UNIVERSITY OF CALIFORNIA, SAN DIEGO

Computational Models of Early Visual Processing Layers

A dissertation submitted in partial satisfaction of the requirements for the degree
Doctor of Philosophy

in

Computer Science and Cognitive Science

by

Honghao Shan

Committee in charge:

Professor Garrison W. Cottrell, Chair
Professor Marian Stewart Bartlett
Professor Serge J. Belongie
Professor Sanjoy Dasgupta
Professor Terrence J. Sejnowski

2010
The dissertation of Honghao Shan is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

______________________________________________________________

______________________________________________________________

______________________________________________________________

______________________________________________________________

______________________________________________________________

Chair

University of California, San Diego

2010
DEDICATION

To my parents.
EPIGRAPH

All our knowledge has its origin in our perceptions.

—Leonardo da Vinci
# TABLE OF CONTENTS

Signature Page ............................................................... iii

Dedication ................................................................. iv

Epigraph ................................................................. v

Table of Contents ....................................................... vi

List of Figures ........................................................ viii

List of Tables ........................................................ xiii

Acknowledgements ..................................................... xiv

Vita ................................................................. xv

Abstract of the Dissertation ........................................ xvi

Chapter 1 Introduction ................................................... 1
  1.1 Efficient Coding ................................................... 3
  1.2 Thesis Outline .................................................... 5

Chapter 2 Recursive ICA ................................................ 7
  2.1 Introduction ......................................................... 7
  2.2 Bayesian Explanation of Linear Efficient Encoding ......... 8
  2.3 Reducing Residual Redundancy .................................. 12
    2.3.1 Discarding the Signs ........................................ 13
    2.3.2 Nonlinear Activation Function ............................ 13
  2.4 Experiments on Natural Images .................................. 15
  2.5 Discussion ....................................................... 18

Chapter 3 RICA For Pattern Recognition ........................... 20
  3.1 Introduction ......................................................... 20
  3.2 Theory of Low-level Human Vision ............................. 24
    3.2.1 The Efficient Coding Hypothesis .......................... 24
    3.2.2 Linear Efficient Coding .................................... 25
    3.2.3 Before Linear Efficient Coding ........................... 26
    3.2.4 After Linear Efficient Coding ............................. 26
  3.3 Methods .......................................................... 27
    3.3.1 Learning Visual Features ................................... 27
    3.3.2 Experiments on Yale Face Dataset ....................... 32
    3.3.3 Experiments on ORL Face Dataset ....................... 33
LIST OF FIGURES

Figure 1.1: Visual information is transformed into neural signals at the retina. After that, it is passed by the lateral geniculate nucleus (LGN) to the primary visual cortex (V1). Then along the ventral visual pathway, visual information is passed to V2, V4, and the posterior and anterior inferior temporal cortex (PIT and AIT). Some neurons in PIT and AIT respond specifically to certain objects. The figure is taken from (Thorpe and Fabre-Thorpe, 2001). ................................................................. 2

Figure 1.2: Visual features learned from grayscale image patches by the sparse coding algorithm. ................................................................. 5

Figure 2.1: The RICA (Recursive ICA) model. After the first layer of linear efficient encoding, sensory inputs \( x \) are now represented by \( s \). The signs of \( s \) are discarded. Then coordinate-wise nonlinear activation functions \( g_i \) are applied to each dimension of \( s \), so that the input of the next layer \( x' = g(|s|) \) satisfies the input constraints imposed by the LEE model. The statistical structure among dimensions of \( x' \) are then extracted by the next layer of linear efficient encoding. ................................................................. 9

Figure 2.2: The distribution of pixel values of whiten images follows a generalized Gaussian distribution (see Section 2.2). The shape parameter of the distribution is about 1.094, which means that the marginal distribution of the inputs to the LEE model is already very sparse. ................................................................. 12

Figure 2.3: Left: \( s_1 \) and \( s_2 \) are ICA filter responses on grayscale natural image patches. The red dashed lines plot the linear regression between them. Right: After the coordinate-wise nonlinear transformation, the two features are no longer uncorrelated. . . 13

Figure 2.4: A subset of the 397 ICA image basis functions. Each basis function is 20 \( \times \) 20 pixels. They are 2D Gabor like filters. . . . 16
Figure 2.5: Sample units from the second layer. The upper panel arranges
the connection weights from layer-2 units to layer-1 units by
the centers of the fitted Gabor filters. Every point corresponds
to one basis function of the first layer, located at the center of
the fitted Gabor filter. Warm colors represent strong positive
connections; cold colors represent negative connections. For
example, the leftmost unit prefers strong activation of layer-1
units located on the right and weak activation of layer-1 units
on the left. The lower panel arranges the connection weights by
the frequencies and the orientations of the fitted Gabor filters.
Now every point corresponds to the Gabor filter’s frequency
and orientation (in polar coordinates). The third leftmost unit
welcomes strong activation of Gabor filters whose orientations
are around $\frac{3}{4}\pi$ but prefers no/little activation from those whose
orientations are around $\frac{1}{4}\pi$.

Figure 2.6: Activation maps on two images (upper and lower panel respec-
tively) for two units per layer. The leftmost two images are the
raw images. The second left column to the rightmost column
are activation maps of two units from the first layer to the third
respectively. The first layer units respond to small local edges,
the second layer units respond to larger borders, and the third
layer units seem to respond to large area of textures.

Figure 3.1: A handwritten euro (the European currency) symbol.

Figure 3.2: Applying PCA on image patches. For each 512 $\times$ 512 image on
the top row, we sample 255,025 8 $\times$ 8 image patches and apply
PCA to them. The top 49 eigenvectors are shown in the lower
panels.

Figure 3.3: One actual activation function $G : F$ learned from natural image
patches. The red dot indicates where $F(|s_j|) = 0.5$, i.e., half
of the $s_j$’s responses in natural image patches are smaller than
this value.

Figure 3.4: 128 basis functions (i.e., columns of $A$) learned on 8 $\times$ 8 natural
image patches. See Section 3.3.1 for the learning procedure.
Each columns of $A$ is reconstructed to the original image space
by reversing the PCA projection.

Figure 3.5: The empirical cdf curve of one $|s_j|$ as well as the fitted cdf
function. The fitted parameters for this curve are $\theta = 0.7027
$ and $\tau = 0.1119$.

Figure 3.6: The pixellation effect. The left figure is the original 32 $\times$ 32
version. The right figure is generated by down-scaling the 64$\times$64
image to 32 $\times$ 32 using the imresize matlab function.
Figure 4.1: The elementary features (the columns of $A$) and the corresponding filters (the rows of $W$) learned from $16 \times 16$ grayscale image patches, using Vincent et al.'s retinal coding model (code courtesy of Ben Vincent).  

Figure 4.2: One grayscale image from van Hateren natural image dataset, before and after applying the cone nonlinearity in Eq (4.10).  

Figure 4.3: Experiments on grayscale image patches. Figure 4.3(a) plots how the objective function changes during learning. The black dash-dotted line plots the objective function using the direct optimization method; the blue dashed line plots the objective function when we use the approximate method to find $A$; the red solid line plots the objective function when we further fine tune the parameters using the direct method after initializing $A$ with the approximate method. Figure 4.3(b) plots the eigenvalues of $C$ from the original data (the blue solid line) versus those from the reconstructed data (the red dashed line). Our model captures 99.23% of the variance that could be captured by an optimal linear model with 100 output neurons. Figure 4.3(c) and 4.3(d) plot the distribution of the connection weights in $A$, learned by our model versus by the standard PCA which normalizes each principal component to have unit variance. 96.31% of the connection weights in our model are absolute zero, whereas none of the connection weights by the standard PCA is zero.  

Figure 4.4: $W$ approximates the model neurons' receptive fields. In Figure 4.4(a), we compare $Wx$ with the optimal $s$ inferred using the direct method during the experiments on grayscale image patches. If $s = Wx$ holds exactly, all the points should lie on the red solid line. Once we have learned the optimal $A$ on grayscale image patches, we feed the model with white noise inputs and infer their optimal outputs, then use $\langle xs^t \rangle$ to estimate the model neurons' receptive fields. The top panel in Figure 4.4(b) displays three rows of $W$; the bottom panel displays the estimated receptive fields.
Figure 4.5: Elementary features and their corresponding optimal filters, learned from $20 \times 20$ grayscale image patches. In Figure 4.5(a), the top panel displays 10 features randomly selected from all the 100 features; the bottom panel displays their optimal filters. We fit the features with Gaussian blobs and plot them as circles in Figure 4.5(b). The center and the radius of each circle represent the center and twice the standard deviation of the fitted Gaussian blob. To help visualize how well the Gaussian blobs fit with the features, we display the first feature in Figure 4.5(a) and highlight its fitted Gaussian blob. These Gaussian blobs will become bigger if we use a smaller number of features to “construct” the image patches, as shown in Figure 4.5(d). Figure 4.5(c) displays the first filter in Figure 4.5(a) as a weighted sum of all the 100 features. Each feature is plotted as a circle, as in Figure 4.5(b) and 4.5(d). The color of each circle represents the weight assigned to this feature (read the main text for details).

Figure 4.6: Experiments on white/pink noise images. Figure 4.6(a) and 4.6(b) display images containing white noise and pink noise. The top panel in Figure 4.6(c) displays 10 filters learned from white noise images; the bottom panel plays the filters learned from pink noise images.

Figure 4.7: Experiments with increased sparseness level $\lambda$. We plot the amplitudes of different frequency component of the filters learned with $\lambda = 0.004$ and $\lambda = 0.02$. As shown in the figure, with an increased $\lambda$ value, the filter becomes less sensitive to low frequencies, but slightly more sensitive to high frequencies.

Figure 4.8: One image from the Kyoto chromatic image dataset, before and after applying the cone nonlinearity. The original image is stored in sRGB color representation. We transform it to the LMS cone representation, then apply the nonlinearity in Eq (4.10). To visualize the image after the cone nonlinearity, we transform the image back to sRGB color representation.
Figure 4.9: Elementary features and the corresponding filters learned from 20 × 20 chromatic image patches with $\lambda = 0.001$. In Figure 4.9(a), the left panel displays 6 representative features; the right panel displays their corresponding filters. The features belong to three categories: black/white blobs, blue/yellow blobs, and red/green blobs. The corresponding optimal filters are center-surround shaped, with black/white, blue/yellow, or red/green antagonism. Figure 4.9(b) plots the connection strength from the filters to the L, M, S cones. Figure 4.9(c), 4.9(d), and 4.9(e) plot the spatial layout of learned features, as we did in Figure 4.5(b).

Figure 4.10: one filter learned from grayscale video.

Figure 4.11: Video features segregate into two groups: those with low-spatial and high-temporal frequencies, and those with high-spatial and low-temporal frequencies.

Figure 4.12: Figure 4.12(a) and 4.12(b) plot the revcor filters estimated from cat’s auditory nerve fibers using the linear reverse correlation technique, as well as the fitted gammatone filter. Figure 4.12(c) and 4.12(d) plot filters learned from the TIMIT speech dataset and the Pittsburgh environmental sound dataset respectively. The blue line plots the estimated filter; the red line plots the fitted gammatone filter.
LIST OF TABLES

Table 3.1: Recognition error rates (in percentage) on the Yale dataset, using \( G(|s|) \) as features. \( D \) denotes the dimensionality of \( s \). \( M \) denotes the number of training images selected from each individual. . . 33

Table 3.2: Recognition error rates (in percentage) on the ORL dataset, using \( G(|s|) \) as features. \( D \) denotes the dimensionality of \( s \). \( M \) denotes the number of training images selected from each individual. The last two rows are from CVPR 2007 papers . . . . . 34

Table 3.3: Recognition error rates (in percentage) on the MNIST dataset, using \( G(|s|) \) as features. \( D \) denotes the dimensionality of \( s \). \( M \) denotes the number of training images selected from each digit. . 35

Table 3.4: Recognition error rates (in percentage) on the Yale dataset, without using the nonlinear activation functions. The bottom three rows replicate the first three of Table 3.1 to ease comparison. . . 36

Table 3.5: Recognition error rates (in percentage) on the Yale dataset, using \( G(|s|) \) as features. However, now we are directly using the 32\times32 version images downloaded from the first author of (Cai et al., 2007). . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 37
ACKNOWLEDGEMENTS

I owe great thanks to my Ph.D. advisor, Professor Garrison Cottrell. He guided me through my Ph.D. journey with his great patience. I benefited a lot from his support, advice, and encouragement. In addition to being an advisor, he has been a friend and a father to me.

I am grateful to Professor Marian Stewart Bartlett, Serge J. Belongie, Sanjoy Dasgupta, Terrence J. Sejnowski for taking the time and energy to serve on my Ph.D. committee. Sanjoy pointed me to researches in ICA. Marni is the expert that I turn for advice on ICA. Serge lent me his hands during my down time. Terry shared his lab resources and funded part of my Ph.D. study.

I would like to thank the GURU members, Matthew Tong, Eric Wiewiora, Rosie Cowell, Christopher Kanan, Wensong Xu, Cory Rieth, Tomoki Tsuchida, Pouya Bozorgmehri, Tim Marks, Jonathan Nelson, for discussions and suggestions.

I want to thank Lingyun Zhang, my best friend at UCSD. She is everything one can expect from a friend and more. I also want to thank my other friends that I can not all name individually. They offer me priceless friendship that I will always cherish.

Lastly, and most importantly, I wish to thank my parents, for their love, support and understanding. To them I dedicate this dissertation.

Chapter 2, in part, is a reprint of the paper published in NIPS 2006 “Recursive ICA”, co-authored with Lingyun Zhang and Garrison W. Cottrell. The dissertation author is the primary investigator and author of the paper.

Chapter 3, in part, is a reprint of the paper published in CVPR 2008 “Looking around the backyard helps to recognize faces and digits”, co-authored with Garrison W. Cottrell. The dissertation author is the primary investigator and author of the paper.

Chapter 4, in part, is a reprint of the paper in preparation “Notes on a retinal coding model”, co-authored with Matthew H. Tong and Garrison W. Cottrell. The dissertation author is the primary investigator and author of the paper.
VITA

1998  B.S. in Computer Science, Peking University
2001  M.S. in Computer Science, Chinese Academy of Sciences
2010  Ph.D. in Computer Science & Cognitive Science, University
       of California, San Diego

PUBLICATIONS


In preparation.

In preparation.


Shan, H., & Cottrell, G. W. (2008). Looking around the backyard helps to recog-
nize faces and digits. Proceedings of IEEE Conference on Computer Vision and
Pattern Recognition.

neural information processing systems (pp. 1273–1280). Cambridge, MA, USA:
MIT Press.
ABSTRACT OF THE DISSERTATION

Computational Models of Early Visual Processing Layers

by

Honghao Shan

Doctor of Philosophy in Computer Science and Cognitive Science

University of California, San Diego, 2010

Professor Garrison W. Cottrell, Chair

Visual information passes through layers of processing along the visual pathway, such as retina, lateral geniculate nucleus (LGN), primary visual cortex (V1), prestiate cortex (V2), and beyond. Understanding the functional roles of these visual processing layers will not only help to understand psychophysical and neuroanatomical observations of these layers, but also would help to build intelligent computer vision systems that exhibit human-like behaviors and performance. One of the popular theories about the functional role of visual perception, the efficient coding theory, hypothesizes that the early visual processing layers serve to capture the statistical structure of the visual inputs by removing the redundancy in the visual outputs. Linear implementations of the efficient coding theory, such as independent component analysis (ICA) and sparse coding, learn visual features
exhibiting the receptive field properties of V1 simple cells when they are applied to grayscale image patches.

In this dissertation, we explore different aspects of the early visual processing layers by building computational models following the efficient coding theory.

