalization, we have re-scaled the gray levels of the foreground pixels of each input image so that the scaled values would give a standard mean of 210 and a deviation of 20.

As to size normalization, we have applied the moment normalization technique [72], which first aligns the centroid of a character image with the geometric center of the normalize plan, and then re-frames the character using second-order moments. This method works better than the traditional linear normalization technique in handwritten character recognition, since it is able to reduce the position variation of the important feature points in the image. Furthermore, it can cut the tails of some elongated strokes in the pattern and thus retain most classification-related information.

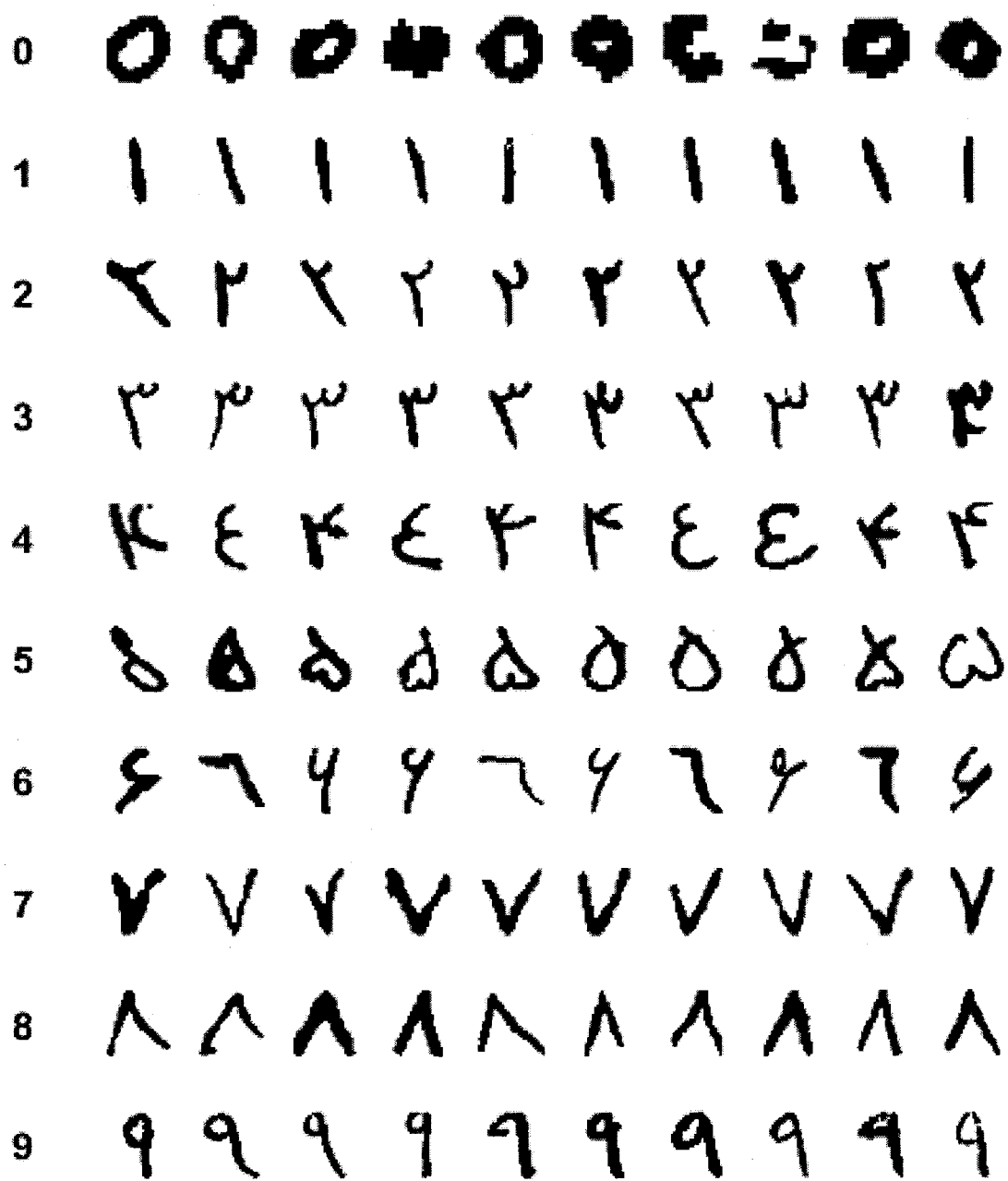


Figure 5.7: Some samples in the HODA Farsi handwritten numeral database.



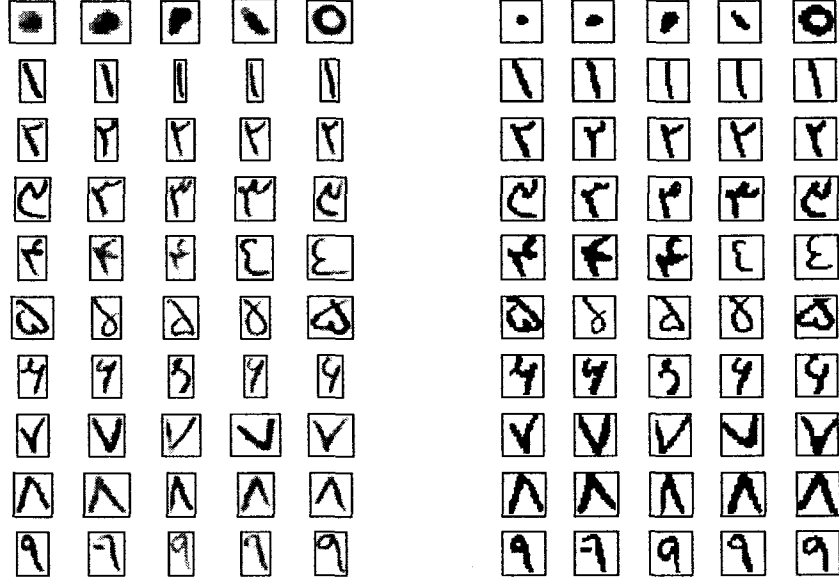


Figure 5.8: Examples of the preprocessing results from the CENPARMI Farsi handwritten numeral database: Left, original samples; Right, corresponding samples after preprocessing.

Some examples of the numerals before and after preprocessing are given in Figure 5.8. Each pattern has been normalized to a size of 35 by 35 pixels.

### 5.5.3 Comparisons between CNN-Based Methods

In this experiment, we will compare the proposed method with the regular CNN classifier. Mainly, we will investigate the impact of the following 4 conditions over the classification performance of these CNNs:

#### a) Over-complete set or regular set

This condition defines which CNN structure parameter set will be applied: the *over-complete* set or the *regular* set. These parameter sets have been discussed in Section 5.3.

**b) Using or not using sparse representations**

This condition states whether the first feature map layer of the CNN will be initialized with the learned over-complete dictionary or just initialized randomly.

**c) The amount of the data available for training**

Another interesting issue is to look at how the algorithm behaves when the amount of data available for training changes. For this purpose, we have created four training sets with different sizes using the training data in the CENPARMI database. One is the whole original training set (set-4-4) and we have built the other three training sets by randomly selecting one-fourth (set-1-4), two-fourths (set-2-4) and three-fourths (set-3-4) of the samples from the original training set.

**d) The number of training epochs**

This condition indicates how fast the learning process can be.

Since the CNN architecture has a large number of weights to be learned, the number of training samples required by this network is also very large. In order to get a better performance in terms of the capability of generalization, one usually adopted practice is to expand the training data set by introducing some distortions or transformations to the data at hand, such as affine transformations [66] and elastic distortions [110]. It is also useful to look at the effect of this data expanding technique on the proposed method.

In this experiment, we combine Simard's elastic distortions with scaling and rotation transforms [110]. The scaling factor is selected uniformly from  $[-0.15, 0.15]$ , with the negative scaling factor standing for shrinkage, while the positive scaling factor for enlargement. The rotation angle is also picked uniformly from  $[-5^\circ, 5^\circ]$ , with the negative angle standing for counterclockwise rotation and the positive angle standing

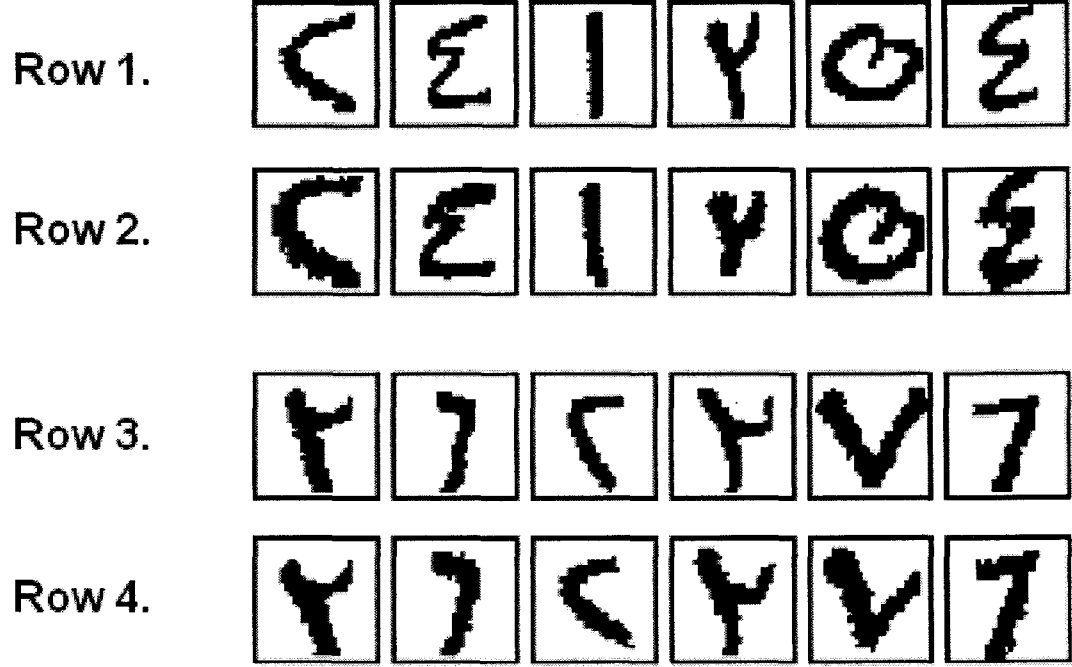


Figure 5.9: Some distorted samples. Rows 1 and 3: samples after preprocessing. Rows 2 and 4: the same samples after distortion.

for clockwise rotation. Figure 5.9 shows some examples of the distorted samples.

All the CNNs evaluated in this experiment have been trained in 90 epochs. The error rates of these CNNs in these experiments are given in Table 5.1. The configurations of these CNNs have been discussed in Section 5.3.

The results in this table show that if we go to over-complete CNN (increasing the number of the feature maps in the first feature map layer of the CNN) but do not

Table 5.1: Test error rates (%) of CNN based methods on the CENPARMI database.

|         | SP-OC     |           |             | RD-OC     |           |           | Regular   |           |           |
|---------|-----------|-----------|-------------|-----------|-----------|-----------|-----------|-----------|-----------|
|         | 30 epochs | 60 epochs | 90 epochs   | 30 epochs | 60 epochs | 90 epochs | 30 epochs | 60 epochs | 90 epochs |
| set-1-4 | 1.9       | 1.8       | <b>1.6</b>  | 2.16      | 1.90      | 1.8       | 3.12      | 2.54      | 2.3       |
| set-2-4 | 1.48      | 1.36      | <b>1.14</b> | 2.06      | 1.74      | 1.48      | 2.36      | 1.98      | 1.74      |
| set-3-4 | 1.38      | 1.24      | <b>1.02</b> | 1.72      | 1.44      | 1.36      | 2.08      | 1.66      | 1.42      |
| set-4-4 | 0.90      | 0.82      | <b>0.78</b> | 1.58      | 1.44      | 1.2       | 1.64      | 1.28      | 1.2       |

initialize the CNN with the pre-learned dictionary, then the performance of the over-complete CNN is better than the regular CNN only when the amount of data for training is small. However, when the size of the training data set increases, the difference between the RD-OC CNN and the regular CNN becomes smaller. On the other hand, if we initialize the CNN as proposed in this research and introduce sparsity into the learning procedure, significantly better performance can be achieved, even when the size of the training data set becomes larger. This demonstrates the effectiveness of the proposed method. Another interesting property shown by these results is that the proposed method converges faster than the other two CNNs in the learning process. When all the training data is applied in the training procedure, the proposed method achieves very good results after only 30 epochs.

Now we look more closely at the misrecognized numerals by the proposed method. As shown in Table 5.1, the error rate of the proposed method is 0.78%, which means that it has committed 39 errors on the test set of the CENPARMI database. These misrecognized samples are shown in Figure 5.10. To further analyze these errors, we present the confusion matrix in Table 5.2, where we can identify the following common error cases: a) '2' - '3' misclassification, b) '3' - '4' misclassification and c) '1' - '0' misclassifications. As shown in Figure 5.10, most of these misclassified samples are very similar in shape. Some are due to the defects in the samples, which might have been introduced by the writers participating in the data collection process. Possible improvement in eliminating these errors could be achieved by combining different types of classifiers. However, for some common misclassifications as in the '2' - '3' cases, it would be necessary to analyze the writing habits of Persians.