(1) We develop a hierarchical model, Recursive ICA, that captures nonlinear statistical structures of the visual inputs that cannot be captured by a single layer of ICA. The model is motivated by the idea that higher layers of the visual pathway, such as V2, might work under similar computational principles as the primary visual cortex. Hence we apply a second layer of ICA on top of the first layer ICA outputs. To allow the second layer of ICA to better capture nonlinear statistical structures, we derive a coordinate-wise nonlinear activation function that transforms the first layer ICA’s outputs to the second layer ICA’s inputs. When applied to grayscale image patches, the model’s second layer learns nonlinear visual features, such as texture boundaries and shape contours.

We apply the above model to natural scene images, such as forest and grassland, to learn some generic visual features, and then use these features for face and handwritten digit recognition. We get higher recognition rates than those systems built with features designed for face and digit recognition.

(2) We show that retinal coding, the pre-cortical stage of visual processing, can also be explained by the efficient coding theory. The retinal coding model turns out to be another variation of Sparse PCA, a technique widely applied in signal processing, financial analysis, bioinformatics, etc. Compared with ICA, which removes the redundancy among the input dimensions, Sparse PCA removes redundancy among the input samples. We apply Sparse PCA to grayscale images, chromatic images, grayscale videos, environmental sound, and human speech, and learn visual and auditory features that exhibit the filtering properties of retinal ganglion cells and auditory nerve fibers. This work suggests that the pre-cortical stages of visual and auditory pathway might work under similar computational principles.
Chapter 1

Introduction

Vision constitutes an important portion of the brain functionality. Much of our knowledge comes from our visual perception; many of our actions and decisions are made in response to our visual inputs. Meanwhile, vision seems to share the same underlying neural mechanisms as other sensory modalities. For example, the auditory cortex in the deaf is re-organized for visual processing (Finney et al., 2001). Hence, studying visual processing will likely help us to understand the cortical processing in general.

Visual information passes through layers of processing in the visual pathway. As shown in Figure 1.1 (taken from (Thorpe and Fabre-Thorpe, 2001)), visual information is transformed into neural signals at the retina, then passed through lateral geniculate nucleus (LGN) to the primary visual cortex (V1). After that, visual information is processed along the ventral pathway through V2 and V4 until it finally reaches the inferior temporal cortex, where neurons respond specifically to certain objects.

In this thesis, we explore the functional roles of the visual processing layers. This study serves two purposes. First, recognizing the functional roles of the visual processing layers helps to understand their neuroanatomical properties. As Barlow put in his seminal work half a century ago (Barlow, 1961), “the anatomical structure of wings would be a mystery if one did not know that birds flew.” Second, this study might help to build intelligent computer vision systems by borrowing ideas learned from the human visual system.
Figure 1.1: Visual information is transformed into neural signals at the retina. After that, it is passed by the lateral geniculate nucleus (LGN) to the primary visual cortex (V1). Then along the ventral visual pathway, visual information is passed to V2, V4, and the posterior and anterior inferior temporal cortex (PIT and AIT). Some neurons in PIT and AIT respond specifically to certain objects. The figure is taken from (Thorpe and Fabre-Thorpe, 2001).

We focus our interest on the processing layers that appear early in the visual pathway, such as retinal ganglion cells (RGC), lateral geniculate nucleus (LGN), primary visual cortex (V1). These layers capture simpler visual features than later layers, and they receive less top-down influence. As a result, neuroscientists have collected more details of the early visual processing layers, which makes it easier to decide how well our models match with the actual visual system. Besides, studying the early visual processing layers will lay a solid ground for studying higher visual layers, because the latter receive inputs from the early processing layers.

We explore the functional roles of the early visual processing layers by building computational models that follow the efficient coding theory, a popular theory of visual perception.
1.1 Efficient Coding

It has long been hypothesized that visual perception serves to capture statistical structure of the visual inputs (see (Barlow, 2001b,a) for brief review). However, we need to answer two questions before turning this hypothesis into a quantitative study: (1) what’s the quantitative definition of “statistical structure”? (2) what does it mean to “capture” the statistical structure?

Attneave (1954) provided an answer to the first question. He noticed that how much structure we perceive from an image depends on how well we can predict a removed part of the image by observing its remaining parts. Hence he proposed to use the maximal information, or the redundancy, of the inputs as the measurement of the statistical structure in the inputs. Let $x \in \mathbb{R}^L$ denote the input samples, the maximal information is defined as (Cover and Thomas, 2006):

$$I(x) = \left( \sum_i H(x_i) \right) - H(x)$$

where $H$ denotes the entropy. The redundancy is minimized to zero when $x$’s dimensions are statistically independent.

Barlow (1961) further hypothesized that one possible way for the sensory system to capture the statistical structure of the inputs is to remove the redundancy in its outputs, because to do so the sensory system must have a complete knowledge of the statistical structure in the inputs. This hypothesis was later referred to as the redundancy reduction principle or the efficient coding theory. Although the name of the theory appears to emphasize on the economy of the sensory code, its essence is still about capturing the statistical structure of the sensory inputs. As Barlow (2001a) suggested later, this theory might better be named as “redundancy exploitation”.

Linear implementations of the efficient coding theory, such as independent component analysis (ICA) and sparse coding, have been used to explain the functional roles of V1 simple cells, the first stage of cortical visual processing (Bell and Sejnowski, 1997; Olshausen and Field, 1996). These models can be best described by a generative model, in which the observed input vector $x \in \mathbb{R}^L$ is assumed to
be a linear mixture of underlying signals $s \in \mathbb{R}^M$ with additive Gaussian noise $\epsilon$:

$$x = As + \epsilon$$

where $A \in \mathbb{R}^{L \times M}$ denote the dictionary of elementary features (columns of $A$) that the model uses to generate the observations. There are two assumptions imposed on the underlying signals. First, the underlying signals are statistically independent ($p(s) = \prod_i p(s_i)$). Second, the marginal distribution of the signals $p(s_i)$ follow the sparse distributions, which are characterized by a peak at zero and two heavy tails on both sides of zero. The independence assumption integrates the efficient coding theory into the model. The sparseness is desirable because (1) it allows the neural system to easily assign actions to the corresponding inputs (Barlow, 1961); (2) it save metabolic cost (Levy and Baxter, 1996); (3) many real-world signals do follow the sparse distribution (Lewicki and Sejnowski, 2000).

When there is no noise and $x$ and $s$ have the same number of dimensions, maximizing the model likelihood leads to the ICA algorithm (Bell and Sejnowski, 1997). When the additive noise are small, we can instead minimize the following objective function (Olshausen, 1996):

$$E = \left\langle \frac{\|x - As\|^2}{2} - \lambda \log p(s) \right\rangle$$

where $\langle \cdot \rangle$ denotes taking average over the observed inputs; $\lambda$ denotes the variance of the additive noise. When $p(s_i)$ is the Laplacian distribution, the above objective function can be written as:

$$E = \left\langle \frac{\|x - As\|^2}{2} + \lambda |s|_1 \right\rangle$$

As shown in Figure 1.2, when the above algorithm is applied to grayscale images of natural scenes, it learns visual features that resemble the receptive field properties of V1 simple cells (Olshausen and Field, 1996). Besides V1 simple cells, computational models following the efficient coding theory has also been used to explore the functional roles of other visual processing layers. For example, Atick and Redlich (1992) proposed that retinal coding might serve to capture the second-order statistical structure of the visual inputs. Recently, hierarchical
models following the efficient coding theory have been proposed to explore the functional roles of higher visual processing layers (Hyvarinen and Hoyer, 2001; Osindero et al., 2005; Karklin and Lewicki, 2005, 2009).

1.2 Thesis Outline

In the following thesis, we present some computational models that follow the efficient coding model.

In Chapter 2, we develop a hierarchical model, Recursive ICA (or RICA in short), that captures nonlinear statistical structures of the visual inputs that cannot be captured by a single layer of ICA. The model is motivated by the idea that higher layers of the visual pathway, such as V2, might work under similar computational principles as the primary visual cortex. Hence we apply a second layer of ICA on top of the first layer ICA outputs. To allow the second layer of ICA to better capture nonlinear statistical structures, we derive a coordinate-wise nonlinear activation function that transforms the first layer ICA’s outputs to the second layer ICA’s inputs. When applied to grayscale image patches, the model’s second layer learns nonlinear visual features, such as texture boundaries and shape.
In Chapter 3, we apply the RICA model to natural scene images, such as forest and grassland, to learn some generic visual features, and then use these features for face and handwritten digit recognition. We get higher recognition rates than those systems built with features designed for face and digit recognition.

In Chapter 4, we show that retinal coding, the pre-cortical stage of visual processing, can also be explained by the efficient coding theory. The retinal coding model turns out to be another variation of Sparse PCA, a technique widely applied in signal processing, financial analysis, bioinformatics, etc. Compared with ICA, which removes the redundancy among the input dimensions, Sparse PCA removes redundancy among the input samples. We apply Sparse PCA to grayscale images, chromatic images, grayscale videos, environmental sound, and human speech, and learn visual and auditory features that exhibit the filtering properties of retinal ganglion cells and auditory nerve fibers. This work suggests that the pre-cortical stages of visual and auditory pathway might work under similar computational principles.
Chapter 2

Recursive ICA

2.1 Introduction

Linear implementations of Barlow’s efficient encoding hypothesis\(^1\), such as ICA (Bell and Sejnowski, 1997) and sparse coding (Olshausen and Field, 1996), have been used to explain the very first layers of auditory and visual information processing in the cerebral cortex (Bell and Sejnowski, 1997; Olshausen and Field, 1996; Lewicki, 2002). Nevertheless, many interesting structures are nonlinear functions of the stimulus inputs, which are unlikely to be captured by a linear model. For example, for natural images, it has been observed that there is still significant statistical dependency between the variance of the filter outputs (Schwartz and Simoncelli, 2001). Several extensions of the linear ICA algorithm (Hyvarinen and Hoyer, 2001; Wainwright and Simoncelli, 2000; Karklin and Lewicki, 2005; Osindero et al., 2005) have been proposed to reduce such residual nonlinear redundancy, with an explicit or implicit aim of explaining higher perceptual layers, such as complex cells in V1. However, none of these extensions are obviously recursive, so it is unclear how to generalize them to multi-layer models in order to account for even higher perceptual layers.

In this paper, we propose a hierarchical redundancy reduction model in which the problem of modeling the residual nonlinear dependency is transformed\(^1\)We refer to such algorithms as linear efficient encoding (LEE) algorithms throughout this chapter.
into another LEE problem, as illustrated in Figure 2.1. There are at least two reasons why we want to do this. First, this transforms a new and hard problem into an easier and previously solved problem. Second, different parts of the brain share similar anatomical structures and it is likely that they are also working under similar computational principles. For example, fMRI studies have shown that removal of one sensory modality leads to neural reorganization of the remaining modalities (Finney et al., 2001), suggesting that the same principles must be at work across modalities. Since the LEE model has been so successful in explaining the very first layer of perceptual information processing in the cerebral cortex, it seems reasonable to hypothesize that higher layers might also be explained by a LEE model.

The problem at hand is then how to transform the problem of modeling the residual nonlinear dependency into a LEE problem. To achieve this goal, we need to first make clear what the input constraints are that are imposed by the LEE model. This is done in Section 2.2. After that, we will derive the transformation function that “prepares” the output of ICA for its recursive application, and then test this model on natural images.

2.2 Bayesian Explanation of Linear Efficient Encoding

It has long been hypothesized that the functional role of perception is to capture the statistical structure of the sensory stimuli so that appropriate action decisions can be made to maximize the chance of survival (see (Barlow, 2001a) for a brief review). Barlow provided the insight that the statistical structure is measured by the redundancy of the stimuli and that completely independent stimuli cannot be distinguished from random noise (Barlow, 1961). He also hypothesized that one way for the neural system to capture the statistical structure is to remove the redundancy in the sensory outputs. This so-called redundancy reduction principle forms the foundation of ICA algorithms.

Algorithms following the sparse coding principle are also able to find in-
Figure 2.1: The RICA (Recursive ICA) model. After the first layer of linear efficient encoding, sensory inputs $x$ are now represented by $s$. The signs of $s$ are discarded. Then coordinate-wise nonlinear activation functions $g_i$ are applied to each dimension of $s$, so that the input of the next layer $x' = g(|s|)$ satisfies the input constraints imposed by the LEE model. The statistical structure among dimensions of $x'$ are then extracted by the next layer of linear efficient encoding.
teresting structures when applied to natural image patches (Olshausen and Field, 1996). Later it was realized that although ICA and sparse coding algorithms started out from different principles and goals, their implementations can be summarized in the same Bayesian framework (Lewicki and Olshausen, 1999). In this framework, the observed data $x$ is assumed to be generated by some underlying signal sources $s$:

$$x = As + \epsilon$$

where $A$ is a linear mixing matrix and $\epsilon$ is additive Gaussian noise. Also, it is assumed that the features $s_j$ are independent from each other, and that the marginal distribution of $s_j$ is sparse. For the sparse coding algorithm described in (Olshausen and Field, 1996), although it started from the goal of finding sparse features, the algorithm’s implementation implicitly assumes the independence of $s_j$’s. For the infomax ICA algorithm (Bell and Sejnowski, 1997), although it aimed at finding independent features, the algorithm’s implementation assumes a sparse marginal prior ($p(s_j) \propto \text{sech}(s_j)$). The energy-based ICA algorithm using a student-t prior (Teh et al., 2003) can also be placed in this framework for complete representations.

The moral here, though, is that in practice, the samples available are always insufficient to allow any efficient inference without making some assumptions about the data distribution. A sparseness and independence assumption about the data distribution is appropriate because: (1) independence allows the system to capture the statistical structure of the stimuli, as described above, and (2) sparse distribution of the sensory outputs is energy-economic. This is important for the survival of the biological system, considering the fact that human brain consists 2% of the body weight but accounts for 20% of its resting metabolism (Attwell and Laughlin, 2001). The linear efficient encoding model captures the important characteristics of sensory coding: capturing the statistical structure (independence) of sensory stimuli with minimum cost (sparseness).

This generative model describes our assumption about the data. How well the algorithms perform depends on how well this assumption matches the real data. Hence, it is very important to check what kind of data the model generates. If the input data strongly deviate from what can be generated by the model (in other
words, the observed data strongly deviate from our assumption), the results could be errant no matter how much effort we put into the model parameter estimation. As to the LEE model, there is a clear constraint on the marginal distribution of $x_i$.

Here we limit our study on those ICA algorithms that produce basis functions resembling the simple cells’ receptive fields when applied on natural image patches. Such algorithms (Bell and Sejnowski, 1997; Teh et al., 2003; Hyvarinen and Oja, 1997) typically adopt a symmetric \(^2\) and sparse marginal prior for $s_j$’s that can be well approximated by a generalized Gaussian distribution. In fact, if we apply linear filters resembling the receptive fields of simple cells on natural images, the distribution of the filter responses can be well approximated by a generalized Gaussian distribution.

Here we show that such a prior suggests that the $x_i$’s should also be symmetric. A random variable $x$ is symmetric if and only if its characteristic function is real valued. In the above Bayesian framework, we assume that $s_j$’s are independent and the marginal distribution of $s_j$ is symmetric about zero. The characteristic function is then given by:

$$E[e^{\sqrt{-1}tx_i}] = E[e^{\sqrt{-1}t \sum_j a_{i,j}s_j}] = E[\prod_j e^{\sqrt{-1}t(a_{i,j}s_j)}] = \prod_j E[e^{\sqrt{-1}t(a_{i,j}s_j)}]$$

Since $a_{i,j}s_j$ is symmetric, it is easy to see that $x_i$ must also be symmetric.