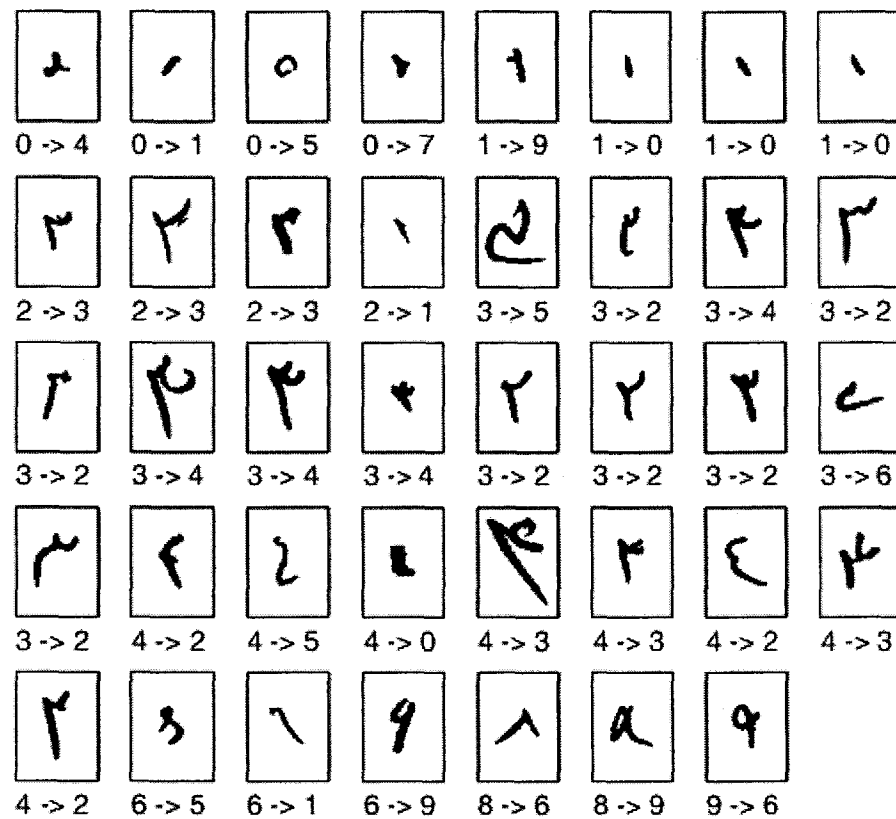


Figure 5.10: Misrecognized samples from the test set of the CENPARMI database by the proposed method.

Table 5.2: Confusion matrix of the proposed method on the CENPARMI database.

| Rec. Result<br>True Label | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   | Rec. Rate(%) |
|---------------------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|--------------|
| 0                         | 496 | 1   |     |     | 1   | 1   |     | 1   |     |     | 99.2         |
| 1                         | 3   | 496 |     |     |     |     |     |     |     | 1   | 99.2         |
| 2                         |     | 1   | 496 | 3   |     |     |     |     |     |     | 99.2         |
| 3                         |     |     | 7   | 487 | 4   | 1   | 1   |     |     |     | 97.4         |
| 4                         | 1   |     | 3   | 3   | 492 | 1   |     |     |     |     | 98.4         |
| 5                         |     |     |     |     |     | 500 |     |     |     |     | 100          |
| 6                         |     | 1   |     |     |     | 1   | 497 |     |     | 1   | 99.4         |
| 7                         |     |     |     |     |     |     |     | 500 |     |     | 100          |
| 8                         |     |     |     |     |     |     | 1   |     | 498 | 1   | 99.6         |
| 9                         |     |     |     |     |     |     | 1   |     |     | 499 | 99.8         |
| Total(errors only)        | 4   | 3   | 10  | 6   | 5   | 4   | 3   | 1   | 0   | 3   | 99.22        |

#### 5.5.4 Comparison with Other Classifiers

To further evaluate the proposed method, we will compare it with two other types of classifiers, e.g. the MQDF classifier and the SVM classifier. These comparisons are conducted on both CENPARMI Farsi handwritten numeral database [112] and the Hoda Farsi handwritten numeral database [56].

The MQDF [60] is the modification of the ordinary QDF by replacing the minor eigenvalues of each class with a constant. There are two parameters to be determined before using this classifier: the number  $m$  of the principal eigenvectors to be retained and the constant  $\delta$ , which is used to replace those small eigenvalues. Usually  $m$  is heuristically chosen as a fixed constant while  $\delta$  is set to the average of the minor eigenvalues. In our experiments,  $m$  is set to 35.

As for the SVM classifier, we have used the LIBSVM software [15]. The popular RBF kernel is chosen in our experiments. The penalty parameter  $C$  and the variance parameter  $\gamma$  of the RBF kernel have been chosen via 5-fold cross validations.

Contrary to the CNN-based methods, the above two classifiers require feature extraction from the input patterns. Two different sets of features, namely, the *gradient features* and the *profile features*, have been investigated in this work. The former

feature set is generally applied in different pattern recognition problems, while the latter is somewhat specifically designed for Farsi handwritten digits. In the following paragraphs, we first briefly describe these two sets of features, and then we show the performance of these classifiers together with that of the proposed method.

**Gradient Features** Gradient features are among the most effective features in character recognition [72, 108]. In this research, our feature extraction procedure is similar to that proposed in [72]. First, we apply the Sobel operator to calculate the gradient vector for each pixel in the input image. Then, we divide the input pattern into  $5 \times 5$  zones and calculate local orientation histograms by decomposing those gradient vectors into eight equally spaced standard directions, starting from 0 degrees. If a gradient vector lies between two standard directions, it is decomposed into two components in the two standard directions, as shown in Figure 5.11. The feature vector is built by concatenating these local orientation histograms and normalizing all the values to  $[0, 1]$ . In total, we have 200 features for each pattern.

**Outer Profile Features** These features actually include several types of features, including outer profiles, crossing counts and projection histograms of the image calculated at multiple orientations. Details on the procedure to extract these features can be found in [113].

The results of these experiments are shown in Table 5.3. There we can see that, generally speaking, MQDF does not perform as well as SVM or CNN-based methods. This is because MQDF usually works well for data whose underlying distribution is Gaussian. However, the deviation from Gaussian distribution of the handwritten Farsi numerals is significant, especially for Farsi digits like ‘2’, ‘3’, ‘4’ and ‘6’, where each pattern has more than one representative shapes. If our target is to recognize each

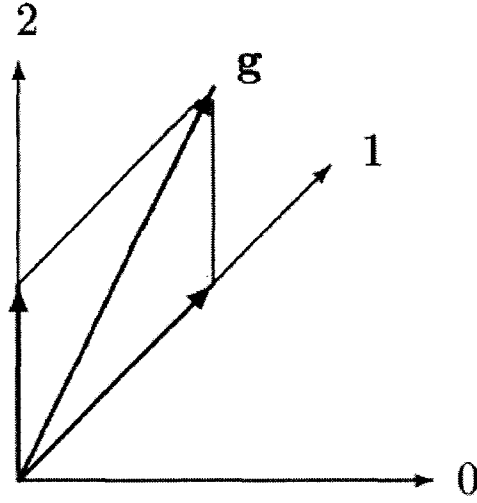


Figure 5.11: Decomposition of a given gradient direction.

Table 5.3: Test error rates (%) of the classifiers investigated in this research.

|                   | MQDF              |                  | SVM               |                  | CNN-Based |       |       |
|-------------------|-------------------|------------------|-------------------|------------------|-----------|-------|-------|
|                   | Gradient Features | Profile Features | Gradient Features | Profile Features | Regular   | RD-OC | SP-OC |
| CENPARMI Database | 2.12              | 3.18             | 1.02              | 2.68             | 1.2       | 1.2   | 0.78  |
| HODA Database     | 1.68              | 1.98             | 0.78              | 1.08             | 0.74      | 0.86  | 0.50  |

shape, instead of recognizing each pattern/digit, then we may get better results. The difference between SVM and regular CNN is insignificant, while the CNN initialized with learned sparse and over-complete representations generates the best results. All classification techniques produce better results on the HODA database, which has more data for training. Compared with the HODA database, the CENPARMI database has less data and the number of samples from different shapes are less balanced and thus it is a more challenging database.



## 5.6 Conclusion

In this chapter, we have proposed a new handwritten Farsi numeral recognition method that makes use of the over-complete and sparse structure within the data. The proposed method has been applied to two publicly available handwritten Farsi numeral databases: the CENPARMI handwritten Farsi numeral database and the HODA handwritten Farsi numeral database. We have evaluated the proposed method using data sets with different sizes and compared it with different network structures and initialization methods. Comparison between the proposed method and two other popular classifiers (SVM and MQDF) has also been made. The good experimental results have justified the benefit of exploiting the over-complete and sparse structure in the data in handwritten Farsi numeral recognition problems.

## Chapter 6

# Rotation Invariant Texture Classification by Ridgelet Transform

A new rotation invariant feature extraction method in the Ridgelet transform domain for texture classification is presented in this chapter. This feature extraction technique is an example of the global feature extraction strategy using over-complete and sparse representation. As mentioned in Chapter 2, the Ridgelet transform can be divided into two stages: the Radon transform stage and the 1-D wavelet transform stage. According to the *Projection-Slice* theorem<sup>1</sup>, the Radon transform actually provides information about the image data on a polar-grid in the frequency domain. Frequency information on a polar grid is ideal for rotation invariant feature extraction. Furthermore, by using wavelets that have compact support in the frequency domain, we can actually use the Ridgelet transform to achieve frequency-orientation decompositions for the given image data, which is similar with the multi-channel filtering technique. Such decompositions enable the proposed method to effectively capture texture properties in different frequency bands and orientations. Experimental results show that a good performance can be achieved by the proposed method.

---

<sup>1</sup>This theorem is described in Section 6.3.1

A rotational invariant feature extraction technique in the Ridgelet transform domain has been proposed in [19, 20]. This method works well for patterns like characters and shapes. However, it usually generates features with high dimensions. Given a  $64 \times 64$  character pattern, for example, it generates features with a dimension as high as 168. This makes it inappropriate for texture classification, where the size of the texture image could be even larger.

## 6.1 Related Works

Texture classification is a topic that has been investigated by many researchers during the past few decades. Research in this direction plays an important role in many applications such as remote sensing [53], document image processing [7, 75, 24, 119], medical imaging [5] and content-based image retrieval [40]. Some surveys of related research can be found in [132, 105]. Texture classification methods that are invariant under transformations, such as rotation and scaling, are of great interest. In this section, techniques for planar rotation invariant texture classification are reviewed. They are broadly divided into two categories: model-based methods and filtering-based methods.

### 6.1.1 Model-Based Methods

Many model-based methods for rotation invariant texture classification can be found in the literature. In these methods, a texture image is modeled as a probability model or as a linear combination of a set of basis functions. The coefficients of these models are used to characterize texture images. How to choose the correct model for the textures under consideration and how to estimate the coefficients of these models are the core issues for the model-based methods.

### 6.1.1.1 Simultaneous Autoregressive (SAR) Model-Based Methods

Let  $f(s)$  be the gray level value of a pixel at site  $s = (i, j)$  in an  $N \times N$  textured image,  $i, j = 1, 2, \dots, N$ . The SAR model is then defined as:

$$f(s) = u + \sum_{r \in \omega} \theta(r) f(s + r) + \varepsilon(s), \quad (6.1)$$

where  $\omega$  is the set of neighbors of the pixel at site  $s$ ,  $\varepsilon(s)$  is an independent Gaussian random variable with a zero mean and a variance of  $\sigma^2$ ,  $u$  is the bias independent of the mean gray value of the image, and  $\theta(r)$ s represent the model parameters which can be used as texture features.

The SAR model is the basis of the Circular Simultaneous Autoregressive (CSAR) model proposed in [54], where the neighborhood of a pixel is circular in order to handle rotated texture images. In CSAR, only one circle around a given pixel is considered. Mao and Jain [83] have extended the CSAR model so that several circles around a pixel can be handled. This extended model is called Rotation-Invariant SAR (RISAR) model. There are two limitations related to the application of the RISAR model. One is how to choose a proper neighborhood size in which pixels can be regarded as independent. The other is how to select an appropriate window size in which the texture is regarded as being homogenous.

To overcome these limitations, multi-resolution image representation has been introduced so that models with different neighborhood sizes and window sizes are unified into one, which is named Multi-Resolution RISAR (MR-RISAR) model in [83].

### 6.1.1.2 Markov Model

In [26], Cohen et al. have extended the 2D Gaussian Markov Random Field (GMRF) model with a likelihood function to estimate rotation and scale parameters. The

problem of this method is that the likelihood function is highly nonlinear and local maxima may exist. In addition, the algorithm must be realized by using an iterative method that is computationally intensive.

Chen and Kundu [21] have addressed rotation invariance by using multichannel sub-band decomposition and Hidden Markov Model (HMM). The proposed method can be divided into two stages. In the first stage, the Quadrature Mirror Filter (QMF) bank is used to decompose the texture image into sub-bands. In the second stage, the sub-bands are modeled in sequence using HMM, which is able to exploit the dependence of these sub-bands and capture the trend of the changes caused by the rotation. Two sets of statistical features are extracted from each sub-band. The first set consists of the third- and fourth-order central moments normalized with respect to the second-order central moments. The second set is composed of normalized entropy and energy. Since the feature vectors derived from the original texture and its rotated version through the QMF bank are obviously different, the proposed method implicitly achieves rotational invariance via the learning capability of the HMM. As the number of texture classes grows, the rate of the correct texture classification of this method would decrease.

To better handle the variations in the feature values introduced by image rotation, Wu and Wei [130] have further developed the above idea by re-sampling the texture image along a spiral contour and converting it into a 1-D signal. Then, the QMF decomposition and HMM are applied to the 1-D signal and a better performance can be achieved.

### **6.1.2 Filtering-Based Methods**

In this section, we mainly introduce two types of filtering-based rotational invariant texture classification methods: Gabor multi-channel filtering and wavelet transforms.

A comprehensive review of the filtering-based texture classification methods can be found in [105].

#### **6.1.2.1 Gabor Multi-Channel Filtering**

It is now widely accepted that the processing of pictorial information in the Human Visual System (HVS), and the visual cortex in particular, involves a set of parallel and quasi-independent mechanisms or channels [126]; each of which is tuned to a specific narrow band of spatial frequency and orientation. These properties with respect to the response of the visual cortex of mammals can be modeled satisfactorily using Gabor functions [85].

Gabor filters are complex sinusoidal gratings modulated by 2-D Gaussian functions in the space domain, and shifted Gaussian functions in the frequency domain. They can be configured to have various shapes, bandwidths, orientations and center frequencies. Furthermore, Gabor functions have been shown to achieve the optimum space-frequency localization [25]. These results have laid the foundation for lots of work using Gabor multi-channel filtering in texture classification, where a bank of Gabor filters are applied to extract features from texture images [45, 41, 102, 119]. A commonly adopted Gabor filter bank configuration is shown in Figure 6.1. Comparative studies show that Gabor filtering usually performs well [105]. However, how to appropriately configure the filter bank in applications is very tricky.