A surprising fact about our perceptual system is that there does exist such a process that regularizes the marginal distribution of the sensory inputs. In the visual system, for example, the data is whitened in the retina and the LGN before transmission to V1. The functional role of this process is generally described as removing pairwise redundancy, as natural images (as well as natural sounds) obey the $1/f$ power law in the frequency domain (Field, 1994). However, as shown in Figure 2.2, it also regulates the marginal distribution of the input to follow a $2p(x)$ is symmetric if $x$ and $-x$ have the same distribution.
Figure 2.2: The distribution of pixel values of whiten images follows a generalized Gaussian distribution (see Section 2.2). The shape parameter of the distribution is about 1.094, which means that the marginal distribution of the inputs to the LEE model is already very sparse.

generalized-gaussian-like distribution\(^3\).

This phenomenon has long been observed. We believe that besides the functional role of removing second-order redundancy, whitening might also serve as a role of formatting the sensory input for the cortex. For example, it has been observed (Bell and Sejnowski, 1997) that without pre-whitening the images, the learned basis functions by ICA do not cover a broad range of spatial frequencies.

In this work, we will make the assumption that the marginal distribution of the inputs to the LEE model is a generalized gaussian distribution, as this enables the LEE model to work more efficiently. Also, as just discussed, at least for sound and image processing, there is an effective way to achieve this neurally.

2.3 Reducing Residual Redundancy

For the filter outputs $s$ of a layer of LEE, we will first discard information that provides no interesting structure (i.e., redundancy), and find an activation function such that the marginal distribution obeys the input requirements of the next layer.

\(^3\)For all the image patches we tried, the distribution of pixel values on whitened image patches can be well fitted by a generalized Gaussian distribution. This is true even for small image patches. The only exception we have discovered occurs when the original image contains only binomial noise.
2.3.1 Discarding the Signs

It has been argued that the signs of the filter outputs do not carry any redundancy (Hyvarinen and Hoyer, 2001). The models proposed in (Wainwright and Simoncelli, 2000; Karklin and Lewicki, 2005; Osindero et al., 2005) also implicitly or explicitly discard the signs. We have observed the usefulness of this process in a study of natural image statistics. We applied the FastICA algorithm (Hyvarinen and Oja, 1997) to 20x20 natural image patches, and studied the joint distribution of the filter outputs. As shown in the left plot of Figure 2.3, \( p(s_i | s_j) = p(s_i \mid -s_j) \), i.e. the conditional probability of \( s_i \) on \( s_j \) only depends on the absolute value of \( s_j \). In other words, the signs of \( s \) do not provide any dependency among the dimensions. By removing the sign and applying our transformation (described in the next section), the nonlinear dependency between the \( s_i \)'s is exposed (see Figure 2.3, right).

2.3.2 Nonlinear Activation Function

The only problem left is to find the coordinate-wise activation function \( g_i \) for each dimension of \( s \) such that \( x'_i = g_i(|s_i|) \) follows a generalized Gaussian distribution, as required by the next layer of LEE. In this work, we make the transformed features have a normal distribution. By doing so, we force the LEE
model of the higher layer to set more $a'_{ij}$ to nonzero values (so that the Central Limit Theorem takes effect to make $x'_i$ a Gaussian distribution), which leads to more global structures at the higher layer. We used two methods to find this activation function in our experiments.

**Parametric Activation Function**

Assume $s$ approximately follows a generalized Gaussian distribution (GGD). The probability density function of a GGD is given by:

$$f(s; \sigma, \theta) = \frac{\theta}{2\sigma \Gamma(1/\theta)} \exp\left\{ -\frac{|s|}{\sigma}^{\theta} \right\} \quad (2.4)$$

where $\sigma > 0$ is a scale parameter and $\theta > 0$ is a shape parameter and $\Gamma$ denotes the gamma function. These two parameters can be estimated efficiently by an iterative algorithm developed by (Song, 2006).

$s$ is then transformed into a normally distributed $N(0, 1)$ random variable by the function $g$:

$$u = g(|s|) = F^{-1}\left( \frac{\gamma\left(\frac{|s|}{\sigma^{\theta}}, \frac{1}{\theta}\right)}{\Gamma\left(\frac{1}{\theta}\right)} \right) \quad (2.5)$$

where $F$ denotes the cumulative density function (cdf) of standard normal distribution and $\gamma$ denotes the incomplete gamma function. This transformation can be seen as three consecutive steps:

- Discard the sign: $u \leftarrow |s|$, now $u$ bears pdf $\frac{\theta}{\sigma \Gamma\left(\frac{1}{\theta}\right)} \exp\left\{ -\frac{u^{\theta}}{\sigma} \right\}$, $0 \leq u < \infty$ and cdf $\frac{\gamma\left(\frac{|s|}{\sigma^{\theta}}, \frac{1}{\theta}\right)}{\Gamma\left(\frac{1}{\theta}\right)}$, $0 \leq u < \infty$.

- Transform to a uniform distribution $U[0, 1]$ by applying its own cdf: $u \leftarrow \frac{\gamma\left(\frac{|s|}{\sigma^{\theta}}, \frac{1}{\theta}\right)}{\Gamma\left(\frac{1}{\theta}\right)}$.

- Transform to a Gaussian distribution by applying the inverse cdf of $N(0, 1)$: $u \leftarrow F^{-1}(u)$.

**Nonparametric Activation Function**

When the number of samples $N$ is sufficiently large, a non-parametric activation function works more efficiently. In this approach, all the samples $|s_i|$ are sorted in ascending order. For each sample $s$, cdf($|s|$) is approximated by the ratio
of its ranking in the list with \( N \). Then \( u = F^{-1}(\hat{\text{cdf}}(|s|)) \) will approximately follow the standard normal distribution. Note that since \( u_i \) depends only on the rank order of \(|s_i|\), the results would be the same if the signs are discarded by taking \( s_i^2 \).

### 2.4 Experiments on Natural Images

To test the behavior of our model, we applied it to small patches taken from digitized natural images. The image dataset is available on the World Wide Web from Bruno Olshausen \(^4\). It contains ten \( 512 \times 512 \) pre-whitened images. We took 151, 290 evenly distributed \( 20 \times 20 \) image patches. We ran the FastICA algorithm (Hyvärinen and Oja, 1997) and obtained 397 basis functions. As reported in other models, the basis functions are Gabor-like filters (Figure 2.4).

The nonparametric method was used to transform the marginal distribution of the outputs’ absolute values to a standard normal distribution. Then the FastICA algorithm was applied again to retrieve 100 basis functions\(^5\). We adopted the visualization method employed by (Lewicki and Olshausen, 1999) to investigate what kind of structures the second layer units are fond of. The basis functions are fitted to Gabor filter functions using a gradient descent algorithm (Lewicki and Olshausen, 1999). The connection weights from a layer-2 unit to layer-1 units are shown in Figure 2.5, arranged by either the center or frequency/orientation of the fitted Gabor filters. The layer-2 units are qualitatively similar to those found in (Karklin and Lewicki, 2003). Some units welcome strong activation of layer-1 units within a certain orientation range but have no preference for locations, while others have a location preference but welcome activation of layer-1 units of all frequencies and orientations, and some develop a picky appetite for both.

Again, the nonparametric method was used to transform the marginal distribution of the absolute values of the outputs from the second layer to a standard normal distribution, and FastICA was applied to retrieve 20 basis functions. We

\(^4\)http://redwood.berkeley.edu/bruno/sparseenet/

\(^5\)This reduction in the number of units follows the example of (Karklin and Lewicki, 2003). In general, there appears to be less information in later layers (as assessed by eigenvalue analysis), most likely due to the discarding of the sign.
Figure 2.4: A subset of the 397 ICA image basis functions. Each basis function is 20 × 20 pixels. They are 2D Gabor like filters.

Figure 2.5: Sample units from the second layer. The upper panel arranges the connection weights from layer-2 units to layer-1 units by the centers of the fitted Gabor filters. Every point corresponds to one basis function of the first layer, located at the center of the fitted Gabor filter. Warm colors represent strong positive connections; cold colors represent negative connections. For example, the leftmost unit prefers strong activation of layer-1 units located on the right and weak activation of layer-1 units on the left. The lower panel arranges the connection weights by the frequencies and the orientations of the fitted Gabor filters. Now every point corresponds to the Gabor filter’s frequency and orientation (in polar coordinates). The third leftmost unit welcomes strong activation of Gabor filters whose orientations are around $\frac{3}{4}\pi$ but prefers no/little activation from those whose orientations are around $\frac{1}{4}\pi$.

had no initial guess of what kind of statistical structure these third layer units might capture. The activation map of a couple of these units, however, seemed to suggest that they might be tuned to respond to complicated textures. In particular, one unit seems more activated by seemingly blank background, while another seems to like textures of leaves (Figure 2.6). We think that probably a larger database than merely 10 images, and larger image patches would be helpful for producing cleaner high level units.

The same procedure can be repeated for multiple layers. However, at this point, until we develop better methods for analyzing the representation developed by these deeply embedded units, we will leave this for future work.
Figure 2.6: Activation maps on two images (upper and lower panel respectively) for two units per layer. The leftmost two images are the raw images. The second left column to the rightmost column are activation maps of two units from the first layer to the third respectively. The first layer units respond to small local edges, the second layer units respond to larger borders, and the third layer units seem to respond to large area of textures.
2.5 Discussion

The key idea of our model is to transform the high-order residual redundancy to linear dependency that can be easily exploited again by the LEE model. By using activation functions that are dependent on the marginal distribution of the outputs, a normal Gaussian interface is provided at every layer. This procedure can then repeat itself and a hierarchical model with same structure at every level can thus be constructed. As the redundancy is reduced progressively along the layers, statistical structures are also captured to progressively higher orders.

Our simulation of a three layer Recursive ICA shows the effectiveness of our model. The first layer, not surprisingly, produces the Gabor like basis functions as linear ICA always does. The second layer, however, produces basis functions that qualitatively resemble those produced by a previous hierarchical generative model (Karklin and Lewicki, 2005). This is remarkable given that our model is essentially a filtering model with no assumptions of underlying independent variables, but merely targeting redundancy reduction. The advantage of our model is the theoretical simplicity of generalization to a third layer or more. For the Karklin and Lewicki model, the assumption that the ultimate independent causal variables are two layers away from the images has to be reworked for a three layer system. It is not clear how the variables at every layer should affect the next when an extra layer is added. Osindero et al. (2005) employed an energy based model. The energy function used at the first layer made it essentially a linear ICA algorithm, thus it also produces Gabor like filters. The first layer outputs are squared to discard the signs and then fed to the next layer. The inputs for the second layer are thus all positive and bear a very different marginal distribution from those for the first layer. The energy function is changed accordingly and the second layer is essentially doing nonnegative ICA. The output of this layer, however, will all be positive, which makes discarding the signs no longer an effective way of exposing higher-order dependence. Thus, to extend to another layer, new activation functions and new energy function must be derived.

The third layer of our model produces some interesting results in that some units seem to have preferences for complicated textures (Figure 2.6). However, as
the statistical structure represented here must be of very high order, we are still looking for an effective visualization method. Also, as units at the second layer have larger receptive field than those at the first layer, it is reasonable to expect the third layer to bear even larger ones. We believe that a wider range of visual structure will be picked up by the third layer units with a larger patch size on a larger training set.

Acknowledgments

Chapter 2, in full, is a reprint of the paper published in NIPS 2006 “Recursive ICA”, co-authored with Lingyun Zhang and Garrison W. Cottrell. The dissertation author is the primary investigator and author of the paper.
Chapter 3

RICA For Pattern Recognition

3.1 Introduction

One notable difference between the human visual system and most computer vision systems is that we human beings can learn a new visual category based on a few or even one example of the new category, while most computer systems require a large number of training samples to work reasonably well. For example, we can recognize the euro symbol (shown in Figure 3.1) under a wide variety of visual conditions after having seen a single example. Such an ability is necessary for the wide application of computer vision, as collecting training examples is generally expensive. In fact, due to the curse of dimensionality, almost all real-world pattern recognition problems can be regarded as problems of learning with few examples.

The problem of learning with scarce training samples has been addressed

Figure 3.1: A handwritten euro (the European currency) symbol.
in different ways. We find two of these approaches related to our approach. One approach, namely semi-supervised learning (Blum and Mitchell, 1998), deals with the situation when labeled training samples are scarce by utilizing information from unlabeled samples. This approach reduces the amount of human intervention necessary to build a working system, since unlabeled data are generally plentiful and cheap to collect. It does not help, however, when unlabeled samples are also rare or expensive to collect, whereas human beings still exhibit reasonable performance in these cases. Miller et al. (2000) adopted an alternative approach to address the problem of learning with few examples. They learned a set of transforms from handwritten English letters and then applied them to handwritten digits to regularize the input patterns, which allowed them to build a handwritten digit classifier using only several training examples per class. The limitations of this method, however, are that the forms of the transforms need to be predefined and that the visual knowledge can only be shared between highly similar visual categories.

These two types of techniques share one common characteristic – they both use knowledge obtained from a second information source to help building the classifier. They differ in how they define the knowledge and how the knowledge is learned and transferred, but they both pick a secondary information source which shares some common property with the labeled data, using either unlabeled data belonging to the same category as the labeled data, or labeled data belonging to a different but visually similar category, so that knowledge transfer is plausible.

The human visual system seems to adopt a similar approach. Low-level visual layers, such as retina, LGN (the lateral geniculate nucleus) and V1 (the primary visual cortex), are shared components that process all visual information we perceive. These layers develop and mature during childhood, and provide the basis for all the visual tasks encountered in the rest of life. If we loosely define the term “visual feature” as any function of the image pixel values, the above phenomenon can be interpreted as learning a set of universal visual features from the scenes encountered during childhood. These visual features are later applied to all visual stimuli and are used to perform a variety of visual tasks. Presumably
these visual features provide one way to transfer the knowledge obtained from previous visual experiences and should help to build a classifier when labeled data are rare.

The idea of “universal visual features” might at first sight appear implausible, since this concept suggests that all visual stimuli share some characteristic in common such that knowledge obtained from one set of stimuli can be applied to a completely different set of visual stimuli. What is the common property shared by the appearance of your husband/wife’s face and the view out your kitchen window that would allow you to better recognize the first by simply browsing the latter?

One observation is that they share similar local statistical structure. For example, if we take all of the $255,025$ $8 \times 8$ image patches from each of the two images shown on the top row of Figure 3.1, subtract the local mean from each image patch, and apply PCA to them, the resulting basis functions (i.e., eigenvectors) all resemble DCT filters. In fact, if we apply the PCA projection learned from image patches extracted from one image to the image patches from the other, and calculate the covariance matrix of the projected features, we will see that most off-diagonal cells have much smaller values than the diagonal cells. That is, although the projection matrix is calculated to capture the second-order local structure of one image, it also approximately captures the second-order local structure of the other image. The observation here is that, although the two images display very different visual content, they share very similar local/low-level statistical structure.

In this paper we apply the hypothesis of universal visual features to several classification tasks and achieve state of the art results. We begin by briefly reviewing neuroscience theories of low-level human vision in Section 3.2 from a computational perspective. Then we show in Section 3.3 how to learn a set of visual features from ten randomly collected natural images by simulating visual information processing up to the simple cells, the first layer of visual information processing in the cerebral cortex using ICA. The main difference between our approach and previous approaches is that we apply a non-linearity to the ICA features on a component-wise basis, that converts them from having a sparse distribution to having a gaussian distribution. We then apply PCA to that representation to
Figure 3.2: Applying PCA on image patches. For each $512 \times 512$ image on the top row, we sample $255,025$ $8 \times 8$ image patches and apply PCA to them. The top 49 eigenvectors are shown in the lower panels.
reduce the dimensionality and feed the resulting vectors into a single layer classifier with a softmax activation function. Using this approach, we have achieved recognition performance comparable to recently proposed computer vision techniques on the Yale face dataset, the ORL face dataset, and the MNIST dataset. We then show through an example when the technique will seriously fail, as well as a solution to deal with such cases. We end the paper by discussing possible approaches to further improve this technique.

3.2 Theory of Low-level Human Vision

3.2.1 The Efficient Coding Hypothesis

What is the utility of unsupervised visual feature extraction from images? A similar question has been considered in the neuroscience community for years: what is the functional role of low-level visual layers in the human visual pathway which appear to receive little top-down (i.e., task driven) influence?

One hypothesis is that they capture the statistical structure of sensory inputs so that corresponding high-level decisions can be made accordingly (see (Barlow, 2001a) for a brief review). This hypothesis engenders two questions: (1) how to (quantitatively) define the statistical structure; and (2) how to capture the statistical structure. Attneave (1954) pointed out that whether we perceive structures in an image depends on how well we can predict a missing part of the image by its remaining parts. This insight suggests that we can use the dependency among the input features (i.e., dimensions), or the redundancy of the inputs, as a quantitative measurement of the statistical structure provided by the sensory inputs. Based on this observation, Barlow (1961) hypothesized that one plausible way for a neural system to capture the statistical structure of its inputs was to remove their redundancy in its outputs, because to do so, the neural system must have a complete knowledge about statistical structures contained in its inputs. This hypothesis is later referred to as the redundancy reduction principle or the efficient coding principle. Although the name appears to suggest pursuing an economic coding, its essence is still focused upon capturing the statistical structure of the
sensory inputs.

3.2.2 Linear Efficient Coding

Linear implementations of the efficient coding hypothesis, such as independent component analysis (Bell and Sejnowski, 1997) and sparse coding (Olshausen and Field, 1996), have been used to explain the functional role of simple cells in the primary visual cortex, the first layer of visual information processing in the cerebral cortex. These algorithms are best described by a generative model, in which the observed data \( \mathbf{x} \in \mathbb{R}^d \) is assumed to be generated by linearly mixing underlying signal source \( \mathbf{s} \in \mathbb{R}^D \):

\[
\mathbf{x} = \mathbf{A}\mathbf{s} + \epsilon
\]  

(3.1)

where \( \mathbf{A} \in \mathbb{R}^{d \times D} \) is the linear mixing matrix, \( \epsilon \in \mathbb{R}^d \) denotes additive gaussian noises. The signal sources \( s_j \)'s are assumed to be statistically independent, which incorporates the desire of capturing statistical structure in \( \mathbf{x} \). For natural image statistics studies, a sparse marginal distribution for each \( s_j \) is assumed, which is characterized by a peak at zero and two heavy tails symmetrically residing on both sides of zero, such as the Student-t distribution or the Laplacian distribution. It was argued (Olshausen and Field, 1996) that such a distribution incorporates the need to transfer more information with minimum energy cost, which is very important for a biological system.

There are two optimization problems associated with this model. One is to learn the most probable \( \mathbf{A} \) given \( n \) observations \( \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \) (see (Lewicki and Sejnowski, 2000) on the learning algorithm). Once the optimal \( \mathbf{A} \) is learned, the inference problem is to infer the most probable signal source \( \mathbf{s} \) given the mixing matrix \( \mathbf{A} \) and an observation \( \mathbf{x} \). When the marginal distribution is assumed to be a Laplacian distribution, the inference problem is a convex optimization problem and can be solved efficiently.

If we apply the linear efficient coding algorithms to natural image patches, the resulting basis functions (i.e., columns of \( \mathbf{A} \)) resemble simple cell receptive fields, as shown in Figure 3.4. It is widely observed (Bell and Sejnowski, 1997;
Hyvarinen and Hoyer, 2001; Olshausen and Field, 1996) that the basis functions learned from different image datasets are qualitatively similar, which suggests even though these images display very different global contents they share similar local statistical structure.

3.2.3 Before Linear Efficient Coding

The visual information passes through retina and LGN before reaching the primary visual cortex. What happens there and why?

The classical theory about what happens before V1 is the whitening theory (Field, 1987), which states that retina and LGN serve to flatten the magnitudes of the images on the frequency domain. It was observed that natural images approximately follow the $1/f$ law in the frequency domain. That is, if we apply 2D Fourier transform on a natural image, most of the time we will observe that the magnitude of each component decreases with increasing component frequency. Supposedly the visual system removes the redundant information by flattening the magnitudes on the frequency domains. Later it was pointed out that this operation also approximately makes pixel values uncorrelated (Field, 1994). In a recent paper (Shan et al., 2007) we also noted that this operation regulates the distribution of the inputs to V1 so that the linear efficient coding algorithm can work more efficiently. We showed that when ICA is treated as a generative model, the marginal distributions of the $x_i$’s are zero-mean Gaussians. Hence the whitening process is “formatting” the inputs for the ICA procedure.

3.2.4 After Linear Efficient Coding

A further development in the (Shan et al., 2007) paper was that by applying a component-wise nonlinearity to the absolute value of the $s_j$s (the sign is redundant), one can convert them into Gaussian distributions, and apply ICA again to obtain an efficient encoding of the first layer. This process may be repeated multiple times to obtain a multilayer ICA. We applied this algorithm to develop a two layer ICA, and the resulting second-layer basis functions appeared to code for
texture boundaries and corners. In this paper, we stop at one layer, but apply the nonlinear function to the outputs. We have found this improves performance over simply using linear ICA.

The procedure for computing the nonlinearity is as follows: For each dimension of the layer-1 outputs $s_j$, the cdf $\mathcal{F}$ of $|s_j|$ is estimated. Then the coordinate-wise activation function $G$ was defined as:

$$G(s_j) = \mathcal{G}(\mathcal{F}(|s_j|))$$

where $\mathcal{G}$ denotes the inverse cdf function of a standard normal distribution. This coordinate-wise activation function works to discard the signs of layer-1 outputs, and then transform the marginal distributions to be Gaussian distributions.

Here is an intuitive explanation of the activation functions. The layer-1 basis function can be roughly considered as edge/bar detectors. Taking the absolute values of $s_j$ introduces some shift invariance. Figure 3.3 plots one actual activation function learned from natural images. As shown in the figure, the activation function can be roughly divided into segments. Between zero and the red dots, the activation function amplifies the $|s_j|$ values that are highly peaked near zero from their small range on the $x$-axis to a large range on the $y$-axis. This segment serves to increase the distance between two $|s_j|$ values and may help the classifier to distinguish their differences. When the $|s_j|$ value is bigger than where the red dot indicates, few $|s_j|$’s are for any one image, and the activation function flattens the $s_j$’s.

### 3.3 Methods

#### 3.3.1 Learning Visual Features

We apply the sparse coding algorithm (Olshausen and Field, 1997) on ten 512 × 512 natural images available from Olshausen’s homepage. The images are whitened using the whitening filter described in (Olshausen and Field, 1996), whose matlab code is also available from Olshausen’s homepage. As discussed earlier, this whitening process is supposed to simulate the processing in retina and LGN. We
Figure 3.3: One actual activation function $G \cdot F$ learned from natural image patches. The red dot indicates where $F(|s_j|) = 0.5$, i.e., half of the $s_j$’s responses in natural image patches are smaller than this value.
normalize each image to have zero mean and unit variance. After whitening, six pixels off the boundary are discarded to avoid boundary effects. Each image is now $500 \times 500$ in size. We extract all the $2,430,490$ $8 \times 8$ possible image patches, subtract the local mean from each image patch, then calculate the PCA projection matrix. Now each image patch is represented as a 63 dimensional vector $\mathbf{x}$. While this does not reduce the dimensionality, it appears to make the ICA algorithm learn better features. After the PCA projection, we scale each dimension of $\mathbf{x}$ to have unit variance.

The sparse coding algorithm is applied on these image patches. It corresponds to the linear efficient coding model described in Equation 3.1, with the following marginal prior:

$$p(s_j) \propto \frac{1}{(1 + (s_j/\sigma))^{\beta}}$$

(3.3)

In our experiments, we set $\sigma = 1$ and $\beta = 0.4$. We have found a relatively wide range of these values all work well. The variances of the noise $\epsilon$ (see Equation 3.1) is also set to 1.

We initialize the linear mixing matrix $A$ with gaussian random variables. Then, on each iteration, we randomly pick 100 image patches, and calculate their PCA projections $\mathbf{x}_1, \ldots, \mathbf{x}_{100}$. The mixing matrix $A$ is then updated by:

$$A = A + \frac{\eta}{100} \sum_{i=1}^{100} (\mathbf{x}_i - As_i)s'_i$$

(3.4)

where $\eta$ denotes the learning rate, $s_i$ is the most probable underlying signal given the observation $\mathbf{x}_i$ and the current mixing matrix $A$ (see Olshausen and Field, 1997 for the inference algorithm). After each update, the columns of $A$ are normalized to unit length to speed up the learning process. We repeat this process for 100,000 iterations with $\eta = 0.1$, followed by another 100,000 iterations with $\eta = 0.01$.

To evaluate the effect of overcompleteness (i.e., the ratio between the dimensionality of $s$ and the dimensionality of $\mathbf{x}$) on classification performance, we learn three different sets of features, with the dimensionality of $s$ equal to 64, 128, or 192. Figure 3.4 displays the basis functions learned when the dimensionality of $s$ is set to 128.
Figure 3.4: 128 basis functions (i.e., columns of $A$) learned on $8 \times 8$ natural image patches. See Section 3.3.1 for the learning procedure. Each column of $A$ is reconstructed to the original image space by reversing the PCA projection.

After that, we estimate the cdf function of $|s_j|$ for each dimension. This is done by first inferring the underlying signal representation $s$ for all the 2,430,490 extracted image patches. We calculate the histogram of $|s_j|$ with bins between 0 and 15 and a bin size of 0.0001. The empirical cdf function of $|s_j|$ is generated from this histogram, and then fitted by the following function:

$$F_j(|s_j|) = \Gamma((|s_j|/\tau)\theta, 1/\theta)$$

where $\Gamma$ denotes the incomplete Gamma function. Figure 3.5 displays the empirical cdf function of one $|s_j|$ as well as the fitted cdf function, when the dimensionality of $s$ equals 128. This activation function converts the highly sparse distribution learned via ICA into a Gaussian distribution, which in previous work we have used to apply ICA again. However, in this work, we simply investigate the utility of this nonlinear function applied to one layer of ICA features.

After the linear mixing matrix $A$ and the nonlinear activation functions $G_j = \mathcal{G}(F_j)$ are learned from the ten natural images, we apply them on various classification tasks without any adjustment of the parameters. The classification performance based on these features outperforms many recently proposed computer vision techniques, even if we just use a single layer classifier.
Figure 3.5: The empirical cdf curve of one $|s_j|$ as well as the fitted cdf function. The fitted parameters for this curve are $\theta = 0.7027$ and $\tau = 0.1119$. 
3.3.2 Experiments on Yale Face Dataset

We first test the features on the Yale dataset (Belhumeur et al., 1997), which contains 165 gray-scale images of 15 individuals. Each individual has 11 images. The manually aligned and cropped images can be downloaded from the homepage of the first author of (Cai et al., 2007). We downloaded the $64 \times 64$ processed images, and then downscaled the images to $32 \times 32$ using the imresize matlab function. This procedure does not suffer from the pixellation artifacts that occur in the $32 \times 32$ images provided on the website. Then we whitened each image using the whitening filter discussed in Section 3.3.1 and normalize each image to have zero mean and unit variance. For each image, we extract all the 625 $8 \times 8$ image patches, and infer the most probable $s$ for each image patch. After that, we apply the nonlinear activation $G_j$ on each dimension of $s_j$. When the dimensionality of $s$ is 64 (1 times overcomplete), each facial image is represented by a $625 \times 64 = 40,000$ dimensional vector. For 2 times overcompleteness, each image is represented by a $80,000$ dimensional vector, etc.

We followed the method in (Cai et al., 2007) to randomly divide the images into the training and the testing sets. For each experiment, we randomly select $M = 2, 3, \ldots, 8$ images from each individual as the training images, and use the rest as testing images. For each setting of $M$, we tested 50 random splits.

When the training set and the testing set are selected, we take the 40,000 dimensional representation (for 1 times overcomplete, etc.) and reduce its dimensionality using PCA. The number of principal components are chosen so that 95% of the variance is captured. For example, as $M$ increases from 2 to 8, the number of principal components using this criterion range from 27 to 105. We then use the projected training examples to train a one layer network using the softmax activation function. The network is updated for 1,000 epochs or until the network weights have converged, using the scaled conjugate gradient algorithm (the glm + netopt function in the netlib library implement this algorithm). The test images are projected using the PCA projection matrix trained on the training images, and fed to the one layer network.

Table 3.1 lists the recognition performance on the test images, with different
Table 3.1: Recognition error rates (in percentage) on the Yale dataset, using $G(|s|)$ as features. $D$ denotes the dimensionality of $s$. $M$ denotes the number of training images selected from each individual.

<table>
<thead>
<tr>
<th></th>
<th>$D=64$</th>
<th>$M=2$</th>
<th>$M=3$</th>
<th>$M=4$</th>
<th>$M=5$</th>
<th>$M=6$</th>
<th>$M=7$</th>
<th>$M=8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>28.86</td>
<td>19.55</td>
<td>13.71</td>
<td>9.96</td>
<td>7.71</td>
<td>6.43</td>
<td>4.93</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>27.13</td>
<td>17.78</td>
<td>12.04</td>
<td>8.36</td>
<td>6.96</td>
<td>5.50</td>
<td>4.04</td>
<td></td>
</tr>
<tr>
<td>192</td>
<td>27.11</td>
<td>17.38</td>
<td>11.71</td>
<td>8.16</td>
<td>6.27</td>
<td>5.07</td>
<td>3.82</td>
<td></td>
</tr>
</tbody>
</table>

(Cai et al., 2007) 42.4 27.7 22.2 18.3 - - -
(Hua et al., 2007) - - - 13.2 - - -
(Cai et al., 2007) update 37.5 25.5 19.3 14.7 12.3 10.3 8.7

overcompleteness and different numbers of training images from each individual. The results are extremely good given the small number of training examples. Results reported in CVPR2007 in (Cai et al., 2007) and (Hua et al., 2007) are given for comparison. We noticed that on the homepage of the first author of (Cai et al., 2007), they reported new results. These are given on the last line. However, there is still an obvious gap between their results and our results, and our results are based on first layer features not adapted to the training set.

3.3.3 Experiments on ORL Face Dataset

The Olivetti Research Laboratory (ORL) database (Samaria and Harter, 1994) contains 400 face images of 40 persons, with 10 per person, taken at different time, under different lighting conditions, and with different facial expressions. We downloaded the manually aligned and cropped $64 \times 64$ images from the homepage of the first author of (Cai et al., 2007), and then downsampled them to $32 \times 32$ size using the imresize matlab function. We implemented the same experiments as on the Yale datasets. We randomly select $M = 2, 3, \ldots, 8$ images per person for training and the rest for testing. The average recognition error rates are reported in Table 3.2, with the results from two CVPR 2007 papers for comparison.
Table 3.2: Recognition error rates (in percentage) on the ORL dataset, using \(G(|s|)\) as features. \(D\) denotes the dimensionality of \(s\). \(M\) denotes the number of training images selected from each individual. The last two rows are from CVPR 2007 papers.