Another limitation of the Gabor multi-channel filtering is that, when rotational invariancy is required, sampling in orientation has to be dense enough at each central frequency. This means that many Gabor filtering operations have to be performed in the feature extraction process. To speed up this process, a steerable approximation technique has been proposed in [97]. The idea is to find, for each central frequency, a basis with a smaller number of functions than that of the filters in the Gabor filter

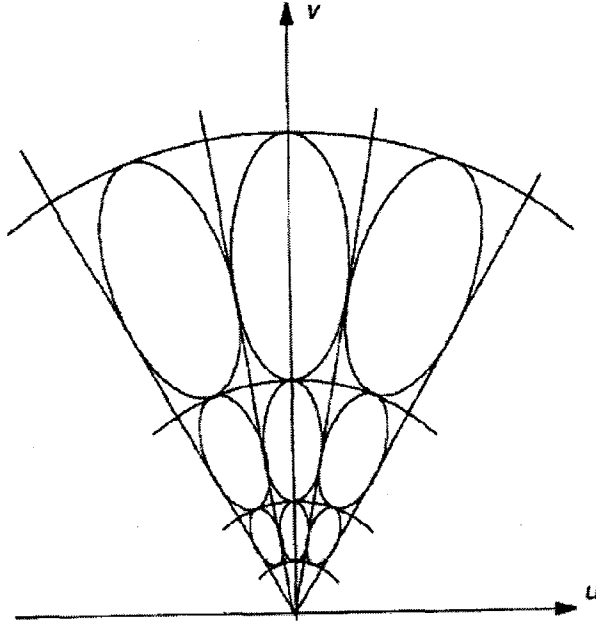


Figure 6.1: A commonly adopted Gabor filter bank configuration.

bank, where the linear combinations of the functions in the basis provide least-square optimal approximations to those Gabor filters. Filtering via the Gabor filters can then be achieved by filtering using a small number of basis functions followed by a linear combination. The optimal approximation is achieved using the Singular Value Decomposition (SVD) algorithm. Suppose the matrix  $G$  represents the  $p$  digitized Gabor filters with the same central frequency, with one column corresponding to one filter, then the SVD gives:

$$G = [f_1, f_2, \dots, f_p] = USV^T = UW. \quad (6.2)$$

The columns of  $U$  and the rows of  $W$  give the basis functions and the coefficients for representing the Gabor filters, respectively.  $S$  is a diagonal matrix of non-negative singular values, in decreasing order of magnitude. These singular values play a role of weighting in this representation. Those basis functions corresponding to the small

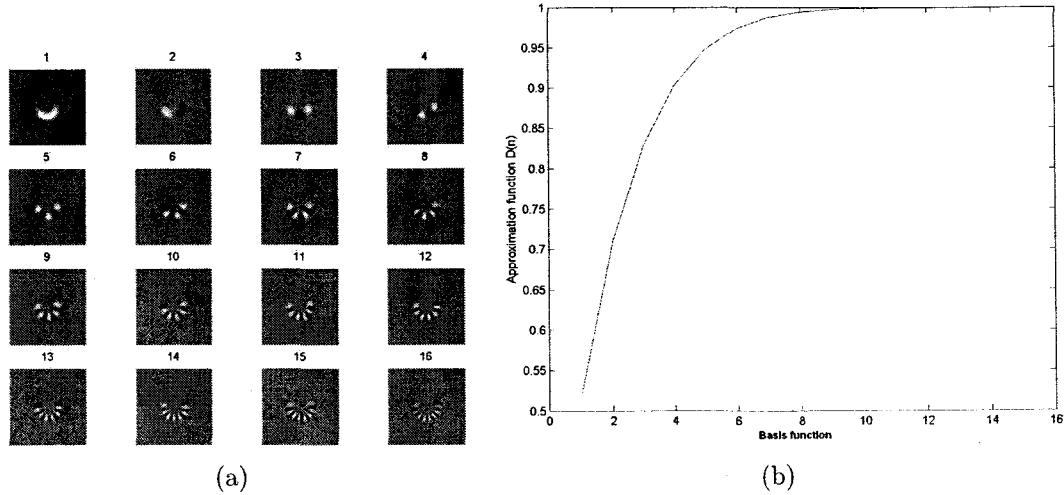


Figure 6.2: (a) Basis functions given by SVD decomposition. (b) Normalized accumulative sum of the singular values.

singular values get a small weight and do not make a significant contribution to the representation. Therefore, least-square optimal approximation can be achieved by just ignoring those insignificant basis functions. Figure 6.2 shows an example of the SVD decomposition results. In this example, a band of 16 isotropic Gabor filters with a central frequency set to 16 cycles/image have been digitized and the SVM decomposition is applied. Figure 6.2 (a) gives the basis functions and Figure 6.2 (b) shows the normalized accumulative sum of the corresponding singular values. It is clear that most contributions in the linear representation come from the leading eight basis functions, while the other basis functions can be ignored. Experimental results show that up to 40% of computations can be eliminated compared with the traditional Gabor multi-channel filtering method. In the meantime, almost the same high texture classification correct rate can be achieved.

#### 6.1.2.2 Wavelet Decompositions

Wavelet transformations also play an important role in texture classification. Methods using Discrete Wavelet Transform (DWT) have been pioneered by Mallard [81],



where texture images are decomposed into dyadic sub-band structures. Later, the work by Chang and Kuo [16] indicates that texture features are most prevalent in intermediate frequency bands, thus the octave band decomposition is not optimal. Instead, they have proposed a discrete wavelet packet transform, which achieves a better performance.

The DWT and the discrete wavelet packet transform are critically sampled multi-rate filter banks. However, critically sampled filter banks typically are not translation invariant, as mentioned in the Introduction chapter, and they imply inaccurate texture edge localization [105]. To alleviate this problem, an over-complete wavelet representation, called the wavelet frame, has been proposed [125].

Although significant progress has been made, these methods are unsatisfactory when rotation invariant features are needed. This is because textures have different frequency components along different orientations, while the ordinary wavelet transform has a limited orientation selective capability. For example, 2D DWT only distinguishes three directions (horizontal, vertical and diagonal) while 2D complex wavelet [61] distinguishes only six directions. Many attempts have been made to overcome this disadvantage. In [17] and [33], the authors exploited the steerability of certain wavelets to calculate the wavelet transform at different orientations to extract rotation invariant features. Meanwhile, image transforms, such as the log-polar transform [103] and the Radon transform [50], were applied at the preprocessing stage to convert image rotation into circular shifting of the transformed data. Afterwards, a shift-invariant wavelet transform was applied to extract rotation invariant features. Different from the above methods, the authors in [49] applied the Radon transform to estimate the principal direction of the texture image and then applied the wavelet transform along that direction for feature extraction.

## 6.2 Wavelet Selection In Ridgelet Transformation

In the following two sections, we introduce the proposed rotational invariant texture classification algorithm using the Ridgelet transform. As seen in Chapter 2.1, the 1-D wavelet transform is an important building block of the Ridgelet transform. In the 1-D wavelet transform stage, it is very important to choose the appropriate wavelet transform for feature extraction in the applications. In texture classification, we expect the selected wavelet transform to have a good localization in the frequency domain, so that features can be extracted from different frequency sub-bands. This makes the common time domain compactly-supported wavelets inappropriate here. Therefore, frequency B-spline wavelets [121] have been chosen in this research.

The frequency B-spline wavelets are defined in the frequency domain on a compact frequency interval of support described in terms of a desired center frequency  $f_c$  and a desired bandwidth  $f_b$ . These wavelets consist of an entire family of valid analyzing wavelets, indexed by an integer order parameter  $m$ . If  $m = 1$ , then the frequency B-spline wavelet is the Shannon wavelet.

Let functions  $\theta_m(t)$  be defined as:

$$\theta_m(t) = \left( \frac{\sin \pi t}{\pi t} \right)^m, m = 1, 2, 3, \dots \quad (6.3)$$

In the frequency domain, this corresponds to:

$$\hat{\theta}_m = \underbrace{1_{[-1/2, 1/2)} * 1_{[-1/2, 1/2)} \cdots * 1_{[-1/2, 1/2)}}_{m \text{ times}}. \quad (6.4)$$

Here, the symbol ‘\*’ stands for convolution and  $1_{[-1/2, 1/2)}$  is the indicator function of the interval  $[-1/2, 1/2)$ .

Then, the frequency B-spline wavelet is defined in the frequency domain interval  $(f_c - f_b/2, f_c + f_b/2]$  through the translation and dilation of  $\hat{\theta}_m$  as:

$$\hat{g}_m = \tau_{f_c} D_{mf_b^{-1}} \hat{\theta}_m. \quad (6.5)$$

Here,  $\tau_\alpha$  stands for the operator of  $\alpha$  translation, and  $D_\alpha$  stands for the operator of  $\alpha$  dilation.

Taking the inverse Fourier transform gives:

$$g_m(t) = (f_b/m)^{(1/2-m)} e^{j2\pi f_c t} \left( \frac{\sin(\frac{\pi f_b t}{m})}{\pi t} \right)^m. \quad (6.6)$$

More properties about frequency B-spline wavelets can be found in [121]. Figure 6.3 shows one third-order B-spline wavelet in the time and frequency domain, with  $f_c = 0.256$  radian per sample and  $f_b = 0.512$  radian per sample. We can see that this wavelet is a band-pass filter in the frequency domain. In this research, we use six third-order real frequency B-spline wavelets with the following parameters:  $f_c^1 = 0.256, f_b^1 = 0.512; f_c^2 = 0.512, f_b^2 = 0.768; f_c^3 = 0.768, f_b^3 = 1.024; f_c^4 = 1.024, f_b^4 = 1.28; f_c^5 = 1.28, f_b^5 = 1.537; f_c^6 = 1.537, f_b^6 = 1.793$ , respectively. These parameters are measured in radians per sample. They are chosen empirically (based on the experiments on the training set built from Bradatz textures) to have linear increments, a strategy different from the Gabor transform-based method [41], where a dyadic frequency decomposition was chosen.

### 6.3 Rotation Invariant Texture Feature Extraction in the Ridgelet Domain

In this section, we show that the Radon transform stage in the Ridgelet transform actually enables us to calculate the 2-D polar Fourier transform (Fourier transform on polar grids, instead of on traditional Cartesian grids), which is more appropriate for rotation invariant feature extraction. This is possible due to the well-known *Projection-Slice* theorem. We also give a simple and effective rotation invariant texture feature extraction method using Ridgelet transform coefficients.

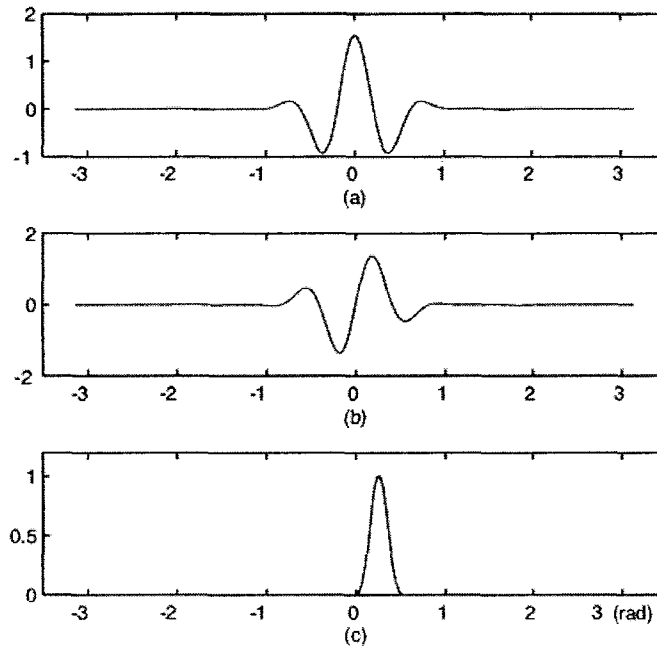


Figure 6.3: Plot of a B-spline wavelet in the time and frequency domain with  $f_c = 0.256$  and  $f_b = 0.512$  radians per sample: (a) The real part of the wavelet in the time domain, (b) the imaginary part of the wavelet in the time domain, and (c) the Fourier spectrum of this wavelet (properly scaled).

### 6.3.1 Calculating the Polar Fourier Transform from Radon Transform

We start this section by introducing the well-known *Projection-Slice* theorem, which has been commonly used in image reconstruction from projection methods [46]. Basically, the theorem states that the one-dimensional Fourier transform of a projection of a function  $f(\mathbf{x})$  is a “slice” through the 2-dimensional Fourier transform of this function. More specifically, let  $F_f(\omega)$  be the 2-dimensional Fourier transform of  $f(\mathbf{x})$ , then:

$$F_f(\xi \cos \theta, \xi \sin \theta) = \int_{\mathbb{R}} e^{-j\xi t} R_f(\theta, t) dt \quad (6.7)$$

This theorem gives us a way to calculate the polar-Fourier transform of a given image using the Radon transform. Let  $I(m, n)$  be an image with size  $N$  by  $N$  pixels. We choose a disk area within the image, as shown in Figure 6.4. Then, the Radon transform is applied to the selected area, followed by 1-D Fourier transform applied to each projection. The final result consists of the discrete Fourier transform of the given image on the polar grids. This polar Fourier transform is ideal for rotation invariant feature extraction, since rotation in the space domain corresponds to rotation in the frequency domain.