<table>
<thead>
<tr>
<th>(D)</th>
<th>(M=2)</th>
<th>(M=3)</th>
<th>(M=4)</th>
<th>(M=5)</th>
<th>(M=6)</th>
<th>(M=7)</th>
<th>(M=8)</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>15.99</td>
<td>8.64</td>
<td>5.08</td>
<td>2.91</td>
<td>2.14</td>
<td>1.40</td>
<td>1.10</td>
</tr>
<tr>
<td>128</td>
<td>15.46</td>
<td>8.20</td>
<td>4.75</td>
<td>2.70</td>
<td>1.78</td>
<td>1.20</td>
<td>0.90</td>
</tr>
<tr>
<td>192</td>
<td>15.11</td>
<td>8.02</td>
<td>4.46</td>
<td>2.43</td>
<td>1.78</td>
<td>1.12</td>
<td>0.90</td>
</tr>
</tbody>
</table>

(Cai et al., 2007) 14.8 7.7 4.2 2.8 - -
(Hua et al., 2007) - - - 3.0 - -

3.3.4 Experiments on MNIST Digit Dataset

In this experiment, we apply our features derived from natural scene statistics to the MNIST handwritten digits dataset. The MNIST dataset has a training set of 60,000 examples and a test set of 10,000 examples. The digits have been size normalized and centered in 28 \(\times\) 28 images. We downsampled each image to 18 \(\times\) 18 size by the imresize matlab function. Each image is whitened and normalized to have zero mean and unit variance. 121 \(8 \times 8\) image patches are extracted from each image for the one times overcomplete case, so each image is represented by a 121 \(\times\) 64 = 7744 dimensional vector. We also give the results for two and three times overcomplete.

We randomly select \(M = 10, 20, 30, 40, 50\) training examples for each digit from the training dataset. For each \(M\), we try 40 different sets of training examples. As with the experiments on face datasets, we apply PCA to reduce the dimensionality while retaining 95% of the variance, and use a one-layer softmax classifier. We test the classifier on the 10,000 test images, and the average recognition error rates are reported in Table 3.3. The bottom row displays the results reported in (Teo et al., 2008).

These results are comparable to recently proposed computer vision techniques. For example, in NIPS2007, the authors of (Teo et al., 2008) reported how to learn a set of kernels which are invariant to 20 transforms: 1-pixel and 2-pixel
Table 3.3: Recognition error rates (in percentage) on the MNIST dataset, using $G(|s|)$ as features. $D$ denotes the dimensionality of $s$. $M$ denotes the number of training images selected from each digit.

<table>
<thead>
<tr>
<th>$D$</th>
<th>$M=10$</th>
<th>$M=20$</th>
<th>$M=30$</th>
<th>$M=40$</th>
<th>$M=50$</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>12.62</td>
<td>8.73</td>
<td>7.26</td>
<td>6.49</td>
<td>5.92</td>
</tr>
<tr>
<td>128</td>
<td>12.20</td>
<td>8.20</td>
<td>6.84</td>
<td>6.08</td>
<td>5.52</td>
</tr>
<tr>
<td>192</td>
<td>12.05</td>
<td>8.12</td>
<td>6.73</td>
<td>5.93</td>
<td>5.32</td>
</tr>
</tbody>
</table>

| Invar-SVM | 14.9 | 8.9 | 7.8 | 5.8 | 5.5 |

shift in 4 and 8 directions, rotations by $\pm 10$ degrees, scaling by $\pm 0.15$, and shearing in vertical or horizontal axis by $\pm 0.15$. SVM based on these kernels yields results shown on the bottom row of Table 3.3. In CVPR2007, the authors of (Ranzato et al., 2007) trained a four-layer network to extract visual features. They used all the 60,000 unlabeled training examples to train the network, and then use a subset of labeled training examples to train a two-layer neural network, using the visual features extracted by the four-layer network. They reported an average error rate of 7.18% on the test dataset, using just 300 training examples, as we do here. However, they achieved much better results when they used more training examples (less than a percent). It remains to be seen how well we can do with more training examples.

3.3.5 Why This Works

Our above experiments raise an interesting question: why should these visual features help classification tasks in general? We have noticed that when we apply the visual features to the images, we in fact map the image to a much higher dimensional space nonlinearly. The nonlinearity comes from two operations. First, given input $x$, the optimal $s$ is inferred by minimizing the reconstruction error $\|x - As\|^2$ while minimizing the sparsity penalty $\log p(s)$. As a result, $s$ is a nonlinear function $x$ when $D > d$. Instead of mapping to a linear subspace of the higher dimensional space, as a linear transform $Wx$ would do, the inference
Table 3.4: Recognition error rates (in percentage) on the Yale dataset, without using the nonlinear activation functions. The bottom three rows replicate the first three of Table 3.1 to ease comparison.

<table>
<thead>
<tr>
<th>D</th>
<th>M=2</th>
<th>M=3</th>
<th>M=4</th>
<th>M=5</th>
<th>M=6</th>
<th>M=7</th>
<th>M=8</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>36.92</td>
<td>26.63</td>
<td>19.37</td>
<td>15.24</td>
<td>12.77</td>
<td>11.00</td>
<td>10.22</td>
</tr>
<tr>
<td>128</td>
<td>36.00</td>
<td>25.72</td>
<td>18.65</td>
<td>13.82</td>
<td>11.87</td>
<td>9.87</td>
<td>9.02</td>
</tr>
<tr>
<td>192</td>
<td>34.00</td>
<td>23.78</td>
<td>16.72</td>
<td>12.49</td>
<td>11.04</td>
<td>8.37</td>
<td>8.31</td>
</tr>
<tr>
<td>64</td>
<td>28.86</td>
<td>19.55</td>
<td>13.71</td>
<td>9.96</td>
<td>7.71</td>
<td>6.43</td>
<td>4.93</td>
</tr>
<tr>
<td>128</td>
<td>27.13</td>
<td>17.78</td>
<td>12.04</td>
<td>8.36</td>
<td>6.96</td>
<td>5.50</td>
<td>4.04</td>
</tr>
<tr>
<td>192</td>
<td>27.11</td>
<td>17.38</td>
<td>11.71</td>
<td>8.16</td>
<td>6.27</td>
<td>5.07</td>
<td>3.82</td>
</tr>
</tbody>
</table>

process spreads the optimal $s$ in the high dimensional space. Another nonlinearity comes from the activation function $G_j(|s_j|)$ we apply on each dimension of $s$. We hypothesize by mapping the data to a higher dimensional space, we can benefit from what kernel SVM’s have benefited from – the samples are much more likely to be linearly separable. However, since the nonlinear functions are learned from natural images and capture some meaningful structure, we suffer little from the problem of over-fitting.

To evaluate the utility of the nonlinear activation function $G$, we repeat the experiments on the Yale dataset. The experiment settings are the same as described in Section 3.3.2, except that now we only keep the absolute values of $s$ without applying the nonlinear activation functions. As shown in the table, the nonlinear activation functions make a significant difference in the final classification performance.

We also tried using Gabor magnitudes (3 scales and 8 orientations, followed by PCA), as in previous work (Dailey et al., 2002) instead of our ICA features with the nonlinear activation function. The results were worse than Cai et al.’s results, and much worse than our results. Hence the ICA features, with the nonlinear activation function, are an important part of the model.

Another observation in all the tables is that generally the higher the dimensionality, the better the recognition performance. This accords with our hypothesis.
Table 3.5: Recognition error rates (in percentage) on the Yale dataset, using $G(\|s\|)$ as features. However, now we are directly using the $32 \times 32$ version images downloaded from the first author of (Cai et al., 2007).

<table>
<thead>
<tr>
<th></th>
<th>M=2</th>
<th>M=3</th>
<th>M=4</th>
<th>M=5</th>
<th>M=6</th>
<th>M=7</th>
<th>M=8</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>48.73</td>
<td>36.73</td>
<td>30.72</td>
<td>24.22</td>
<td>19.73</td>
<td>16.27</td>
<td>13.69</td>
</tr>
<tr>
<td>128</td>
<td>46.46</td>
<td>34.30</td>
<td>28.93</td>
<td>22.13</td>
<td>17.81</td>
<td>13.97</td>
<td>11.69</td>
</tr>
<tr>
<td>192</td>
<td>45.78</td>
<td>33.92</td>
<td>28.34</td>
<td>21.16</td>
<td>17.41</td>
<td>13.63</td>
<td>11.69</td>
</tr>
</tbody>
</table>

that we are benefitting from mapping the data to a higher dimensional space non-linearly. Would this explain the observation that the primary visual cortex constitutes an about 100 times overcomplete representation of the sensory inputs? Will the performance stop improving or even start to degenerate as we keep increasing the dimensionality of $s$? We leave these questions for future work.

3.3.6 When It Will Fail

As discussed at the beginning of this paper, the whole idea of the universal visual feature hypothesis is based on the observation that visually quite different images share similar local statistical structures. Hence, the method will greatly fail when the local statistics of the images under consideration are very different from the images on which the visual features are learned. This may occur when some artificial statistics are introduced during the image capture process. This is exactly why we always make the pixel values of each image to have zero mean and unit variance.

Here we show how the method will seriously fail. We download the processed $32 \times 32$ Yale face images from (Cai et al., 2007)’s first author’s homepage. And then apply the same procedure as described in Section 3.3.2. That is, everything is the same except that we are directly using the $32 \times 32$ version images, instead of using the $64 \times 64$ version followed by a downsampling operation. As shown in Table 3.5, the performance degenerates greatly.

It turns out that the software the authors used to down-sample the images introduced pixellation effects, as shown in Figure 3.3.6. As the original images
Figure 3.6: The pixellation effect. The left figure is the original $32 \times 32$ version. The right figure is generated by down-scaling the $64 \times 64$ image to $32 \times 32$ using the imresize matlab function.

on which we have learned the visual features contains are smooth outdoor scenes, the pixellation errors make the images being tested to have quite different local statistical structure. This effect, however, can be easily counteracted by simply down-scaling or blurring the images slightly. This is also why we down scaled the handwritten digits from $28 \times 28$ to $18 \times 18$ size.

This, however, raises an interesting question – how to avoid/remove artificial statistics introduced during image capture? For the human visual system, there is a unique set of image capturing devices. Any artificial statistics introduced by these devices will be applied to future testing images, and should not cause much trouble. However, for a computer vision system in which the images are captured using different devices, artificial statistics introduced during image capture might be a serious problem. Although in our experiments we found that simply shrinking/blurring the images will suffice to yield good performance, how to detect and remove artificial statistical structures is still an important research topic in the study of these universal visual features.

3.4 Discussion

In this work, we have shown how to learn a set of visual features from randomly collected images, and apply these features on different datasets without special adjustment of the features. These ideas are inspired by the structure of human visual system, and based on the observation that visually different im-
ages share similar local statistical structures. We show that these features help to yield recognition performance comparable to state-of-the-art computer vision techniques.

There is a great deal of recent work using deep networks for recognition (Hinton and Salakhutdinov, 2006; Serre et al., 2005). The work we report here suggests that shallow but wide networks may be sufficient for recognition. This observation is consistent with another recent article using shallow networks with a nonlinear activation function for object recognition (Pinto et al., 2008).

However, as we have shown in Section 3.3.6, we need to apply this technique with caution. Although we showed in the paper that simply shrinking the images being tested will help to remove some artificial statistics, it remains an interesting question how to automatically detect and remove such structures.

We feel that the best way to interpret the linear efficient encoding hypothesis as instantiated by ICA is that it is trying to provide a set of universal visual features. This is plausible because globally different visual stimuli in fact share similar local statistical structure. This is beneficial because all we need to train the features is a set of natural images. These images do not need to be labeled, and apparently, they do not need to share similar properties, as evidenced by the application of our approach to hand written digits.

Acknowledgments

Chapter 3, in full, is a reprint of the paper published in CVPR 2008 “Looking around the backyard helps to recognize faces and digits”, co-authored with Garrison W. Cottrell. The dissertation author is the primary investigator and author of the paper.
Chapter 4

Sparse PCA: A Retinal Coding Model

4.1 Introduction

Sensory information goes through various forms of processing before it reaches the cerebral cortex. For example, visual information is transformed into neural signals at the retina, where it passes through the retinal ganglion cells that are characterized by their center-surround shaped receptive fields (Enroth-Cugell and Robson, 1966); auditory information, on the other hand, is passed to the brain through the auditory nerve fibers whose filtering properties can be well described by gammatone filters (Kiang et al., 1965). Since such peripheral processing prepares the data which the subsequent cortical processing relies on, its functional roles have attracted a great deal of attention in the past several decades (Srinivasan et al., 1982; Field, 1987; Atick and Redlich, 1992; Lewicki, 2002; Vincent et al., 2005; Graham et al., 2006; Doi and Lewicki, 2007).

Nevertheless, there are still mysteries concerning the functional roles of the pre-cortical processing. For example, do different sensory modalities (visual, auditory, olfactory, etc.) adopt the same computational principles in their pre-cortical stages? Although it is tantalizing to assume so, recent studies suggest otherwise. For example, Lewicki (2002) learned the gammatone filters from natural
sound using independent component analysis (ICA), which was previously applied to natural image patches to learn the edge/bar shaped filters resembling the V1 simple cells’ receptive fields (Olshausen and Field, 1996; Lewicki and Sejnowski, 2000). Since gammatones model pre-cortical auditory nerve fibers while V1 is a region of cortex, this gives rise to a puzzle: Why would the brain use the same strategy for preprocessing at a pre-cortical stage in the auditory pathway and early cortical processing in the visual pathway (Olshausen and O’Connor, 2002)?

In fact, questions remain even for peripheral processing of a single modality. Retinal coding, for example, has been hypothesized to “whiten” the visual inputs (Srinivasan et al., 1982) by producing a flattened response spectrum for natural visual inputs from a specific range of spatial frequencies. This whitening theory links the properties of retinal coding with the statistics of natural scenes and is now part of the prevailing view of retinal processing. However, as discussed by Graham et al. (2006), there are at least two different motivations behind the whitening theory: response equalization and output decorrelation. The response equalization theory hypothesizes that retinal coding seeks a representation that ensures that each neuron has approximately the same average activity level when the animal is presented with natural scenes (Field, 1987; Brady and Field, 1995; Field and Brady, 1997; Brady and Field, 2000). The output decorrelation theory, on the other hand, follows the efficient coding principle (Atick and Redlich, 1992). It hypothesizes that retinal coding serves to capture the second-order statistical structure of the visual inputs by making the signals from these neurons uncorrelated. However, these two theories alone are not sufficient to predict the center-surround receptive field design one actually observes (Graham et al., 2006). This leaves open questions about the pre-processing’s true purpose and why such fields would emerge.

Given the above considerations, it is desirable to build a retinal coding model that integrates the different ideas behind the whitening theory and explains the origins of the observed center-surround receptive fields. Ideally, this model should also able to explain the pre-cortical processing of other modalities.

In this work, we attempt to achieve this goal by revising and building upon a retinal coding model proposed by Vincent et al. (2005). We revise their model
by imposing a threshold on the model neurons’ maximum activity level, which consequently introduces response equalization into the model. This revised model turns out to be another variation of the widely used Sparse PCA algorithm (Zou et al., 2006; Mairal et al., 2010). We derive an efficient algorithm to learn the model parameters, and apply the model to grayscale images, chromatic images, grayscale videos, environmental sound, mammal vocalization, and human speech. As a result, this retinal coding model learns visual and auditory filters that qualitatively resemble the filtering properties of retinal ganglion cells and auditory nerve fibers. The learned filters have many desired properties, such as spatiotemporal filters trained with video that capture some of the differences between the magno-/parvo-pathways. The properties of response equalization and output decorrelation also fall out naturally from the revised model. These results are encouraging given that the model is only a linear (and hence a highly simplified) model of retinal processing.

4.2 A Retinal Coding Model

In this section, we analyze and revise the retinal coding model proposed by Vincent et al. (2005), then derive an efficient algorithm to learn its model parameters. We will also show that the response equalization and the output decorrelation will fall out from our revised model.

4.2.1 The Original Model

Vincent’s Model Images can be represented as high-dimensional real-valued data; if L neurons are representing the input image, the observed image can be represented as \( x \in \mathbb{R}^L \). Given input vectors \( x \in \mathbb{R}^L \), Vincent et al. (2005) seek to find the output vectors \( s \in \mathbb{R}^M \) (the signal from the retinal ganglion cells) and basis functions \( A \in \mathbb{R}^{L \times M} \) such that the following objective function is minimized:

\[
E = \left\langle \frac{\|x - As\|^2}{2} \right\rangle + \lambda \|A\|_1
\]  

(4.1)
where \(\langle \cdot \rangle\) denotes taking average over all the input samples. They also constrain each column of \(A\) to have unit L2 norm.

Vincent et al. interpret \(A\) as the synaptic strength between the input and output neurons; and hence they interpret the sparsity penalty on \(A\) as the desire to minimize the neural wiring cost. Such a connection that links \(A\) to some physical property of the neural system is likely. However, here we choose to interpret the objective function at the conceptual level: from a generative point of view, the columns of \(A\) are the elementary features that the model uses to “construct” the observed inputs. Thus the overall objective function can be understood as capturing most information of the inputs using an economic dictionary of elementary features.

Vincent et al. apply the above model to natural scene image patches. For each input \(x\), they find the optimal output \(s\) by applying gradient descent on the objective function with respect to \(s\). Once the optimal \(s\) is obtained, they update \(A\) by taking stochastic gradient descent of \(E\) with respect to \(A\). At the end of learning, they visualize the columns of \(A\) as the model neurons’ receptive fields, which they report to be center-surround shaped.

**The Model Neurons’ Receptive Fields** When \(A\) is fixed and full rank, it is obvious that the optimal output is given by:

\[
s^* = (A^t A)^{-1} A^t x = W x \quad \text{(let} \ W \text{ denote} \ (A^t A)^{-1} A^t) \quad (4.2)
\]

Hence if we apply the linear reverse correlation technique to the model neurons (Chichilnisky, 2001), which recovers the filters transforming the inputs to the outputs, we should get the rows of \(W\) (the pseudo-inverse of \(A\)) as the model neurons’ receptive fields.

The columns of \(A\) and the rows of \(W\) describe different aspects of the model. The columns of \(A\) describe the elementary *features* that the model uses to construct the observed inputs; this forms a kind of visual dictionary. The rows of \(W\), on the other hand, represent the best linear *filters* to recover the weights assigned to the elementary features when generating the observed inputs. Figure 4.1 displays
Figure 4.1: The elementary features (the columns of $A$) and the corresponding filters (the rows of $W$) learned from $16 \times 16$ grayscale image patches, using Vincent et al.’s retinal coding model (code courtesy of Ben Vincent).

the columns of $A$ as well as the corresponding rows of $W$ learned from grayscale image patches.

4.2.2 The Revised Model

Our Revision  Vincent et al. constrain each column of $A$ to have unit L2 norm. Here we propose to use the following constraint instead:

$$\langle s_i^2 \rangle \leq 1 \quad \text{for } i = 1, 2, \ldots, M$$

(4.3)

where $s_i$ denotes the $i$-th output neuron’s response, and $\langle \cdot \rangle$ again denotes taking average over all the samples. This constraint conveys two messages: (1) each neuron has a maximum average activity level threshold; (2) the same threshold is imposed to all the neurons.

We impose $\langle s_i^2 \rangle \leq 1$ because it allows us to include response equalization (i.e., $\langle s_i^2 \rangle = 1$ for all the output neurons), a classic theory of retinal coding that is supported by neurophysiological studies (see (Graham et al., 2006) for discussion). Upon convergence, the model should satisfy $\langle s_i^2 \rangle = 1$. Otherwise, we would get $\alpha = \langle s_i^2 \rangle < 1$. Then by dividing $s_i$ by $\sqrt{\alpha}$ and multiplying the corresponding column of $A$ with $\sqrt{\alpha}$, we can reduce the objective function while still satisfy the constraint. Hence, by imposing $\langle s_i^2 \rangle \leq 1$ we implicitly impose response equalization $\langle s_i^2 \rangle = 1$. 
Sparse PCA The above revision not only integrates the response equalization theory into the model, but it also allows us to derive an efficient algorithm to learn the model parameters, because the revised model turns out to be another variation of Sparse PCA (Zou et al., 2006). Eq (4.1) can be written as:

$$E = \frac{\|X - AS\|^2_F}{2n} + \lambda \|A\|_1$$  \hspace{1cm} (4.4)

where the columns of $X$ and $S$ store the input and output vectors; $\|\|_F$ denotes taking the Frobenius norm of a matrix (i.e., the square root of the sum of squared entries); $n$ denotes the number of samples. The constraint in Eq (4.3) means that each row of $S$ is constrained to have L2 norm less than or equal to $\sqrt{n}$.

As shown by Mairal et al. (2010), the Sparse Coding problem (Olshausen and Field, 1996) can be expressed in the matrix factorization form as:

$$E = \frac{\|X - AS\|^2_F}{2} + \lambda \|S\|_1$$  \hspace{1cm} (4.5)

Each column of $A$ is constrained to have L2 norm less than or equal to $q$. Sparse PCA can then utilize Sparse Coding algorithms because the objective function in Eq (4.4) can be re-written as:

$$E = \frac{\|X^t - S^tA^t\|^2_F}{2n} + \lambda \|A^t\|_1$$  \hspace{1cm} (4.6)

Now we see that Eq (4.6) can be symbolically mapped to Eq (4.5), if we replace $X^t$ with $X$, $S^t$ with $A$, $A^t$ with $S$, with extra care taken to deal with $n$ and $\sqrt{n}$.

The above discussion suggests another interpretation of the retinal coding model: it can be interpreted as removing redundancy between input samples, because Sparse Coding is usually interpreted as removing redundancy between input dimensions (Lewicki and Olshausen, 1999; Lewicki and Sejnowski, 2000). In this sense, Architecture-1 ICA (Bartlett et al., 1998), which applies ICA to $X^t$ instead of $X$, was perhaps the first Sparse PCA algorithm ever proposed.

We then only need to pick up some efficient Sparse Coding algorithms to optimize our model parameters. Typically, optimizing the objective function in Eq (4.5) is factored into two sub-problems: optimizing $A$ while fixing $S$, and optimizing $S$ while fixing $A$. Both sub-problems are convex optimization problems.
(that’s one reason we impose \( \langle s_i^2 \rangle \leq 1 \) instead of \( \langle s_i^2 \rangle = 1 \); otherwise optimizing \( S \) is no longer a convex optimization problem). Recently, it was shown that the coordinate descent algorithm is considerably faster than competing methods for both sub-problems (Friedman et al., 2010; Mairal et al., 2010). These algorithms are described in Appendix A.1.

### 4.2.3 Approximate Optimization

The above algorithm has a computational complexity that depends on the number of samples \( n \). Here we give an approximate algorithm whose complexity only relies on \( L \) and \( M \), the input and output dimensionalities. In our experiments on grayscale images, this reduces the computation time from 37 minutes to 10 seconds, with the learned parameters very close to those learned without approximation. The derivation of the algorithm also helps us to see the connection between this retinal coding model and the output decorrelation theory (Atick and Redlich, 1992).

Our algorithm utilizes two approximations, both of which rely on the condition that \( \lambda \) is small and \( L > M \). Under such a condition, the first approximation we use is to calculate the optimal output \( s \) using (See Appendix A.2 for detailed derivations):

\[
\mathbf{s}^* \approx \mathbf{Wx} \quad (\mathbf{W} = (\mathbf{A}^t \mathbf{A})^{-1} \mathbf{A}^t)
\]  

(4.7)

Replacing the above approximation into the original objective function, we get

\[
E \approx \frac{\text{Tr} ((\mathbf{I} - \mathbf{A}\mathbf{W})\mathbf{C}(\mathbf{I} - \mathbf{A}\mathbf{W})^t)}{2} + \lambda \|\mathbf{A}\|_1
\]  

(4.8)

where \( \mathbf{C} = \langle \mathbf{x}\mathbf{x}^t \rangle \), \( \mathbf{I} \) denotes the identity matrix, and \( \text{Tr} \) denotes taking the trace of a matrix. The constraint \( \langle s_i^2 \rangle \leq 1 \) can be expressed as \( \text{diag}(\mathbf{WCW}^t) \leq 1 \).

Since \( \mathbf{C} = \langle \mathbf{x}\mathbf{x}^t \rangle \) is positive semidefinite, we can factor it using the eigenvalue decomposition \( \mathbf{C} = \mathbf{UVU}^t \), where \( \mathbf{U} \) is a unitary matrix (i.e., \( \mathbf{UU}^t = \mathbf{I} \)) containing the eigenvectors as its columns; \( \mathbf{V} \) is a diagonal matrix with the eigenvalues on its diagonal. Let \( \mathbf{B} = \mathbf{UV}^{1/2} \), we get \( \mathbf{C} = \mathbf{BB}^t \). Replacing \( \mathbf{C} = \mathbf{BB}^t \)
into Eq (4.8), we get

\[ E \approx \frac{\|B^t - Z^tA^t\|_F^2}{2} + \lambda \|A^t\|_1 \]  

(4.9)

where \( Z = WB \). The second approximation we use is to relax \( Z \) to a free variable instead of constraining it to \( Z = WB \). The constraint becomes that each column of \( Z^t \) should have L2 norm less than or equal to 1.

Now we see that Eq (4.9) can also be symbolically mapped to Eq (4.5), if we replace \( B^t \) with \( X \), \( Z^t \) with \( A \), and \( A^t \) with \( S \). Hence the objective function in Eq (4.9) can also be optimized by efficient Sparse Coding algorithms, such as the coordinate descent algorithms in Appendix A.1. Since Sparse Coding reduces the redundancy between inputs, the above derivation also brings up another interpretation of the retinal coding model: it can be seen as removing the redundancy between the eigenvectors of \( C \) when \( \lambda \) is small and \( L > M \).

### 4.2.4 Relationship With Previous Work

The revised retinal coding model integrates three theories: response equalization (Field, 1987), output decorrelation (Atick and Redlich, 1992), and the economy of elementary features (Vincent et al., 2005). The response equalization is expressed in the constraint term. The economy of elementary features is imposed through the sparsity penalty term. When \( \lambda \) is small and \( L > M \), the reconstruction error reduces to a term that only involves \( C = \langle xx^t \rangle \), which concurs with the output decorrelation theory.

As shown in Appendix A.2, the three terms in the objective functions determines different aspects of the basis functions (i.e., the columns of \( A \)): the reconstruction error determines the subspace that the basis functions span; the constraint specifies the lengths of the basis functions; and the sparsity penalty rotates the basis functions within the subspace determined by the reconstruction error. In this sense, all three terms, and hence all three theories that they embody, are necessary to fully characterize the retinal coding model. This observation agrees with the prediction by Graham et al. (2006): “we conclude that a minimum of three constraints must be considered to account for the known linear properties
– decorrelation, response equalization, and size/sparseness.”

Some previous models also integrate the above three theories. For example, Doi and Lewicki (2007) proposed a model in which the retinal output is a linear transform of the input \( \mathbf{s} = \mathbf{Wx} \) if we ignore the optical blur. The functional role of their model is to minimize the difference between the input \( \mathbf{x} \) and its reconstruction \( \mathbf{AWx} \). Hence their model can be seen as the approximation we take when \( \lambda \) is small and \( L > M \). They also include two extra terms during learning, to regularize the average amplitude of the outputs, and to impose sparseness on \( \mathbf{W} \) (instead of \( \mathbf{A} \)). However, their formulation does not have the convexity properties as in our model, and they didn’t explore how the receptive fields should change with different levels of sparseness on \( \mathbf{W} \).

4.3 Experiments

We apply our algorithm to grayscale images, chromatic images, grayscale videos, environmental sounds, mammal vocalization, and human speech. Below we display the learned elementary features \( \mathbf{A} \) and the corresponding filters \( \mathbf{W} \), and compare the filters to the filtering properties of retinal ganglion cells and auditory nerve fibers.

4.3.1 Grayscale Images

**Learning Method** We applied the model to four image datasets: van Hateren natural images (van Hateren and van der Schaaf, 1998), Kyoto natural images (Doi et al., 2003), Berkeley segmentation dataset (Martin et al., 2001), and Caltech-256 object category dataset (Griffin et al., 2007). We didn’t observe qualitative difference between the features learned from these datasets. The features reported below are obtained using van Hateren natural images. We use a subset of it selected by Karklin and Lewicki (2009), which contains 110 1536 \( \times \) 1024 grayscale images.

For each image, we discard two pixels off the image borders, normalize the pixel values to \([0, 1]\), then apply a nonlinear function that simulates the cone
Figure 4.2: One grayscale image from van Hateren natural image dataset, before and after applying the cone nonlinearity in Eq (4.10).

\[ x = 1 - \exp(-k \cdot x) \]  

where \( x \) denotes the pixel value, and \( k \) is selected for each image such that its average pixel value equals 0.5 after the nonlinearity. This nonlinearity does not seem to alter the features learned by the retinal coding model, but might help to expose higher-order statistical structure (Karklin and Lewicki, 2009). Figure 4.2(a) and 4.2(b) display one image before and after applying the nonlinearity. Then we randomly sample 1000 20 × 20 image patches from each image.

Our algorithm efficiently minimizes the objective function. We test two optimization methods: directly optimize the objective function in Eq (4.6), or first initialize \( A \) by optimizing Eq (4.9) and then optimize Eq (4.6). We implement the experiments in single precision on a computer server with Intel Core i7 processors. When we reduce the dimensionality from 400 to 100 and set \( \lambda = 0.004 \), it takes 37 minutes to directly optimize Eq (4.6). On the other hand, it takes less than 10 seconds to initialize \( A \) by optimizing Eq (4.9) and another 40 seconds to optimize Eq (4.6). As shown in Figure 4.3(a), our approximate method efficiently minimizes the objective function. In fact, after \( A \) is initialized with the approximate method, further optimizing the objective function with the direct method only causes a less than 0.1% change of the objective function, on average the weights in \( A \) are
only changed by 0.4%, and no visible difference in the features and filters can be observed. The closeness of the approximation means that in later experiments (color images, video, and sound) the approximation provided by Eq (4.9) is used without additional optimization.

Our model captures most of the variance of the inputs while having sparse weights in $\mathbf{A}$. To check how well our model preserves information, we calculate the eigenvalues of the covariance matrix of the reconstructed data and compare them to the eigenvalues from the original data. As shown in Figure 4.3(b), our model captures 99.23% of the variance that could be captured by a linear model with 100 output neurons. On the other hand, this amount of variance is captured by a surprisingly small number of connections. Figure 4.3(c) and 4.3(d) plot the distribution of the connection weights in $\mathbf{A}$, learned by our model versus by the standard PCA. It turns out that 96.31% of the connection weights in our model are absolute zero; in contrast, none of the connection weights from the standard PCA is zero. If $\mathbf{A}$ characterizes the connection strength between the input and output neurons, this would suggest that each output neuron is connected to only about 3.69% of the input neurons, which could be advantageous for a biological system.

**Features and Filters** Experiment result confirms our previous analysis that we can use $\mathbf{Wx}$ to approximate the optimal output, and that $\mathbf{W}$ describes the model neurons’ receptive fields. As shown in Figure 4.4(a), $\mathbf{Wx}$ closely approximates the optimal output. One difference, however, is that the optimal $\mathbf{s}$ tends to have a smaller amplitude than $\mathbf{Wx}$. As shown in Figure 4.4(b), when we use the linear reverse correlation technique to estimate the model neurons’ receptive fields, the estimated receptive fields closely resemble the corresponding rows of $\mathbf{W}$. Hence, in the remaining text we will just compare the properties of $\mathbf{W}$ with the linear filtering properties of the retinal ganglion cells.