### 6.3.2 Rotation Invariant Feature Extraction

Given a square texture image of size  $N \times N$ , we first select a disc region at the center of the image, as shown in Figure 6.4. We then apply the Radon transform on the selected area with equal spaced projections (0.75 degrees apart from each other) within the orientation interval  $[0^0, 180^0)$ . The 1-D DFT is then applied on each projection to convert the Radon transform coefficients into the 2-D polar Fourier domain. The 1-D frequency B-spline wavelets given in the previous section are applied to decompose each projection into six sub-bands. In the end, we get the Ridgelet

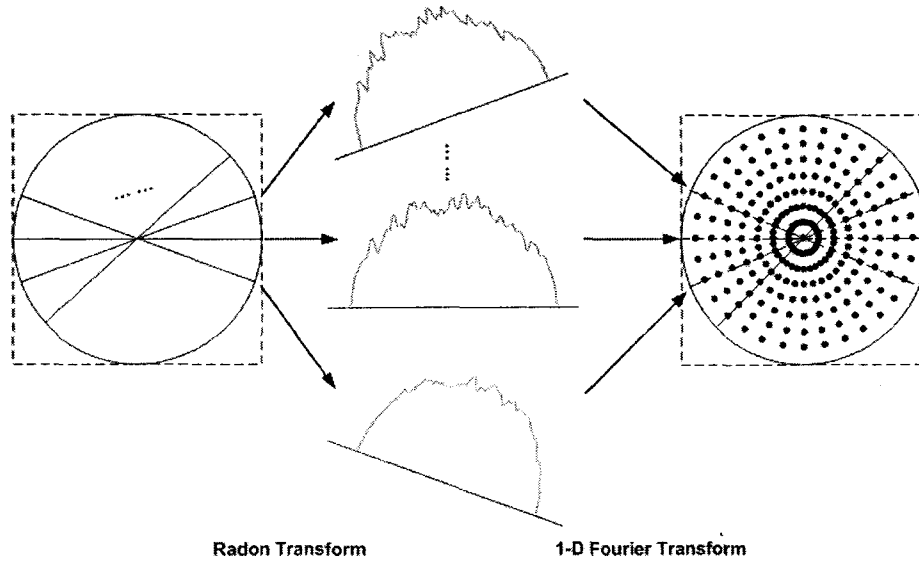


Figure 6.4: Calculation of the polar Fourier transform using the Radon transform.

transform coefficients in six  $M \times L$  matrices, with row index  $M$  indicating each projection in the Radon transform. The column index  $L$  stands for the length of each projection, whose value depends on how the Radon transform is implemented. It is easy to see that the rotation of the given image would result in a circular shift along  $M$  in each matrix.

For each sub-band of the Ridgelet transform coefficients in a matrix, features are extracted as follows:

1. Group the Ridgelet transform coefficients across orientations by equally dividing the  $M$  rows in the matrix into 16 groups. For example, group 1 contains rows 1 to 15 of the given matrix, group 2 contains rows 16 to 30 of the given matrix, and so on.
2. From each group of coefficients, calculate the first and the second order moments of the amplitudes and store them cumulatively into two arrays,  $m$  and  $s$ ,

respectively:

$$m(i) = \frac{1}{N_i} \sum_{e \in \text{group } i} |e| \quad (6.8)$$

$$s(i) = \left( \frac{1}{N_i} \sum_{e \in \text{group } i} (e - m(i))^2 \right)^{\frac{1}{2}} \quad (6.9)$$

with  $N_i$  being the number of coefficients in group  $i$ .

3. Apply the 1-D DFT to array  $m$ . The amplitudes of the most significant Fourier coefficients are selected as features. In our case, only the amplitudes of the first five most significant coefficients are selected. The same procedure is applied to  $s$ . Since the amplitudes of the 1-D DFT coefficients are shift invariant, the extracted features are rotation invariant.

Here, the grouping of the columns of the coefficient matrix is equivalent to the sampling of the orientation variable. It reduces the amount of data to be processed and the amount of features to be extracted. We have chosen a total of 16 groups/sample in order to ensure the rotation invariance of the extracted features.

### 6.3.3 Relation of the Proposed Method to Multi-Channel Filtering

Looking at the Fourier domain, we can see that the proposed method decomposes the image data into parts. Each has a different frequency and orientation selectivity. Using the Radon transform and the grouping method in the feature extraction procedure, we split the data into different orientations. Then, the wavelet transform allows us to divide the data into different frequency sub-bands. These procedures are depicted in Figure 6.5. Therefore, the proposed method is closely related to multi-channel filtering.

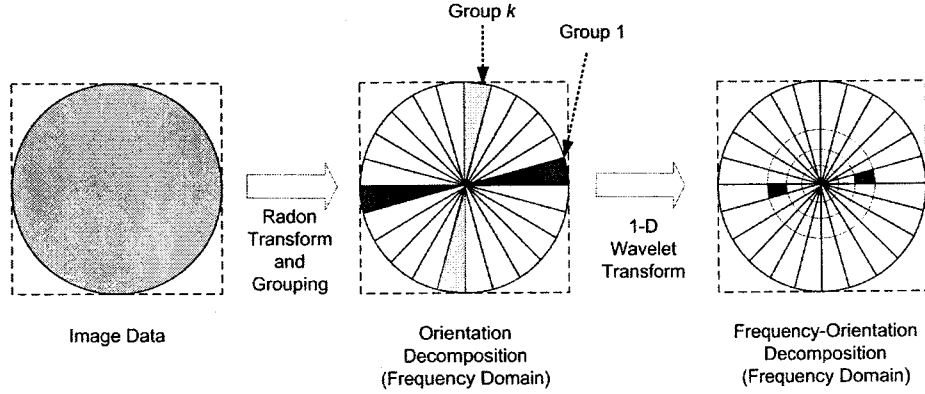


Figure 6.5: Demonstration of the frequency-orientation decomposition achieved by the proposed method.

## 6.4 Experiments and Discussion

### 6.4.1 Experimental Results

To evaluate the performance of the proposed method, different data sets have been used. We also compare the proposed method with Jafari-Khouzani’s method [49] (Method I) and the Gabor multichannel filtering method [41] (Method II).

We use three data sets to demonstrate the effectiveness of the proposed method. The details about how to generate these data sets will be described below. In order to make the textures nondiscriminable for the local mean gray level or local variance, the images in these data sets have been separately histogram equalized prior to being used, as in [105]. We note that in many papers, this factor has been ignored.

For all of these experiments, a  $k$ -nearest neighbor classifier has been used for feature classification. Three different values of  $k$  are considered:  $k = 1, 3$ , and  $5$ . The features are normalized before classification using  $\hat{f}_{i,j} = (f_{i,j} - \mu_j)/\sigma_j$ , where  $f_{i,j}$  is the  $j$ th feature of the  $i$ th image, and  $\mu_j$  and  $\sigma_j$  are the mean and variance of the  $j$ th feature, respectively, in the training set.

Data set 1 consists of 60 texture images of  $640 \times 640$  pixels from the Brodatz



album (the same texture images as in data set 3 of [49]). Each image has been divided into 25 non-overlapping sub-images of  $128 \times 128$  pixels to create a training set of 1500 ( $60 \times 25$ ) images. To create the test set, each original texture image has been rotated to 16 orientations (10, 20, 30, ..., 150, 160 degrees, respectively). From the center of each rotated image, four non-overlapping sub-images have been extracted. Therefore, a total of 3840 ( $60 \times 16 \times 4$ ) test images have been obtained.

To show the discrimination capability of the proposed features, we first show, in Figure 6.6, the clusters corresponding to the ten textures (D15, D20, D23, D34, D37, D46, D57, D81, D87, and D93) using the samples in the training set of data set 1 and the two features with the highest discrimination power as x and y axes. We can see that, with these textures, a good discrimination capability can be achieved using only two features. When more textures are added, more features are needed to get a good classification performance. All classification results are presented in Table 6.1.

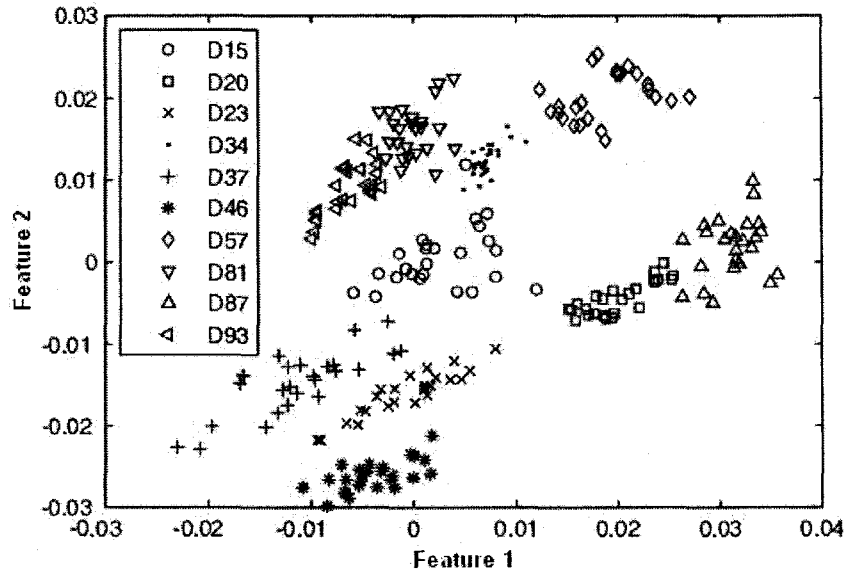
Data set 2 consists of the 24 texture images used in [90]. Each image has been captured at nine rotation angles (0, 5, 10, 15, 30, 45, 60, 75, and 90 degrees). Each image is of  $538 \times 716$  pixels. Twenty non-overlapping sub-images of  $128 \times 128$  pixels have been extracted from each image. All of the extracted images with zero degrees of rotation have been used for training and the rest of the images have been used for testing. Therefore, there are 480 ( $24 \times 20$ ) training samples and 3840 ( $24 \times 20 \times 8$ ) testing samples in this data set. The classification results are presented in Table 6.2.

Table 6.1: The Correct Classification Rate for Data Set 1 Using Different Feature Extraction Methods and Different  $k$  Values for  $k$ -NN Classifier (%).

|                  | $k =$ | 1     | 3     | 5     | 7     | 9            |
|------------------|-------|-------|-------|-------|-------|--------------|
| Method I:        | db4   | 93.65 | 94.48 | 94.61 | 94.22 | 94.30        |
|                  | db6   | 92.66 | 94.17 | 94.45 | 94.22 | 94.19        |
| Method II:       |       | 92.79 | 92.79 | 93.05 | 93.15 | 93.23        |
| Proposed Method: |       | 94.71 | 95.36 | 95.73 | 95.91 | <b>95.94</b> |



(a)



(b)

Figure 6.6: Demonstration of the discrimination capability of the proposed features: (a) Ten texture samples from the Brodatz texture album (from left to right and top to bottom: D15, D20, D23, D34, D37, D46, D57, D81, D87, and D93), and (b) the clusters corresponding to the 10 textures in the training set. Here, we use the two features with the highest discrimination power as x and y axes.

Table 6.2: The Correct Classification Rate for Data Set 2 Using Different Feature Extraction Methods and Different  $k$  Values for  $k$ -NN Classifier (%).

|                  | $k =$ | 1            | 3     | 5     | 7     | 9     |
|------------------|-------|--------------|-------|-------|-------|-------|
| Method I:        | db4   | 97.14        | 95.63 | 93.78 | 92.55 | 91.43 |
|                  | db6   | 97.34        | 96.20 | 94.77 | 93.65 | 91.75 |
| Method II:       |       | 94.69        | 94.51 | 94.56 | 94.11 | 93.67 |
| Proposed Method: |       | <b>99.30</b> | 99.04 | 98.91 | 98.78 | 98.33 |

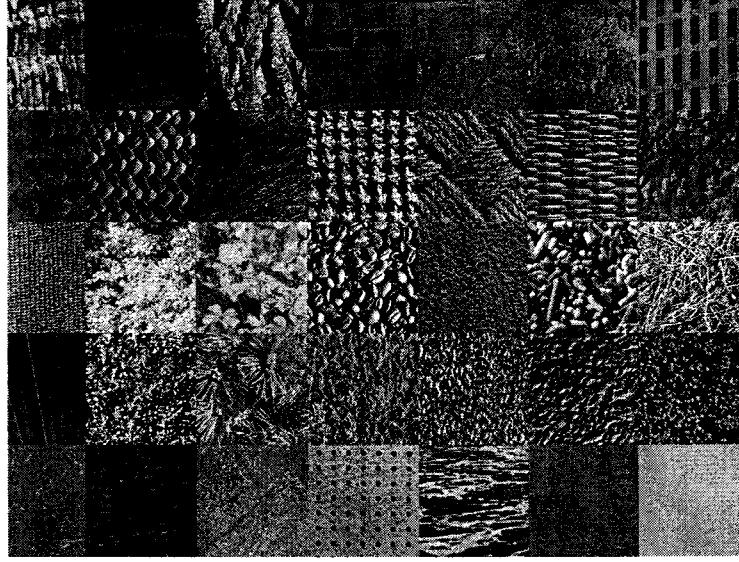


Figure 6.7: The 35 Texture Samples from the VisTex Database[127].

Table 6.3: The Correct Classification Rate for Data Set 3 Using Different Feature Extraction Methods and Different  $k$  Values for  $k$ -NN Classifier (%).

|                  | $k =$ | 1            | 3     | 5     | 7     | 9     |
|------------------|-------|--------------|-------|-------|-------|-------|
| Method I:        | db4   | 95.63        | 95.58 | 94.78 | 94.87 | 94.20 |
|                  | db6   | 95.49        | 95.36 | 94.87 | 94.24 | 94.38 |
| Method II:       |       | 93.44        | 93.44 | 93.57 | 93.26 | 93.66 |
| Proposed Method: |       | <b>97.50</b> | 97.19 | 96.88 | 96.56 | 96.16 |

Data set 3 consists of 35 textures selected from the VisTex database[127] , which are displayed in Figure 6.7. These samples are images taken from different natural scenes. Each image has  $512 \times 512$  pixels. Only gray-scale levels of the images have been used in the experiments. Each original image has been divided into 16 non-overlapping sub-images. Each sub-image has  $128 \times 128$  pixels. Thus, a training database of 560 ( $35 \times 16$ ) images has been built. The testing data set was built in the same way as data set 1. Therefore, there are 2240 ( $35 \times 16 \times 4$ ) testing images in this data set. The classification results are presented in Table 6.3.

From these results, it is evident that a better classification performance can be achieved by the proposed method. Detailed discussion with respect to these results will be given in the following section.