The learned elementary features (i.e., the columns of $\mathbf{A}$) are Gaussian blobs of similar sizes that evenly cover the $20 \times 20$ image patch. The top panel in Figure 4.5(a) displays 10 features randomly selected from all 100 features. We fit all the features with 2D Gaussian blobs, and plot them as circles in Figure 4.5(b).
Figure 4.3: Experiments on grayscale image patches. Figure 4.3(a) plots how the objective function changes during learning. The black dash-dotted line plots the objective function using the direct optimization method; the blue dashed line plots the objective function when we use the approximate method to find $A$; the red solid line plots the objective function when we further fine tune the parameters using the direct method after initializing $A$ with the approximate method. Figure 4.3(b) plots the eigenvalues of $C$ from the original data (the blue solid line) versus those from the reconstructed data (the red dashed line). Our model captures 99.23% of the variance that could be captured by an optimal linear model with 100 output neurons. Figure 4.3(c) and 4.3(d) plot the distribution of the connection weights in $A$, learned by our model versus by the standard PCA which normalizes each principal component to have unit variance. 96.31% of the connection weights in our model are absolute zero, whereas none of the connection weights by the standard PCA is zero.
Figure 4.4: \( \mathbf{W} \) approximates the model neurons’ receptive fields. In Figure 4.4(a), we compare \( \mathbf{Wx} \) with the optimal \( \mathbf{s} \) inferred using the direct method during the experiments on grayscale image patches. If \( \mathbf{s} = \mathbf{Wx} \) holds exactly, all the points should lie on the red solid line. Once we have learned the optimal \( \mathbf{A} \) on grayscale image patches, we feed the model with white noise inputs and infer their optimal outputs, then use \( \langle \mathbf{x}s^t \rangle \) to estimate the model neurons’ receptive fields. The top panel in Figure 4.4(b) displays three rows of \( \mathbf{W} \); the bottom panel displays the estimated receptive fields.
The center and the radius of each circle represent the center and twice the standard deviation of the fitted Gaussian blob. To visualize how well the Gaussian blobs fit with the features, we display the first feature in Figure 4.5(a) and highlight its fitted Gaussian blob. As shown in the figure, the Gaussian blobs provide a mosaic coverage of the image patch: the average standard deviation of the Gaussian blobs is 0.98, whereas the average distance between the centers of neighboring blobs is 2.07. In other words, the features can be roughly viewed as black/white dots of unit radius lying next to each other. This result can be intuitively interpreted as: We need bigger “pixels” to construct an image patch if we have to use fewer pixels. This intuition is further confirmed in Figure 4.5(d), in which we plot 32 features learned from 20 \times 20 image patches with \( \lambda = 0.01 \). It remains an interesting question how this mosaic coverage should look like on an irregular input lattice (Liu et al., 2009); we leave this for future work.

The optimal filters are center-surround shaped, as shown in the bottom panel of Figure 4.5(a). The first filter, for example, recovers the weight assigned to the first feature in the top panel. It is tantalizing to think that some of the filters are ON-centered while others are OFF-centered. However, we can switch the signs of the features (and hence the signs of the optimal filters) without changing the model’s objective function. Hence our model cannot explain phenomena related to the difference between the ON-centered and the OFF-centered cells (Chichilnisky and Kalmar, 2002).

It is interesting to see why a population of Gaussian blob shaped features should give rise to the center-surround shaped filters. Since \( W = (A^tA)^{-1}A^t \), the \( i \)-th filter (i.e., the \( i \)-th row of \( W \)) can be viewed as a weighted sum of all the elementary features, with the weights being the \( i \)-th row of \( (A^tA)^{-1} \). We visualize these weights in Figure 4.5(c) for the first filter in Figure 4.5(a). The elementary features are plotted as circles, as in Figure 4.5(b) and 4.5(d). The color of each circle represents the weight assigned to each feature. The weights are normalized to include the signs and the amplitude of the fitted Gaussian blobs. We set the background to zero to better visualize the weights.

As shown in Figure 4.5(c), the optimal filter can be seen as a sequence of
efforts to recover the contribution of the first feature in Figure 4.5(a) when the image patch is constructed. First, we apply the first feature as a template filter on the image patch, to estimate its contribution. This estimation is inaccurate because this feature overlaps with other features. To get a more accurate estimation, we subtract the contribution from its neighboring features. To get an even more accurate estimation, we add back the contribution from the features neighboring the features that surround the first feature. This process repeats until it reaches all the features. The weight reduces quickly for features removed from the first feature, which makes the resulting filter localized and keeps the filter center-surround shaped (for low lambda, some additional ripples can be observed - however, these additional ripples have also been observed in ganglion receptive fields (Jr. and Granda, 2002)).

**Noise Images** Are the above features truly learned from natural scene images? If so, which aspects of natural scene images give rise to the learned features? To answer these questions, we apply our algorithm to white noise images, which contain no statistical structure, and pink noise images, which follow the $1/f$ power law as natural scene images (Field, 1987) but otherwise contain no structure. Figure 4.6(a) and 4.6(b) display two example images.

We apply our algorithm to $20 \times 20$ image patches extracted from the noise images, reduce the dimensionality from 400 to 64, and set $\lambda = 0.004$. Figure 4.6(c) displays some of the learned filters. On white noise images, the learned features are one-pixel image templates; the corresponding filters also only contain one non-zero pixel. That is, the model simply keeps 64 pixels and ignores the other pixels. That's the best it can do with 64 features, because white noise images contain no structure. On pink noise images, we learn the same elementary features (and hence the same filters) as those learned from natural scene images. This result supports the hypothesis that the center-surround shaped filters come from the $1/f$ power spectrum of natural scene images, which agrees with the classic whitening theory (Srinivasan et al., 1982; Field, 1987; Atick and Redlich, 1992).
Figure 4.5: Elementary features and their corresponding optimal filters, learned from 20 × 20 grayscale image patches. In Figure 4.5(a), the top panel displays 10 features randomly selected from all the 100 features; the bottom panel displays their optimal filters. We fit the features with Gaussian blobs and plot them as circles in Figure 4.5(b). The center and the radius of each circle represent the center and twice the standard deviation of the fitted Gaussian blob. To help visualize how well the Gaussian blobs fit with the features, we display the first feature in Figure 4.5(a) and highlight its fitted Gaussian blob. These Gaussian blobs will become bigger if we use a smaller number of features to “construct” the image patches, as shown in Figure 4.5(d). Figure 4.5(c) displays the first filter in Figure 4.5(a) as a weighted sum of all the 100 features. Each feature is plotted as a circle, as in Figure 4.5(b) and 4.5(d). The color of each circle represents the weight assigned to this feature (read the main text for details).
Figure 4.6: Experiments on white/pink noise images. Figure 4.6(a) and 4.6(b) display images containing white noise and pink noise. The top panel in Figure 4.6(c) displays 10 filters learned from white noise images; the bottom panel plays the filters learned from pink noise images.

**Sparseness Level**  The sparseness level $\lambda$ plays an important role in shaping the learned features and filters. When $\lambda$ gets bigger, the model puts more emphasis on getting sparse connections instead of keeping more information about the inputs. Such a situation could happen to a biological system, e.g., when the system is malnourished. Here we check how the learned filters should change with a increased $\lambda$ value.

By analyzing the filters in Fourier space, we can plot their sensitivity at various frequencies, producing a contrast sensitivity curve. As shown in Figure 4.7, with bigger $\lambda$ value, the model becomes less sensitive to low frequency information, but slightly more sensitive to high frequency information. This change matches with psychophysical studies of contrast sensitivity for children with early malnutrition. Compared with normal children, malnourished children are reported to be less sensitive to low visual frequencies, but slightly more sensitive to high frequencies (dos Santos and Alencar, 2010). This shift of acuity towards high frequencies, as suggested by our result, might due to the effort of the neural system to capture more visual information with a limited neural wiring budget.
Figure 4.7: Experiments with increased sparseness level $\lambda$. We plot the amplitudes of different frequency components of the filters learned with $\lambda = 0.004$ and $\lambda = 0.02$. As shown in the figure, with an increased $\lambda$ value, the filter becomes less sensitive to low frequencies, but slightly more sensitive to high frequencies.

4.3.2 Chromatic Images

Learning Method  We applied our algorithm to four chromatic image datasets: the Kyoto natural image dataset (Doi et al., 2003), the McGill color image dataset (Olmos and Kingdom, 2004), the Berkeley segmentation dataset (Martin et al., 2001), and the Caltech-256 object category dataset (Griffin et al., 2007). From these datasets we learn qualitatively similar features. Below we report the features learned from Kyoto image dataset, which contains 62 1000 × 1280 pixel chromatic images of natural scenes.

First, we estimate the retinal L, M, S cones’ responses to those images. The original images are stored in sRGB color representation (Stokes et al., 1996). We normalize the pixel values of each image to [0, 1], then transform the image to the CIE XYZ color space (Smith and Guild, 1931), from which we estimate the LMS cone responses following the CIECAM02 color appearance model (Fairchild, 2001). In this manner, we can roughly estimate how the retinal L, M, and S cones would respond when presented with the image content.

After that, we apply the cone nonlinearity in Eq (4.10) to the estimated
Figure 4.8: One image from the Kyoto chromatic image dataset, before and after applying the cone nonlinearity. The original image is stored in sRGB color representation. We transform it to the LMS cone representation, then apply the nonlinearity in Eq (4.10). To visualize the image after the cone nonlinearity, we transform the image back to sRGB color representation.

LMS cone responses. Figure 4.8(a) and 4.8(b) display one image before and after applying the nonlinearity. Then we extract all 20 by 20 image patches, and estimate the matrix $C$. We apply the retinal coding model using the approximation in Eq (4.9) to reduce the dimensionality from 1200 to 256 with $\lambda = 0.002$. The resulting model captures 99.75% of the variance that could possibly be captured by an optimal linear model with 256 output neurons, with 96.11% of its connections being absolute zero.

Features and Filters Figure 4.9(a) displays 6 representative features as well as their corresponding filters, learned from chromatic image patches. We visualize the connection strength from three of the filters to the L/M/S channels in Figure 4.9(b). Among all the 256 learned features, 193 are black/white blobs, 48 are blue/yellow blobs, 15 are red/green blobs. Figure 4.9(c), 4.9(d), and 4.9(e) plot the spatial layout of learned features.

The above result replicates the segregation of the spatial channel and the color channel at the retina stage (Calkins and Sterling, 1999). This segregation was explored in previous research that applied information-theoretic methods to natu-
ral color spectra/images, such as PCA (Buchsbaum and Gottschalk, 1983; Derrico and Buchsbaum, 1991) and ICA (Tailor et al., 2000; Wachtler et al., 2001; Lee et al., 2002; Doi et al., 2003; Caywood et al., 2004). One common observation in these studies is that the learned visual features (eigenvectors or independent components) segregate into black/white, blue/yellow, and red/green opponent structures, either shaped as Fourier basis functions or Gabor kernel functions (Wachtler et al., 2001). Our work is the first, to the best of our knowledge, to learn the black/white, blue/yellow, and red/green center-surround shaped opponency structures, which match with the linear filtering properties of retinal ganglion cells.

4.3.3 Grayscale Videos

Learning Method To explore the spatio-temporal structure of natural videos, we collected a video dataset of 27 clips from YouTube (see appendix for their URLs). All the videos are from natural history shows from the BBC World Wide channel (www.youtube.com/user/BBCWorldwide). In order to have a realistic sample of natural videos, we eliminate those that are obviously unnatural, such as walking dinosaurs, cavemen, or pigeon-mounted cameras. We calculated the power spectrum of the videos in order to eliminate interlaced-format videos, which have an ellipse-shaped power spectrum elongated in the horizontal direction. 1

The original videos are in color. We transform them to grayscale videos using the method described by the Matlab function rgb2gray, normalize the pixel values between [0, 1], and apply the cone nonlinearity in Eq 4.10. Then we estimate the correlation matrix $C = xx^T$ for all the $12 \times 12 \times 12$ video cubes (with the local mean removed - keeping in the local mean did not substantially change the qualitative results, but it did seem to produce slightly noisier filters), and apply our algorithm to $C$ using the approximate method.

1We initially applied our algorithm to van Hateren video dataset (van Hateren and Ruderman, 1998). The learned features and filters are qualitatively similar to those reported below, except that some of the learned features are ellipse or even bar shaped and are elongated along the horizontal direction. These features most likely result from the fact that the original videos are interlaced (i.e., recording only the odd numbered lines in one frame, and the even numbered lines in the next frame). Although the videos were de-interlaced by block averaging with $2 \times 2 \times 2$ blocks (van Hateren, personal communication), their spatial power spectrum is still ellipse-shaped with more energy along the horizontal direction.
(a) Elementary features and filters  (b) Connection from the filters to the LMS cones

(c) Red/green  (d) Blue/yellow  (e) Black/white

Figure 4.9: Elementary features and the corresponding filters learned from $20 \times 20$ chromatic image patches with $\lambda = 0.001$. In Figure 4.9(a), the left panel displays 6 representative features; the right panel displays their corresponding filters. The features belong to three categories: black/white blobs, blue/yellow blobs, and red/green blobs. The corresponding optimal filters are center-surround shaped, with black/white, blue/yellow, or red/green antagonism. Figure 4.9(b) plots the connection strength from the filters to the L, M, S cones. Figure 4.9(c), 4.9(d), and 4.9(e) plot the spatial layout of learned features, as we did in Figure 4.5(b).
Features and Filters The learned features are black/white blobs whose contrast changes over time. As shown in Figure 4.10, the features can be well fitted to spatio-temporal Gaussians. The corresponding filters are spatially center-surround shaped. Their temporal profile seems to provide an “edge” detector along the temporal axis, which is similar to the temporal profile describing retinal ganglion cells (Pillow et al., 2008). To see the animation file of the learned filters, see www.cs.ucsd.edu/~hshan/spca/video_spca_w.gif.

Another interesting observation is that all the features segregate into two groups: those with low-spatial and high-temporal frequencies, and those with high-spatial and low-temporal frequencies, as plotted in Figure 4.11(a). This suggests that the division of ganglion cells into the magno-pathway and the parvo-pathway represents an efficient encoding of the visual environment.

This segregation appears to reveal statistical properties of natural videos, instead of coming from our specific algorithm. In fact, this segregation exists even for features learned with standard PCA. For features learned with standard PCA, it is generally true that eigenvectors with small eigenvalues correspond to features with high-spatial frequencies; however, if we check eigenvectors with similar eigenvalues, it is typical that they segregate into either high-spatial low-temporal frequencies, or low-spatial high-temporal frequencies. For example, Figure 4.11(b) plots the eigenvectors with the 30-th and 31-th biggest eigenvalues (0.0883 and 0.0878). To see the animation file of the top 400 eigenvectors, see www.cs.ucsd.edu/~hshan/spca/video_pca.gif.
Figure 4.11: Video features segregate into two groups: those with low-spatial and high-temporal frequencies, and those with high-spatial and low-temporal frequencies.
4.3.4 Sound

Learning Method We applied our algorithm to three sound datasets: the Pittsburgh natural sounds dataset (Smith and Lewicki, 2006), the TIMIT speech dataset (Lamel et al., 1986), and rainforest mammal vocalization dataset (Emmons et al.). The Pittsburgh natural sounds dataset contains 48 recordings of natural sound recorded around the Pittsburgh region, including ambient sounds (such as rain, wind, and streams) and quick acoustic events (such as snapping twigs, breaking wood, and rock impacts). The TIMIT speech dataset contains English speech from 630 speakers, with each person speaking 10 sentences. The rainforest mammal vocalization dataset contains the characteristic sounds of 109 species of rainforest mammals, such as primates, anteaters, bats, jaguars, and manatees.

We get qualitatively similar results on these three datasets. For each dataset, each recording is re-sampled at 16 kHz. We normalize the maximum amplitude of each recording to 1, take all the segments formed using a sliding window of 128 sample points (about a 8 millisecond window), then estimate the matrix $C$ for the sound segments. We then apply the approximate method to learn the features and filters.