### 6.4.2 Discussion

Our discussion is divided into two parts: the effects of the histogram equalization and the error analysis of the proposed method.

**Effects of histogram equalization:** It is worthwhile to investigate the effects of histogram equalization on texture images. Histogram equalization is a well-known non-linear operation that generates an approximately uniform distribution of gray-levels over the available range in an image. It helps to make the textures non-discriminable for the local gray level mean or local gray level variance and allows a more precise evaluation of the capabilities of the texture classification methods.

On the other hand, we point out two issues related to the histogram equalization technique. First, it tends to introduce high frequency components into an image. We will use a synthesized image to illustrate this point. Consider the simple synthetic textured image in Figure 6.8a. It consists of one texture generated by a sinusoid. Figure 6.8b shows the same texture after histogram equalization. For illustration purposes, consider one horizontal scan line through each of the two images, as shown in Figure 6.8c and Figure 6.8d, respectively. We can see that the scan line in the histogram equalized image is much sharper than the one in the image without the histogram equalization. Therefore, some high frequency components are introduced in a histogram equalized image. The second issue is that histogram equalization tends to amplify the background noise in the image, as shown in Figures 6.8e and 6.8f. This phenomenon is more frequently observed

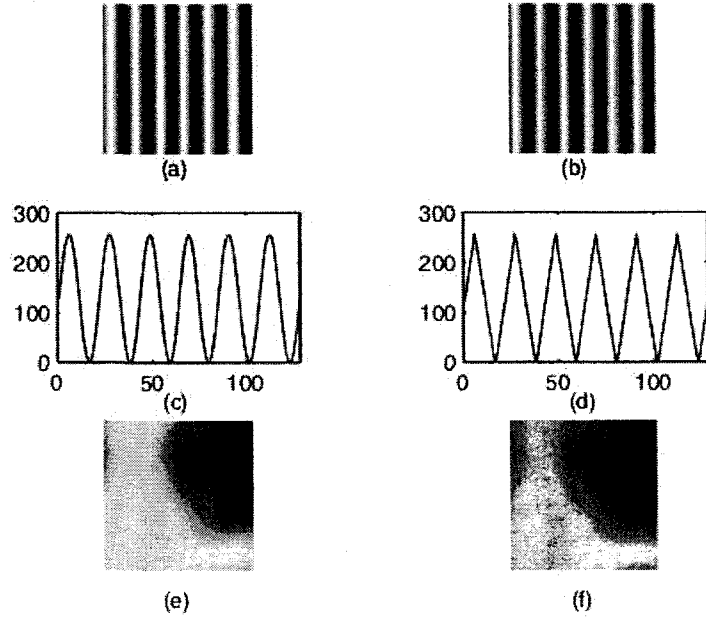


Figure 6.8: Demonstration of the two issues related to histogram equalization: (a) A synthetic texture generated by a sinusoid, (b) the same texture after histogram equalization, (c) a horizontal scan line in texture image (a), (d) a horizontal scan line in texture image (b), (e) texture D48 from Brodatz texture data set, and (f) histogram equalized version of (e), where background noise has been greatly amplified.

in the Brodatz texture data set. In the Outex texture data set, however, the noise amplification effect is not so significant. Since the proposed method works in the Radon domain (where the additive Gaussian white noise with a zero mean tends to be canceled), the noise amplification phenomenon is not so critical.

**Error analysis of the proposed method:** We have performed a detailed error analysis for the proposed method on texture data set 1 only, since the proposed method achieves the lowest performance on this data set as compared with data sets 2 and 3. Actually, the same phenomenon can be observed for all three methods investigated in this research. Here, we use the results generated by the

Table 6.4: The confusion matrix of the proposed method on the Brodatz data set using the 1-NN Classifier. Textures with classification rates higher than 75% are not shown in this table.

|      | D110 | D23 | D27 | D5 | D74 | D98 | D9 |
|------|------|-----|-----|----|-----|-----|----|
| D110 | 31   | 0   | 0   | 0  | 0   | 0   | 33 |
| D23  | 0    | 25  | 13  | 0  | 10  | 16  | 0  |
| D27  | 0    | 15  | 23  | 0  | 24  | 2   | 0  |
| D74  | 0    | 4   | 16  | 1  | 39  | 4   | 0  |
| D98  | 0    | 11  | 11  | 0  | 0   | 42  | 0  |

1-NN classifier for the error analysis.

To begin with, we have singled out those textures with the correct classification rates lower than 75%. The confusion matrix for this data set is shown in Table 6.4. Some of these errors are caused by the resemblance between textures, such as (D110, D9) and (D23, D27). Other errors come from the confusion between textures D23, D27, D74 and D98, where objects with similar shapes are observed. Refer to Figure 6.9 for some examples of these textures with high misclassification rates. Similar texture classification errors have been observed in Method I and Method II. It seems that filter-based methods, such as those investigated in this work, are more efficient in characterizing the edges in textures. However, they are inefficient in capturing the differences between the gray scale distributions of the sub-regions in textures.

It is also interesting to look at the behavior of the extracted features when the number of nearest neighbors in the  $k$ -NN classifier is increased. On the Brodatz texture data set, both the performances of Method II and the proposed method keep increasing as  $k$  increases, while that of Method I reaches the maximum at  $k = 5$  and then drops a little. In this case, the features extracted by Method II and the proposed methods are more coherent.

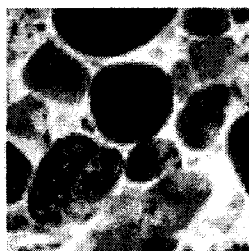
On the Outex texture data set, the performances of all three methods tend to



D110



D9



D23



D27



D74



D98

Figure 6.9: Textures with low classification rates.

drop as  $k$  increases. However, when  $k$  increases from 1 to 9, the decrease in the performance of Method I is much more significant (over 6 percent) than that of the other two methods (about 1% only). The reason for this phenomenon is that, for some textures, the features extracted by Method I from the training data set do not fit well enough with those from the test data set, as shown in Figure 6.10. On the Vistex texture data set, the performances of both Method I and the proposed method tend to drop slightly (between 1 and 1.5%), while that of Method II tends to remain at the same level.

## 6.5 Script Identification - A Case Study

In this section, we will investigate a problem from the document image processing field - script identification. We will apply some texture classification techniques to this problem, including the Ridgelet transform-based texture classification method proposed in this chapter. We will show that, when treated as a texture classification problem, the script identification problem can be solved satisfactorily with the proposed feature extraction technique. We will evaluate some recently published texture classification algorithms and describe the best performance that has been achieved by the Ridgelet transform-based method.

### 6.5.1 Background

Although many Optical Character Recognition (OCR) systems have been developed over the past few decades, almost all existing studies on OCR assume that the language or script in which a document has been prepared is already known. However, as the world is getting more interconnected, documents in different languages have become more pervasive. Therefore, script identification has become an important



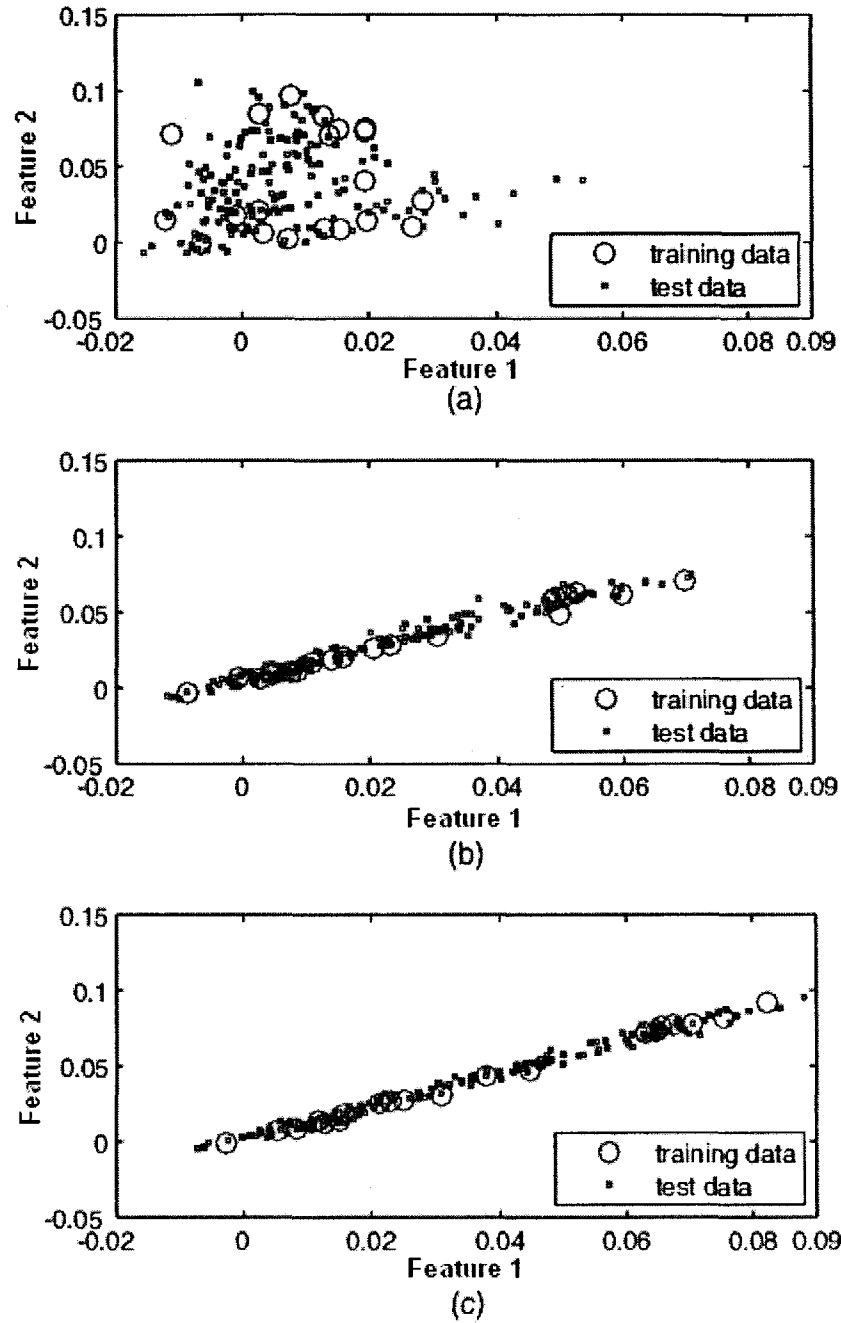


Figure 6.10: Plots of the training and testing samples in the feature space. Each graph (a, b, c) corresponds to one feature extraction method. The features have been extracted from texture *Canvas023* in the Outex data set. For each figure, we used the two features with the highest discrimination power as x and y axes: (a) Features extracted by Method I, (b) features extracted by Method II, and (c) features extracted by the proposed method.

step in the automatic processing of document images in such a multilingual environment, especially where the high volume of documents and the variety of languages make manual identification impractical. Early determination of the language used in a document can greatly facilitate further processing, such as character recognition, document image indexing and translation.

Research work in script identification techniques can be grouped into two categories: local methods and global methods. In local methods, features or specific character/word tokens are extracted at the connected component level or text line level. Therefore, these methods generally require preprocessing like skew removal, page decomposition and connected component analysis. In global methods, language identification is treated as a texture classification problem. Textural features are extracted at the regional level and thus no connected component analysis is needed. Furthermore, since rotation invariant features are available, image skew removal is not necessary in global methods. In the next two subsections, we will provide background information for these two different methods.

#### **6.5.1.1 Local Methods**

There has been a lot of research on local script identification methods. In [114], Spitz first classifies the languages under consideration into Asian languages (Chinese, Japanese, and Korean) or European languages (English, French, German, and Russian) using the vertical distributions of the *upward concavities*. Where two runs of black pixels appear on top of a single scan line of the image, if there is a run of black pixels on that scan line that spans the distance between these two runs, an upward concavity is formed on the line. Statistics of the optical densities (the density of black pixels in the connected components) are calculated to further differentiate the three Asian languages. The European languages are discriminated by means of the

most frequent occurring word shape tokens. These tokens are also derived from the connected components.

Extensions of Spitz's work have been made by Lee et al. [67] and Ding et al. [31] to cope with complex degraded document images and with more languages. More features are extracted to achieve these goals, including the distributions of character height, the top and bottom profiles of character bounding boxes, the ratio of the white to black areas in the horizontal projection profiles, and the relative positions of character bounding boxes.

Hochberg et al. [47] have used cluster analysis to find the frequent character or word shapes in each script and a representative template is defined for each cluster. Script identification is achieved by comparing a subset of the document's textual symbols to these templates and finding the best match.

Pal et al. [95] have performed script identification at the text line level. The scripts under consideration include English, Chinese, Arabic, Devanagari and Bangla. First, the length and position of the longest horizontal run in a text line are used as features to classify these scripts into Indian scripts or Non-Indian scripts. Then, the three non-Indian scripts are differentiated by means of: vertical black run information, character density difference between the original image and the image after applying a modified RLSA algorithm [128], distribution of lowermost points of the component and some features based on water overflow analogy. For Devanagari and Bangla scripts, some specific character level stroke configurations [94] are used to achieve the script identification goal.

Elgammal et al. [36] have studied the English/Arabic script identification at both the text line level and word level. Three classes of features are considered, including peak-number extracted from horizontal projection profiles, moments from horizontal projection profiles and 2-D run-length histograms. Their experiments show that at

the text line level, moment-based features outperform the other two types of features, while the run-length histogram achieves the best results at the word level.

In all the methods mentioned above, features have been chosen manually. In order to automate the feature selection process and reduce the amount of textual information used in script identification, Ablavsky et al. [1] have proposed an automatic feature selection scheme, which works on features extracted from the connected components. At the training stage, 72 features are extracted, including moment features; shape features like compactness, curvature, holes, and eccentricity; and features extracted from the co-occurrence matrix. Then, the RELIEF-F algorithm [62] is applied to select the 25 features that have the top discriminating power. At the test stage, a stream of connected components is generated and the 25 selected features are applied. A simple  $k$  Nearest Neighbor (k-NN) classifier is used to assign the likelihood of a script to each connected component. These likelihood values are fed into the evidence accumulation framework to finally give the script a label.

#### 6.5.1.2 Global Methods

Instead of using local textual features, global methods make use of texture features extracted from document image patches for the script identification purpose. Since different scripts often have their distinctive visual appearances, Tan [119] has formulated the language identification problem as a texture classification problem. Representative features for each script have been obtained by computing the mean of the channel output of Gabor filters at different central radial frequencies and orientations.

To eliminate the need to apply many image filter operations to achieve rotational invariant features, steerable approximation of the Gabor multichannel filtering has been proposed in [98] for feature extraction purposes. Experimental results on document images printed in Chinese, Japanese, Korean and English have shown that an

over 98.5% correct language identification rate has been achieved, while image filtering operations have been reduced by 40%. The possibility of using texture classification methods to identify languages has also been mentioned in [52].

In this chapter, we follow the same idea of the global methods and treat script identification as a texture classification problem. Since paper documents could be scanned with skewness, or even with different orientations, methods that achieve rotational invariant script identification are desirable. Luckily, many texture classification algorithms, including the method proposed in this chapter, are rotationally invariant.

## **6.5.2 Experiments**

In this section, the evaluation of some recent texture classification methods on the problem of script identification is presented. These methods all fall into the group of global methods. First, we will show how the data used in the evaluation procedure is prepared. Then, we will briefly state the methods adopted in this evaluation process. Finally, we will present the experimental results and discussions.

### **6.5.2.1 Data Preparation**

In this research, six languages have been investigated, including Chinese, Korean, Japanese, English, Russian and Arabic. To create a database for the evaluation, we have collected 550 non-overlapping sample image blocks for each language. Of these image blocks, 300 have been used for training and the remaining 250 have been used for testing. These image blocks were extracted from newspapers, books, magazines and computer printouts, which were scanned at a resolution of 200 dpi. Each sample image block has 256 by 256 pixel. Other block sizes are also possible. However, to ensure that enough textural information could be retained for language identification, we have verified that the block size should not be too small.

Font variations have also been taken into consideration during the data preparation process. For example, four fonts, including SongTi, KaiTi, HeiTi and Fang Song, have been used for the Chinese language and three fonts, including Times New Roman, Arial and Courier, have been used for the English language. For the other four languages, different fonts have also been used. Foreign characters, such as English characters in oriental languages, have been preserved in our samples. Figure 6.11 shows some sample images in our database.

In order to evaluate the rotation invariance of the script identification methods, the above database has been extended by rotating each sample in that basic database by an angle randomly chosen within  $(0, 180)$  degrees. The nearest neighbor interpolation method has been adopted in the image rotation procedure. In total, we have 600 samples per script for training and 500 samples per script for testing.

#### **6.5.2.2 Selected Methods for Script Identification**

During the evaluation process, we investigated the following four global rotational invariant methods:

##### **Method-A:**

Gabor multichannel filtering [119], where each feature is extracted as the average of each Gabor channel output. Rotation invariance is achieved through the application of the 1-D Discrete Fourier Transform (DFT) to the features extracted from those Gabor channels that have the same central frequency.

##### **Method-B:**

Steerable Gabor multichannel filtering [98], which provides an efficient rotational invariant feature extraction method.

##### **Method-C:**

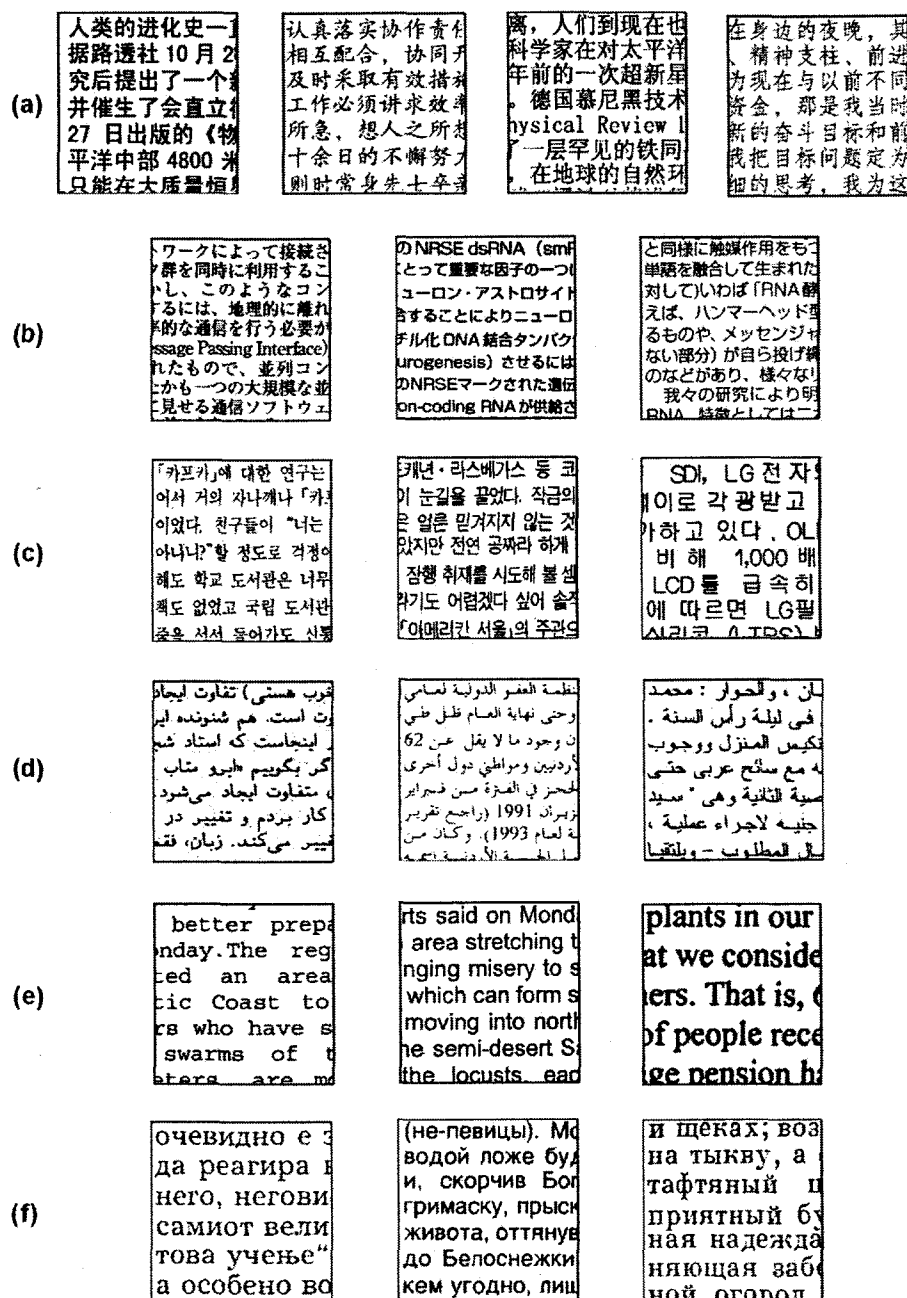


Figure 6.11: Image samples in different scripts with different fonts: (a) Chinese, (b) Japanese, (c) Korean, (d) Arabic, (e) English, and (f) Russian.

Radon transform plus wavelet transform [49], where the Radon transform is applied first to estimate the principal orientation of the texture image, and then DWT is applied along the principal orientation for feature extraction. This method also achieves rotational invariance. Two wavelet transforms are considered in this method: Daubechies orthogonal wavelets db4 and db6. The two variations of this method are denoted as Method-C-db4 and Method-C-db6.

#### **Method-D:**

Ridgelet transform-based method [96], which is the texture classification method proposed in this thesis.

### **6.5.3 Experimental Results And Discussion**

In the experiments, each of the texture classification methods has been applied to the image samples in the database and texture features have been extracted. Then, a  $k$ -NN classifier has been adopted for classification. The effect of the number of the nearest neighbors on the correct classification rate has also been investigated, with  $k = 1, 3, 5$ , and  $7$ , respectively. The results are shown in Table 6.5.

These results show that Gabor multichannel filtering-based methods and the method using the Ridgelet transform (Method-A, Method-B, and Method-D) achieve better performances (about 99% on average) when compared with the DWT-based method (Method-C)(with an average recognition rate lower than 97.5%). This result is expected since characters are composed of strokes at different orientations, which is an important texture property of the document images. Gabor multichannel filtering and Ridgelet transform-based methods have good direction selectivity and therefore are appropriate in capturing this texture property of the document images. On the other hand, although the Radon transform can be applied to find the principal orientation of the document image as a whole in Method-C, the followed DWT



has limited direction selectivity and thus can not adequately capture the dominant texture property of the document images.

For Method-C, the performance is also dependant on the selection of the wavelet. The orthogonal wavelets used in this evaluation are from Daubechies family wavelets, which are usually denoted as  $dbN$ . Here, ‘ $N$ ’ stands for the order, or the number of vanishing moments, of the wavelet. The regularity of a Daubechies wavelet increases with its order. Furthermore, as the order increases, the Daubechies wavelet becomes more localized in the frequency domain. Therefore, higher order Daubechies wavelets can provide better frequency selectivity, and better features can be extracted. This property explains why  $db6$  outperforms  $db4$  (about 1% gain in average recognition rate when  $k=1$ , as shown in Table 6.5). On the other hand, the difference in performance becomes smaller when  $N$  becomes even larger.

The best performance is achieved by the feature extraction method using the Ridgelet transform. Apart from its advantages presented in previous sections, such as frequency-orientation decomposition on a polar grid and localization in the frequency domain, Ridgelet transform is very good at handling the line singularities that are abundant in the document images. As shown in Table 6.5, among all 6 languages under investigation, Chinese and Korean are the scripts that our method can not ideally differentiate, especially when  $k$  becomes larger. On the other 4 scripts, our method achieves 100 percent recognition rate. This is a very interesting result since it reveals that, from texture point of view, Chinese and Korean are closer or similar to each other.

## 6.6 Conclusion

We proposed a new rotation invariant feature extraction method in the Ridgelet transform domain for texture classification. This method is advantageous mainly

Table 6.5: Performance of the script identification algorithms on the basic database (%).

|          | Method-A |      |      |      |       |      |      | Method-B |       |       |      |      |       |      | Method-C-db4 |      |       |       |       |       |       | Method-C-db6 |       |       |       |       |       |       | Method-D |  |  |  |  |  |  |
|----------|----------|------|------|------|-------|------|------|----------|-------|-------|------|------|-------|------|--------------|------|-------|-------|-------|-------|-------|--------------|-------|-------|-------|-------|-------|-------|----------|--|--|--|--|--|--|
|          | k = 1    | 3    | 5    | 7    | k = 1 | 3    | 5    | 7        | k = 1 | 3     | 5    | 7    | k = 1 | 3    | 5            | 7    | k = 1 | 3     | 5     | 7     | k = 1 | 3            | 5     | 7     | k = 1 | 3     | 5     | 7     |          |  |  |  |  |  |  |
|          | k = 1    | 3    | 5    | 7    | k = 1 | 3    | 5    | 7        | k = 1 | 3     | 5    | 7    | k = 1 | 3    | 5            | 7    | k = 1 | 3     | 5     | 7     | k = 1 | 3            | 5     | 7     | k = 1 | 3     | 5     | 7     |          |  |  |  |  |  |  |
| Chinese  | 98.4     | 99.2 | 98.8 | 98.8 | 98.0  | 99.0 | 98.2 | 97.6     | 98.0  | 98.8  | 98.4 | 97.6 | 98.0  | 98.0 | 96.8         | 97.6 | 98.0  | 98.0  | 96.8  | 97.6  | 98.8  | 98.4         | 98.8  | 98.8  | 98.8  | 98.8  | 98.8  | 98.8  |          |  |  |  |  |  |  |
| Japanese | 99.6     | 99.0 | 99.0 | 98.6 | 99.2  | 98.8 | 98.8 | 98.2     | 96.4  | 97.6  | 97.2 | 96.8 | 97.6  | 97.2 | 98.0         | 98.0 | 100.0 | 100.0 | 99.4  | 99.4  | 100.0 | 100.0        | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |          |  |  |  |  |  |  |
| Korean   | 98.4     | 98.2 | 98.0 | 96.4 | 97.8  | 98.0 | 97.6 | 96.2     | 90.8  | 91.2  | 91.2 | 91.2 | 94.0  | 92.8 | 93.6         | 94.0 | 92.8  | 93.6  | 94.0  | 98.0  | 97.2  | 97.2         | 97.2  | 94.8  | 99.2  | 99.6  | 99.2  | 99.2  |          |  |  |  |  |  |  |
| Russian  | 99.0     | 98.2 | 99.0 | 98.6 | 98.2  | 98.0 | 98.8 | 96.0     | 95.2  | 94.0  | 94.0 | 92.8 | 96.0  | 94.4 | 93.6         | 93.6 | 99.6  | 99.4  | 100.0 | 100.0 | 100.0 | 100.0        | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |       |          |  |  |  |  |  |  |
| English  | 98.3     | 98.6 | 98.6 | 98.2 | 98.0  | 98.2 | 98.2 | 98.2     | 99.0  | 100.0 | 99.4 | 99.4 | 99.0  | 99.6 | 99.4         | 99.4 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0        | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |          |  |  |  |  |  |  |
| Arabic   | 100.0    | 99.6 | 99.6 | 99.6 | 99.8  | 99.4 | 99.6 | 99.4     | 98.0  | 98.0  | 97.6 | 98.8 | 99.2  | 99.6 | 99.2         | 99.2 | 100.0 | 100.0 | 100.0 | 99.2  | 100.0 | 100.0        | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 | 100.0 |          |  |  |  |  |  |  |
| Average  | 98.9     | 98.8 | 98.8 | 98.4 | 98.5  | 98.5 | 98.5 | 97.6     | 96.4  | 96.6  | 96.3 | 96.1 | 97.4  | 96.9 | 96.7         | 96.9 | 99.4  | 99.2  | 99.4  | 99.2  | 99.2  | 99.2         | 99.2  | 99.2  | 99.2  | 99.2  | 99.2  | 98.8  |          |  |  |  |  |  |  |

in two aspects. First, the Radon transform stage in Ridgelet transform provides us with information about data on polar grids in the frequency domain, which is ideal for rotation invariant feature extraction. Second, the frequency B-spline wavelet transform brings good frequency selectivity. This enables the extraction of features from appropriate sub-bands in the frequency domain. Another advantage of the proposed method is that working in the Radon transform domain gives features that are more robust against additive noise. We compared the proposed method with two recent methods in the field of texture classification, the wavelet-based method and the Gabor multichannel filtering method. Experimental results show that the proposed method performs better than the other two methods on all three popular data sets. Application of these methods to script identification, a problem from the document image processing field, has also been investigated.