Features and Filters As shown in Figure 4.12, the learned filters can be well fitted to gammatone filters. Gammatone filters resemble the filtering properties of auditory nerve fibers estimated using the reverse correlation technique from animal such as cats (Carney, 1990) and chinchillas (Recio-Spinoso et al., 2005).

Note that this is (to the best of our knowledge) the first time a non-ICA algorithm has learned gammatone-like filters from sound. Also, since we used the same algorithm for both visual and auditory modalities, this provides an answer to the question posed by Olshausen and O’Connor (2002): “Perhaps an even deeper question is why ICA accounts for neural response properties at the very earliest stage of analysis in the auditory system, whereas in the visual system ICA accounts for the response properties of cortical neurons, which are many synapses removed from photoreceptors.” Our model suggests that it is not necessary to use ICA to obtain gammatone filters from sound; rather, Sparse PCA can account for the
Figure 4.12: Figure 4.12(a) and 4.12(b) plot the revcor filters estimated from cat’s auditory nerve fibers using the linear reverse correlation technique, as well as the fitted gammatone filter. Figure 4.12(c) and 4.12(d) plot filters learned from the TIMIT speech dataset and the Pittsburgh environmental sound dataset respectively. The blue line plots the estimated filter; the red line plots the fitted gammatone filter.
receptive fields of neurons at the very earliest stages of analysis in both auditory and visual modalities.

4.4 Discussion

In this work, we devised a model based on the retinal coding model proposed by Vincent et al. The revised model integrates the response equalization theory, the output decorrelation theory, and the sparseness theory, which were predicted to be the necessary components of a retinal coding model (Graham et al., 2006). This model differs from Vincent’s model in that it is consistent with the response equalization theory. The revised has an interesting link to Olshausen’s sparse coding model; both models try to minimize the reconstruction error with response equalization, but our model imposes sparseness on the dictionary, while the sparse coding model imposes sparseness on the output.

We derived an efficient algorithm to learn the model parameters by transforming it into a sparse coding problem. Our approximate algorithm uses only the covariance matrix $C$, and runs orders of magnitude faster than the exact algorithm, while obtaining results that are less than one percent different in the objective function and learned basis functions. Our approximation works well under the assumption that $\lambda$ is small and $L > M$. While fast approximation is not a necessary part of a successful model, the speed of computation allows it to be used on more rich data, such as video.

We applied our model to grayscale images, color images, grayscale videos, human speech, and environmental sound, and learned visual and auditory filters that resemble the filtering properties of retinal ganglion cells and auditory nerve fibers. Some of the learned filters are novel. For example, it learns the right center-surround shaped color filters from color images; it learns the magno and parvo segregation pathways; and it learns the gammatone filters from natural sound.

Finally, as noted above, our model suggests an answer the question of why ICA gives features corresponding to cortical receptive fields but seems necessary to obtain gammatone-like filters for sound, which is a pre-cortical level of processing.
Our suggestion is that ICA is not necessary to obtain gammatone filters; rather, a PCA algorithm with sparsity and response equalization constraints can result in gammatone filters for sound, while also producing receptive fields similar to peripheral neurons in vision.

Acknowledgments

Chapter 4, in full, is a reprint of the paper in preparation “Notes on a retinal coding model”, co-authored with Matthew H. Tong and Garrison W. Cottrell. The dissertation author is the primary investigator and author of the paper.
Chapter 5

Summary

In this dissertation, we proposed a hierarchical model, Recursive ICA, that can capture nonlinear features of the visual inputs not captured by a single layer of ICA. The model is constructed by stacking another layer of ICA on top of the first layer ICA outputs. We derived a coordinate-wise nonlinear activation function to allow the second layer of ICA more efficiently remove the redundancy not removed by the previous layer. We showed that the visual features learned on the first layer, including the nonlinear activation function, help for pattern recognition tasks, such as face recognition and handwritten digit recognition.

In the second half of the dissertation, we analyzed and revised a retinal coding model previously proposed by Vincent et al. (2005). The revised model turned out to be another efficient coding model; however, instead of removing the redundancy among input dimensions, it removes the redundancy among the sample observations. When we apply the model to grayscale images, color images, grayscale videos, environmental sound, and human speech, it learns visual and auditory filters that exhibit the filtering properties of retinal ganglion cells and auditory nerve fibers.

One possible future work is to combine the above two works to build a hierarchical model that contains an alternative sequence of Sparse PCA and Sparse Coding layers. Sparse PCA reduces the dimensionality; and because of the sparseness penalty on the connections, Sparse PCA has to pool together inputs with similar properties. Sparse Coding expands the dimensionality; and because of the
sparseness on the response, each output neuron has to be tuned to specific features. Hence, Sparse PCA and Sparse Coding seem to well model the properties of the complex cells and the simple cells in the visual cortex (Hubel and Wiesel, 1962). Previous computational models that contain an alternative sequence of simple cells and complex cells, such as Neocognitron (Fukushima, 1980) and HMAX (Riesenhuber and Poggio, 1999), have been used to explain different aspects of the early visual processing layers. Hence it is interesting to see how well a hierarchical model that contains Sparse PCA and Sparse Coding layers could model the early visual processing layers.
Appendix A

A.1 Coordinate Descent Algorithms

Suppose we want to optimize an objective function $f(x)$, where $x$ is a high dimensional vector. To do so, the coordinate descent algorithm cyclically optimizes each dimension of $x$. Each time, it optimizes one dimension of $x$ while fixing the other dimensions. Once this dimension is optimized, the algorithm optimizes another dimension. This process repeats until the objective function can no longer be optimized.

Here we list the coordinate descent algorithm for optimizing the Sparse Coding objective function:

$$E = \left\langle \frac{1}{2} \|x - As\|_2^2 + \lambda \|s\|_1 \right\rangle$$  \hspace{1cm} (A.1)

Each column of $A$ is constrained to have L2 norm less than or equal to $q$.

Optimizing $s$ While Fixing $A$ \hspace{0.5cm} When we try to optimize the $i$-th coordinate of $s$, with $A$ and all the other coordinates of $s$ being fixed, the optimal $s_i$ is given by (Friedman et al., 2010):

$$y = a_i^t x - \sum_{j \neq i} a_i^t a_j s_j$$  \hspace{1cm} (A.2)

$$s_i^* = \begin{cases}  
(y - \lambda)/\|a_i\|_2^2, & \text{if } y > \lambda; \\
(y + \lambda)/\|a_i\|_2^2, & \text{if } y < -\lambda; \\
0, & \text{otherwise.}
\end{cases}$$  \hspace{1cm} (A.3)

where $a_i$ denotes the $i$-th column of $A$. 

69
Optimizing A While Fixing s When we try to optimize the $i$-th column of $A$, with $s$ and all the other columns of $A$ being fixed, the optimal $a_i$ is given by (Mairal et al., 2010):

$$u = \langle s_i x \rangle - \sum_{j \neq i} a_j \langle s_j s_i \rangle$$  \hspace{1cm} (A.4)

$$a_i^* = \frac{u}{\max(\langle s_i^2 \rangle, \|u\|_2/q)}$$  \hspace{1cm} (A.5)

A.2 Derivation of The Sparse PCA Approximate Algorithm

The revised retinal coding model aims to minimize the following objective function:

$$E = \left\langle \frac{\|x - As\|_2^2}{2} \right\rangle + \lambda \|A\|_1$$  \hspace{1cm} (A.6)

subject to the constraint that

$$\langle s_i^2 \rangle \leq 1 \quad \text{for every } i$$  \hspace{1cm} (A.7)

Below we show how to approximate the above objective function when $\lambda$ is small and $L > M$ (i.e., when we are reducing the data dimensionality).

The First Approximation Let’s first check the functional roles of the three terms in the objective function: the reconstruction error, the sparsity penalty, and the constraint.

If the objective function only contains the reconstruction error term, the model is reduced to the standard PCA problem (Jolliffe, 2002). The optimal basis functions (i.e., the columns of $A$) should span the subspace spanned by the eigenvectors of $\langle xx^t \rangle$ with top eigenvalues. The optimal outputs are given by $s^* = Wx$, where $W$ is the pseudo-inverse of $A$. Changing the basis functions’ individual directions and lengths within the subspace won’t change the reconstruction error, because for any full-rank matrix $G \in R^{M \times M}$ we have

$$As = (AG)(G^{-1}s)$$  \hspace{1cm} (A.8)
That is, for any \( A_{\text{new}} = AG \) which spans the same subspace as \( A \) but with different individual lengths and directions, we can find \( s_{\text{new}} = G^{-1}s \) such that the reconstruction error remains to be the minimum.

The objective function is not changed by adding the constraint \( \langle s_i^2 \rangle \leq 1 \). For any value of \( A \) and \( s \), we can always divide \( s_i \) (the \( i \)-th coordinate of \( s \)) and multiply \( a_i \) (the \( i \)-th column of \( A \)) with some value \( \alpha \) to satisfy the constraint without changing the reconstruction error. In other words, the constraint term only specifies the lengths of the basis functions.

The sparsity penalty \( \|A\|_1 \) will shrink the basis functions’ lengths and rotate their directions. However, when \( \lambda > 0 \) is sufficiently small such that the subspace that the basis functions span is mainly determined by the reconstruction error, the sparsity penalty will only serve to rotate the basis functions within the subspace determined by the reconstruction error.

Our first approximation utilizes the above analysis. When \( \lambda \) is small and \( L > M \), we use \( s^* \approx Wx \), where \( W \) is the pseudo-inverse of \( A \). Replacing it into the objective function in Eq (4.1), we get:

\[
E \approx \left\langle \frac{\|x - A(Wx)\|^2}{2} \right\rangle + \lambda\|A\|_1 \quad (A.9)
\]

\[
= \left\langle \frac{\text{Tr}((x - AWx)(x - AWx)^t)}{2} \right\rangle + \lambda\|A\|_1 \quad (A.10)
\]

\[
= \frac{\text{Tr}((I - AW)(x^t x^t)(I - AW)^t)}{2} + \lambda\|A\|_1 \quad (A.11)
\]

\[
= \frac{\text{Tr}((I - AW)C(I - AW)^t)}{2} + \lambda\|A\|_1 \quad \text{(let } C = \langle xx^t \rangle \text{)} \quad (A.12)
\]

where \( I \) denotes the identity matrix, \( \text{Tr} \) denotes taking the trace of a matrix. The constraint can now be approximated as:

\[
\langle s_i^2 \rangle \approx \langle (w_i x)(w_i x)^t \rangle = w_i \langle xx^t \rangle w_i^t \leq 1, \quad \text{or } \text{diag}(WCW^t) \leq 1 \quad (A.13)
\]

where \( w_i \) denotes the \( i \)-th row of \( W \), and \( 1 \) denotes a vector of 1’s. Hence, when \( \lambda \) is small and \( L > M \), the model mainly serves to capture the second-order statistical structure of the inputs.

**The Second Approximation** Since \( C = \langle xx^t \rangle \) is positive semidefinite, we can factor it using the eigenvalue decomposition \( C = UVU^t \), where \( U \) is a unitary
matrix (i.e., $UU^t = I$) containing the eigenvectors as its columns; $V$ is a diagonal matrix with the eigenvalues on its diagonal. Let $B = UV^{1/2}$, we get $C = BB^t$. Replacing this into the objective function in Eq (A.12), we get

$$E = \frac{\text{Tr}((I - AW)BB^t(I - AW)^t)}{2} + \lambda\|A\|_1$$  \hspace{1cm} (A.14)

$$E = \frac{\text{Tr}((B - AWB)(B - AWB)^t)}{2} + \lambda\|A\|_1$$  \hspace{1cm} (A.15)

$$E = \frac{\|B - AWB\|_F^2}{2} + \lambda\|A\|_1$$  \hspace{1cm} (A.16)

$$E = \frac{\|B - AZ\|_F^2}{2} + \lambda\|A\|_1 \quad \text{(let } Z = WB)$$  \hspace{1cm} (A.17)

The constraint becomes that each row of $Z$ should have L2 norm less than or equal to 1:

$$\text{diag}(WCW^t) = \text{diag}(WBB'W^t) = \text{diag}(ZZ^t) \leq 1$$  \hspace{1cm} (A.18)

Our second approximation is to relax $Z$ to a free variable instead of constraining it to $Z = WB$ because when $\lambda$ is small and $L > M$, this free variable will converge to $Z \approx WB$ following the same analysis in our first approximation. Now the objective function can be written as:

$$E = \frac{\|B^t - Z^tA^t\|_F^2}{2} + \lambda\|A^t\|_1$$  \hspace{1cm} (A.19)

Each column of $Z^t$ should have L2 norm less than or equal to 1. We see that this objective function can also be symbolically mapped to Eq (4.5). Hence its parameters can be optimized by efficient Sparse Coding algorithms.
A.3 URLs of Video Clips

http://www.youtube.com/watch?v=cMIRwCNvI94
http://www.youtube.com/watch?v=J7eRGHVx3p0
http://www.youtube.com/watch?v=8R1g0t00vGM
http://www.youtube.com/watch?v=K61FRGpvfwE
http://www.youtube.com/watch?v=M-nXN5SGmhw
http://www.youtube.com/watch?v=tOn2RhH36Mc
http://www.youtube.com/watch?v=gc9jFmkjizQ
http://www.youtube.com/watch?v=ZiW96Uci624
http://www.youtube.com/watch?v=1YQrLPW5DdY
http://www.youtube.com/watch?v=xKksJ3fvB1Q
http://www.youtube.com/watch?v=43id_NRajDo
http://www.youtube.com/watch?v=NRWehNVSA1A
http://www.youtube.com/watch?v=oJ-KzdRsQC4
http://www.youtube.com/watch?v=VuMRDZbrdXc
http://www.youtube.com/watch?v=2rlZVtKKWnk
http://www.youtube.com/watch?v=a1QB0NFcFog
http://www.youtube.com/watch?v=B71T_GpA2AM
http://www.youtube.com/watch?v=XB-8Grn6sRo
http://www.youtube.com/watch?v=JxrIWSNhNpko
http://www.youtube.com/watch?v=gVjqL-9Fh3E
http://www.youtube.com/watch?v=yKKabd3W904
http://www.youtube.com/watch?v=4ZFoqh8PQ88
http://www.youtube.com/watch?v=NR3Z4p5hspI
http://www.youtube.com/watch?v=RB9uzMjiYSQ
http://www.youtube.com/watch?v=a7XuXi3mqYM
http://www.youtube.com/watch?v=PBrStxu0Jbs
http://www.youtube.com/watch?v=u6ouWOGJk5E
Bibliography


Nuala Brady and David J. Field. What’s constant in contrast constancy? the
effects of scaling on the perceived contrast of bandpass patterns. *Vision Research*,

Nuala Brady and David J. Field. local contrast in natural images: normalisation

G. Buchsbaum and A. Gottschalk. Trichromacy, opponent colours coding and
optimum colour information transmission in the retina. *Proceedings of the Royal

Deng Cai, Xiaofei He, Yuxiao Hu, Jiawei Han, and Thomas Huang. Learning a
spatially smooth subspace for face recognition. *Proceedings of IEEE Conference

David J. Calkins and Peter Sterling. Evidence that circuits for spatial and color

Laurel H. Carney. Sensitivities of cells in anteroventral cochlear nucleus of cat to
spatiotemporal discharge patterns across primary afferents. *Journal of neuro-

Matthew S. Caywood, Benjamin Willmore, and David J. Tolhurst. Independent
components of color natural scenes resemble V1 neurons in their spatial and


E. J. Chichilnisky and Rachel S. Kalmar. Functional asymmetries in ON and OFF

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John

EMPATH: A neural network that categorizes facial expressions. *Journal of Cogni-

Joel B. Derrico and Gershon Buchsbaum. A computational model of spatiochro-
matic image coding in early vision. *Journal of Visual Communication and Image