# Chapter 7

## Conclusion

In this thesis, two feature extraction strategies using over-complete and sparse representations have been investigated: a) feature extraction in the transformed domain by globally applying over-complete and sparse transformations, and b) feature extraction via local over-complete and sparse structure analysis.

In the first strategy, the over-complete and sparse transformations are directly applied to the input pattern as a whole. This strategy does not focus on the local details of the input patterns. A special case of this strategy, rotation invariant feature extraction using Ridgelet transform, has been applied to the problem of texture classification. The proposed method achieves 95.94% on the Brodatz texture database, 99.3% on the Outex texture database, and 97.5% on the Vistex texture database. These results outperform those that are achieved by the wavelet-based method and the Gabor multichannel filtering method. When applied to the problem of rotational invariant script identification in document images using texture properties, the proposed method has achieved an average recognition rate of 99.4% on a database containing 6 scripts: Chinese, English, Russian, Japanese, Korean, and Arabic.

Contrary to the first global strategy, the second feature extraction strategy explicitly performs local over-complete and sparse structure analysis, by dividing the input patterns into small patches and performing learning on these local data. The

learning process is achieved via the K-SVD algorithm, which has been chosen in this thesis due to its simplicity and flexibility in controlling how sparse the representation could be given by the learned dictionary.

The learned local over-complete and sparse structure has been applied to detect texts from scene images in two different ways. First, the local over-complete and sparse structure is learned solely from text signals so that it is able to give sparse representations to text signals and non-sparse representations to non-text signals. Thus, text detection can be achieved via *sparsity testing*. Experimental results show that a good text detection rate has been achieved with a recall rate of 75.2% by the sparsity testing technique. However, it gives a relatively high frequency of false alarms and the precision rate of text detection is only 67.6%. Meanwhile, the time complexity of this method is also high. To overcome these drawbacks, a second text detection method is proposed in this thesis, where the text detection problem is converted into a shape recognition problem using the Topographic Map Representation of an image. The shape recognition is achieved using a combination of the learned over-complete and sparse dictionary and the CNN. With the second text detection algorithm, we are able to achieve a recall rate of 73.2% and a precision rate of 78.7%. All these results show that over-complete and sparse structure is very useful in the text detection problem.

The idea of combining the learned over-complete and sparse dictionary and the CNN has been extended to problems related to handwritten character recognition. In this thesis, handwritten Farsi numeral recognition has been investigated and very high recognition rates have been achieved (99.22% on the CENPARMI Farsi Numeral Database and 99.5% on the HODA Farsi Numeral Database).

The research work in this thesis has shown promising results of applying over-complete and sparse representations into the field of pattern recognition and detection.

Future research could be conducted along two directions. First, research on learning algorithms that produce over-complete and sparse dictionaries for given data is still at an early stage. More research endeavors should be made in this direction to find new algorithms that are more efficient and more capable in handling larger data sets and larger image patches. Second, applications other than those mentioned in this thesis are worth investigating, using the idea of the over-complete and sparse structure. One possible application, for example, is to apply the over-complete and sparse structure in the feature extraction procedure in handwritten word recognition.

# Bibliography

- [1] V. Ablavsky and M.R. Stevens. Automatic feature selection with applications to script identification of degraded documents. In *Proc. 7th International Conference on Document Analysis and Recognition*, pages 750–754, 2003.
- [2] I.S.I. Abuhaiba, S.A. Mahmoud, and R.J. Green. Recognition of handwritten cursive Arabic characters. *IEEE Trans. Patt. Anal. Mach. Intell.*, 16(6):664–672, 1994.
- [3] M. Aharon, M. Elad, and A.M. Bruckstein. The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Processing*, 54(11):4311–4322, 2006.
- [4] R.R. Bailey and M. Srinath. Orthogonal moment features for use with parametric and non-parametric classifiers. *IEEE Trans. Patt. Anal. Mach. Intell.*, 18(4):389–398, 1996.
- [5] J.S. Bleck, U. Ranft, M. Gebel, H. Hecker, M. Westhoff-Bleck, C. Thiesemann, S. Wagner, and M. Manns. Random field models in the textural analysis of ultrasonic images of the liver. *IEEE Trans. on Medical Imaging*, 15(6):796–801, 1996.
- [6] P.J. Burt and E.H. Adelson. The Laplacian pyramid as a compact image code. *IEEE Trans. Commun.*, 31(4):532–540, 1983.
- [7] A. Busch, W.W. Boles, and S. Sridharan. Texture for script identification. *IEEE Trans. Patt. Anal. Mach. Intell.*, 27(11):1720–1732, 2005.

- [8] F.W. Campbell and J. G. Robson. Application of fourier analysis to the visibility of gratings. *J. Physiol. (Lond.)*, 197:551–566, 1968.
- [9] E. J. Candes. *Ridgelets: Theory And Applications*. PhD thesis, Department of Statistics, Stanford University, 1998.
- [10] E.J. Candes and D.L. Donoho. *Curvelets - a surprisingly effective nonadaptive representation for objects with edges*. Curves and Surfaces, Vanderbilt University Press, Nashville, TN, 2000.
- [11] E.J. Candes and D.L. Donoho. Continuous curvelet transform: I. resolution of the wavefront set. *Appl. Comput. Harmon. Anal.*, 19:162–197, 2003.
- [12] E.J. Candes and D.L. Donoho. Continuous curvelet transform: II. discretization and frames. *Appl. Comput. Harmon. Anal.*, 19:198–222, 2003.
- [13] J. Canny. A computational approach to edge detection. *IEEE Trans. Patt. Anal. Mach. Intell.*, 8(6):679–698, 1986.
- [14] V. Caselles, B. Coll, and J.M. Morel. Topographic maps and local contrast changes in natural images. *International Journal of Computer Vision*, 33(1):5–27, 1999.
- [15] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>, 2001.
- [16] T. Chang and C.-C.J. Kuo. Texture analysis and classification with tree-structured wavelet transform. *IEEE Trans. Image Processing*, 2(4):429–441, 1993.
- [17] D. Charalampidis and T. Kasparis. Wavelet-based rotational invariant roughness features for texture classification and segmentation. *IEEE Trans. Image Processing*, 11(8):825–837, 2002.
- [18] D. Chen, J.M. Odobez, and H. Bourlard. Text detection and recognition in images and video frames. *Pattern Recognition*, 3(37):595–608, 2004.



- [19] G. Y. Chen, T. D. Bui, and A. Krzyzak. Rotation invariant pattern recognition using ridgelet, wavelet cycle-spinning, and fourier features. *Pattern Recognition*, 38(12):2314–2322, 2005.
- [20] G. Y. Chen, T. D. Bui, and A. Krzyzak. Rotation invariant feature extraction using ridgelet and fourier transforms. *Pattern Analysis & Applications*, 9(1):83–93, 2006.
- [21] J.-L. Chen and A. Kundu. Rotation and grey-scale transform invariant texture identification using wavelet decomposition and HMM. *IEEE Trans. PAMI*, 16(2):208–214, 1994.
- [22] Xiangrong Chen and Alan L. Yuille. Detecting and reading text in natural scenes. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 366–373, 2004.
- [23] Xilin Chen, Jie Yang, Jing Zhang, and A. Waibel. Automatic detection and recognition of signs from natural scenes. *IEEE Trans. on Image Processing*, 13(1):87–99, 2004.
- [24] D. Chetverikov, Jisheng Liang, J. Komuves, and R. M. Haralick. Zone classification using texture features. *Proceedings of the 13th International Conference on Pattern Recognition*, 3:676–680, 1996.
- [25] C. K. Chui. *An Introduction to Wavelets*. Academic Press, 1992.
- [26] F.S. Cohen, Z. Fan, and M.A. Patel. Classification of rotated and scaled textured images using Gaussian Markov Random Field models. *IEEE Trans. Patt. Anal. Mach. Intell.*, 13(2):192–202, 1991.
- [27] G. Davis, S. Mallat, and M. Avellaneda. Adaptive greedy approximations. *J. Constr. Approx.*, 13:57–98, 1997.

- [28] M. Dehghan and K. Faez. Farsi handwritten character recognition with moment invariants. In *Proc. 13th International Conference on Digital Signal Processing*, volume 2, pages 507–510, 1997.
- [29] A. Desolneux, L. Moisan, and J. Morel. Edge detection by Helmholtz principle. *International Journal of Computer Vision*, 14:271–284, 2001.
- [30] G. Dimauro, S. Impedovo, G. Pirlo, and A. Salzo. Automatic bankcheck processing: a new engineered system. *Machine Perception and Artificial Intelligence*, 28:5–42, 1997.
- [31] J. Ding, L. Lam, and C.Y. Suen. Classification of oriental and european scripts by using characteristic features. In *Proc. 4th International Conference on Document Analysis and Recognition*, pages 1,023–1,027, 1997.
- [32] M.N. Do. *Directional Multiresolution Image Representations*. PhD thesis, Swiss Federal Institute of Technology, 2001.
- [33] M.N. Do and M. Vetterli. Rotation invariant characterization and retrieval using steerable wavelet-domain hidden markov models. *IEEE Trans. Multimedia*, 4(4):517–526, 2002.
- [34] David L. Donoho, Xiaoming Huo, Ian Jermyn, Peter Jones, Gilad Lerman, Ofer Levi, and Frank Natterer. Beamlets and multiscale image analysis. In *Proc. Multiscale and Multiresolution Methods*, pages 149–196. Springer, 2001.
- [35] D.L. Donoho. De-noising by soft-thresholding. *IEEE Trans. on Information Theory*, 41(3):613–627, 1995.
- [36] A.M. Elgammal and M.A. Ismail. Techniques for language identification for hybrid arabic-english document images. In *Proc. 6th International Conference on Document Analysis and Recognition*, pages 1100–1104, 2001.
- [37] K. Engan, S.O. Aase, and J.H. Husy. Multi-frame compression: Theory and design. *EURASIP Signal Process*, 80(10):2121–2140, 2000.

- [38] L. Fausett. *Fundamentals Of Neural Networks*, pages 218–287. Prentice-Hall, 1994.
- [39] D.J. Field. Wavelets, vision and the statistics of natural scenes. *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, 357(1760):2527–2542, 1999.
- [40] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Qian Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: the QBIC system. *IEEE Computer*, 28(9):23–32, 1995.
- [41] S.R. Fountain and T.N. Tan. Extraction of noise robust invariant texture features via multichannel filtering. *Proceedings ICIP*, 3:197–200, 1997.
- [42] J. Gao and J. Yang. An adaptive algorithm for text detection from natural scenes. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume II, pages 84–89, 2001.
- [43] C. Garcia and X. Apostolidis. Text detection and segmentation in complex color images. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 4, pages 2326–2329, 2000.
- [44] J. Gllavata, R. Ewerth, and B. Freisleben. Text detection in images based on unsupervised classification of high-frequency wavelet coefficients. In *Proceedings of the 17th International Conference on Pattern Recognition*, volume 1, pages 425–428, 2004.
- [45] G.M. Haley and B.S. Manjunath. Rotation-invariant texture classification using a complete space-frequency model. *IEEE Trans. Image Processing*, 8(20):255–269, 1999.
- [46] G. T. Herman. *Image Reconstruction from Projections: The Fundamentals of Computerized Tomography*. Academic Press, 1980.

- [47] J. Hochberg, L. Kerns, P. Kelly, and T. Thomas. Automatic script identification from images using cluster-based templates. *IEEE Trans. Patt. Anal. Mach. Intell.*, 19(2):176–181, 1997.
- [48] Li Huiping, D. Doermann, and O. Kia. Automatic text detection and tracking in digital video. *IEEE Trans. on Image Processing*, 9(1):147–156, 2000.
- [49] K. Jafari-Khouzani and H. Soltanian-Zadeh. Radon transform orientation estimation for rotation invariant texture analysis. *IEEE Trans. Patt. Anal. Mach. Intell.*, 27(6):1004–1008, 2005.
- [50] K. Jafari-Khouzani and H. Soltanian-Zadeh. Rotation invariant multiresolution texture analysis using radon and wavelet transforms. *IEEE Trans. Image Processing*, 14(6):783–795, 2005.
- [51] A. K. Jain and B. Yu. Automatic text location in images and video frames. *Pattern Recognition*, 31(12):2055–2076, 1998.
- [52] A.K. Jain and Y. Zhong. Page segmentation using texture analysis. *Pattern Recognition*, 29(5):743–770, 1996.
- [53] U. Kandaswamy, D. A. Adjeroh, and M. C. Lee. Efficient texture analysis of SAR imagery. *IEEE Trans. on Geoscience and Remote Sensing*, 43(9):2075–2083, 2005.
- [54] R. L. Kashyap and A. Khotanzed. A model-based method for rotation invariant texture classification. *IEEE Trans. Patt. Anal. Mach. Intell.*, 8(4):472–481, 1986.
- [55] D. Keysers, T. Deselaers, C. Gollan, and H. Ney. Deformation models for image recognition. *IEEE Trans. Patt. Anal. Mach. Intell.*, 29(8):1422–1435, 2007.
- [56] Hossein Khosravi and Ehsanollah Kabir. Introducing a very large dataset of handwritten Farsi digits and a study on their varieties. *Pattern Recognition Letters*, 28:1133–1141, 2007.

- [57] JiSoo Kim, SangCheol Park, and SooHyung Kim. Text locating from natural scene images using image intensities. In *Proceedings of eighth International Conference on Document Analysis and Recognition*, volume 2, pages 655–659, 2005.
- [58] K. Kim, H. Byun, Y. Song, Y. Choi, S. Chi, K. Kim, and Y. Chung. Scene text extraction in natural images using hierarchical feature combining and verification. In *Proc. Intl. Conf. Pattern Recognition*, pages 679–682, 2004.
- [59] Kwang In Kim, Keechul Jung, and Jin Hyung Kim. Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm. *IEEE Trans. Patt. Anal. Mach. Intell.*, 25(12):1631–1639, 2003.
- [60] F. Kimura, K. Takashina, S. Tsuruoka, and Y. Miyake. Modified quadratic discriminant functions and the application to Chinese character recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 9(1):149–153, 1987.
- [61] N. G. Kingsbury. Complex wavelets for shift invariant analysis and filtering of signals. *Journal of Applied and Computational Harmonic Analysis*, (3):234–253, 2001.
- [62] I. Kononenko. Estimating attributes: Analysis and extensions of relief. In *Proc. European Conference on Machine Learning*, pages 171–182, 1994.
- [63] K. Kreutz-Delgado, J. F. Murray, B. D. Rao, K. Engan, T. Lee, and T. J. Sejnowski. Dictionary learning algorithms for sparse representation. *Neural Comp.*, 15(2):349–396, 2003.
- [64] Fabien Lauer, Ching Y. Suen, and Grard Bloch. A trainable feature extractor for handwritten digit recognition. *Pattern Recognition*, 40(6):1816–1824, 2007.
- [65] Y. LeCun, L. Bottou, and Y. Bengio. Reading checks with graph transformer networks. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages 151–154, 1997.

- [66] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. of the IEEE*, 86(11):2278–2324, November 1998.
- [67] D.S. Lee, C.R. Nohl, and H.S. Baird. Language identification in complex, un-oriented, and degraded document images. In *Proc. IAPR Workshop on Document Analysis Syst.*, pages 76–98, 1996.
- [68] S. Lesage, R. Gribonval, F. Bimbot, and L. Benaroya. Learning unions of orthonormal bases with thresholded singular value decomposition. In *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, volume 5, pages 293–296, 2005.
- [69] M. S. Lewicki and T. J. Sejnowski. Learning overcomplete representations. *Neural Comp.*, 12:337–365, 2000.
- [70] Rainer Lienhart and Axel Wernicke. Localizing and segmenting text in images and videos. *IEEE Trans. Circuit. Syst. Video Tech.*, 12(4):256–268, 2002.
- [71] C.-L. Liu, K. Nakashima, H. Sako, and H. Fujisawa. Handwritten digit recognition: Benchmarking of the state-of-the-art techniques. *Pattern Recognition*, 36(10):2271–2285, 2003.
- [72] C.-L. Liu, K. Nakashima, H. Sako, and H. Fujisawa. Handwritten digit recognition: Investigation of normalization and feature extraction techniques. *Pattern Recognition*, 37(2):265–279, 2004.
- [73] Cheng-Lin Liu and Hiroshi Sako. Class-specific feature polynomial classifier for pattern classification and its application to handwritten numeral recognition. *Pattern Recognition*, 39(4):669–681, 2006.
- [74] Y. Liu, S. Goto, and T. Ikenaga. A robust algorithm for text detection in color images. In *Proceedings of Eighth International Conference on Document Analysis and Recognition*, volume 1, pages 399–403, 2005.

- [75] Ying Liu and S. N. Srihari. Document image binarization based on texture features. *IEEE Trans. Patt. Anal. Mach. Intell.*, 19(5):540–544, 1997.
- [76] Y. M. Lu and M. N. Do. Multidimensional directional filter banks and surfacelets. *IEEE Trans. on Image Processing*, 16(4):918–931, 2007.
- [77] S.M. Lucas. Icdar 2005 text locating competition results. In *Proceedings of Eighth International Conference on Document Analysis and Recognition*, volume 1, pages 80–84, 2005.
- [78] S.M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young. Icdar 2003 robust reading competitions. In *Proc. Intl. Conf. Document Analysis and Recognition*, pages 682–687, 2003.
- [79] S. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, 2nd edition, 1999.
- [80] S. Mallat and Z. Zhang. Matching pursuit in a time-frequency dictionary. *IEEE Trans. Signal Proc*, 41:3397–3415, 1993.
- [81] S.G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Trans. Patt. Anal. Mach. Intell.*, 11:674–693, 1989.
- [82] S.G. Mallat. *A Wavelet Tour of Signal Processing*. Academic Press, 1998.
- [83] J. Mao and A.K. Jain. Texture classification and segmentation using multiresolution simultaneous autoregressive models. *Pattern Recognition*, 25(2):173–188, 1992.
- [84] Ranzato Marc’Aurelio, Christopher Poultney, Sumit Chopra, and Yann LeCun. Efficient learning of sparse representations with an energy-based model. In MIT Press, editor, *Proc. Advances in Neural Information Processing Systems*, 2006.
- [85] S. Marcelja. Mathematical description of the responses of simple cortical cells. *Optical Society of America Journal A*, 70:1297–1300, 1980.

- [86] V.Y. Mariano and R. Kasturi. Locating uniform-colored text in video frames. In *Proc. 15th International Conference on Pattern Recognition*, volume 4, pages 539–542, 2000.
- [87] P. Monasse and F. Guichard. Fast computation of a contrast-invariant image representation. *IEEE Trans. Image Processing*, 9(5):860–872, 2000.
- [88] A. Mowlaei and K. Faez. Recognition of isolated handwritten Persian/Arabic characters and numerals using support vector machines. In *Proc. IEEE 13th Workshop on Neural Networks for Signal Processing*, pages 547–554, 2003.
- [89] B.K. Natarajan. Sparse approximate solutions to linear systems. *SIAM J. Comput.*, 24:227–234, 1995.
- [90] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Patt. Anal. Mach. Intell.*, 24(7):971–987, 2002.
- [91] B.A. Olshausen and D.J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.
- [92] B.A. Olshausen and D.J. Field. Sparse coding with an overcomplete basis set: a strategy employed by v1? *Vision Research*, 37:3311–3325, 1997.
- [93] N. Otsu. A threshold selection method from gray level histograms. *IEEE Trans. Systems Man Cybernet.*, 9:62–66, 1979.
- [94] U. Pal and B.B. Chaudhuri. Automatic separation of words in multi-lingual multi-script indian documents. In *Proc. 4th International Conference on Document Analysis and Recognition*, pages 576–579, 1997.
- [95] U. Pal and B.B. Chaudhuri. Automatic identification of english, chinese, arabic, devnagari and bangla script line. In *Proc. 6th International Conference on Document Analysis and Recognition*, pages 790–794, 2001.



- [96] W. M. Pan, T. D. Bui, and C. Y. Suen. Rotation invariant texture classification by ridgelet transform and frequency-orientation space decomposition. *Signal Processing*, 88(1):189–199, 2008.
- [97] W.M. Pan, T.D. Bui, and C.Y. Suen. Rotation-invariant texture classification using steerable Gabor filter bank. *Springer Lecture Notes in Computer Science*, 3656:746–753, 2005.
- [98] W.M. Pan, C.Y. Suen, and T.D. Bui. Script identification using steerable Gabor filters. In *Proc. 8th International Conference on Document Analysis and Recognition*, volume 2, pages 883–887, 2005.
- [99] Wumo Pan, T.D. Bui, and C.Y. Suen. Isolated handwritten Farsi numerals recognition using learned sparse-overcomplete representations. Technical report, Center for Pattern Recognition and Machine Intelligence, 26 pages, 2008.
- [100] Wumo Pan, T.D. Bui, and C.Y. Suen. Text detection from scene images using sparse representation. In *Proc. 19th Intl. Conference on Pattern Recognition*, 5 pages, 2008.
- [101] E. Le Pennec and S. Mallat. Sparse geometrical image approximation with Bandelets. *IEEE Trans. Image Processing*, 14(4):423–438, 2004.
- [102] R. Porter and N. Canagarajah. Robust rotation-invariant texture classification: Wavelet, Gabor filter and GMRF based schemes. *IEE Proceedings - Vision Image Signal Processing*, 144(3):180–188, 1997.
- [103] C.M. Pun and M.C. Lee. Log-polar wavelet energy signatures for rotation and scale invariant texture classification. *IEEE Trans. Patt. Anal. Mach. Intell.*, 25(5):590–603, 2003.
- [104] Ye Qixiang, Huang Qingming, Gao Wen, and Zhao Debin. Fast and robust text detection in images and video frames. *Image and Vision Computing*, 23(6):565–576, 2005.

- [105] T. Randen and J. H. Husoy. Filtering for texture classification: A comparative study. *IEEE Trans. Patt. Anal. Mach. Intell.*, 21(4):291–310, 1999.
- [106] T. Saoi, H. Goto, and H. Kobayashi. Text detection in color scene images based on unsupervised clustering of multi-channel wavelet features. In *Proceedings of Eighth International Conference on Document Analysis and Recognition*, volume 2, pages 690–694, 2005.
- [107] J. Serra. *Image Analysis and Mathematical Morphology*. Academic Press, New York, 1982.
- [108] Meng Shi, Yoshiharu Fujisawa, Tetsushi Wakabayashi, and Fumitaka Kimura. Handwritten numeral recognition using gradient and curvature of gray scale image. *Pattern Recognition*, 35(10):2051–2059, 2002.
- [109] M.H. Shirali-Shahreza, K. Faez, and A. Khotanzad. Recognition of handwritten Persian/Arabic numerals by shadow coding and an edited probabilistic neural network. In *Proc. International Conference on Image Processing*, volume 3, pages 436–439, 1995.
- [110] Patrice Y. Simard, Dave Steinkraus, and John Platt. Best practices for convolutional neural networks applied to visual document analysis. In *Proc. International Conference on Document Analysis and Recognition (ICDAR)*, pages 958–962, 2003.
- [111] E.P. Simoncelli, W.T. Freeman, E.H. Adelson, and D.J. Heeger. Shiftable multi-scale transforms. *IEEE Trans. Information Theory*, 38(2):587–607, 1992.
- [112] F. Solimanpour, J. Sadri, and C.Y. Suen. Standard databases for recognition of handwritten digits, numerical strings, legal amounts, letters and dates in Farsi language. In *Proc. 10th International Workshop on Frontiers in Handwriting Recognition*, pages 3–7, 2006.

- [113] H. Soltanzadeh and M. Rahmati. Recognition of Persian handwritten digits using image profiles of multiple orientations. *Pattern Recognition Lett.*, 25(14):1569–1576, 2004.
- [114] A.L. Spitz. Determination of the script and language content of document images. *IEEE Trans. Patt. Anal. Mach. Intell.*, 19(3):235–245, 1997.
- [115] S.N. Srihari and E.J. Keubert. Integration of handwritten address interpretation technology into the United States Postal Service Remote Computer Reader system. In *Proc. Fourth International Conference on Document Analysis and Recognition*, volume 2, pages 892–896, 1997.
- [116] J.L. Starck, E. Candes, and D.L. Donoho. The curvelet transform for image denoising. *IEEE Trans. on Image Processing*, 11(6):670–684, 2002.
- [117] G. Strang. *Introduction to Linear Algebra*. Wellesley-Cambridge, 3 edition, 1998.
- [118] C.Y. Suen, K. Liu, and N.W. Strathy. Sorting and recognizing cheques and financial documents. In *Proc. of third IAPR workshop on document analysis systems*, pages 1–18, 1998.
- [119] T.N. Tan. Rotation invariant texture features and their use in automatic script identification. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(7):751–756, 1998.
- [120] D.S. Taubman and M.W. Marcellin. *JPEG2000: Image Compression Fundamentals, Standards and Practice*. Springer, 2001.
- [121] A. Teolis. *Computational Signal Processing with Wavelets*. Birkhauser, 1998.
- [122] ICDAR 2003 Text Location Contest trial test database. <http://algoval.essex.ac.uk/icdar/datasets.html>.
- [123] Oivind D. Trier, A. K. Jain, and T. Taxt. Feature extraction methods for character recognition - a survey. *Pattern Recognition*, 29(4):641–662, 1996.

- [124] J. A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Trans. Inf. Theory*, 50(10):2231–2242, 2004.
- [125] M. Unser. Texture classification and segmentation using wavelet frames. *IEEE Trans. Image Processing*, 4:1549–1560, 1995.
- [126] R. L. De Valois and K. K. De Valois. *Spatial Vision*. Oxford University Press, 1988.
- [127] MIT Media Lab Vision Textures. <http://vismod.www.media.mit.edu>.
- [128] F.M. Wahl, K.Y. Wong, and R.G. Casey. Block segmentation and text extraction in mixed text/image documents. *Computer Graphics and Image Processing*, 20:375–390, 1982.
- [129] V. Wu, R. Manmatha, and E. Riseman. Textfinder: An automatic system to detect and recognize text in images. *IEEE Trans. Patt. Anal. Mach. Intell.*, 21(11):1224–1229, 1999.
- [130] W. Wu and S. Wei. Rotation and gray-scale transform invariant texture classification using spiral resampling, sub-band decomposition, and hidden markov model. *IEEE Trans. Image Process*, 5(10):1423–1434, 1996.
- [131] Qixiang Ye, Jianbin Jiao, Jun Huang, and Hua Yu. Text detection and restoration in natural scene images. *Journal of Visual Communication and Image Representation*, 18(6):504–513, 2007.
- [132] J. Zhang and T. Tan. Brief review of invariant texture analysis methods. *Pattern Recognition*, 35(3):735–747, 2002.
- [133] P. Zhang, T.D. Bui, and C.Y. Suen. A novel cascade ensemble classifier system with a high recognition performance on handwritten digits. *Pattern Recognition*, 40(12):3415–3429, 2007.

- [134] M. Ziaratban, K. Faez, and F. Faradji. Language-based feature extraction using template-matching in Farsi/Arabic handwritten numeral recognition. In *Proc. Ninth International Conference on Document Analysis and Recognition*, volume 1, pages 297–301, 2007.