Dipartimento di Informatica e
Scienze dell'Informazione

A General Framework for Image Kernel Engineering

Annalisa Barla

Theses Series **DISI-TH-2005-XX**

DISI, Università di Genova
v. Dodecaneso 35, 16146 Genova, Italy http://www.disi.unige.it/

Università degli Studi di Genova

Dipartimento di Informatica e Scienze dell'Informazione

Dottorato di Ricerca in Informatica

Ph.D. Thesis in Computer Science

A General Framework for Image Kernel Engineering

Annalisa Barla

April, 2005

Dottorato di Ricerca in Informatica
Dipartimento di Informatica e Scienze dell'Informazione
Università degli Studi di Genova

DISI, Univ. di Genova
via Dodecaneso 35
I-16146 Genova, Italy
http://www.disi.unige.it/

Ph.D. Thesis in Computer Science

Submitted by Annalisa Barla
DISI, Univ. di Genova
barla@disi.unige.it

Date of submission: March 2005

Title: A General Framework for Image Kernel Engineering

Advisor: Prof. Alessandro Verri
Dipartimento di Informatica e Scienze dell'Informazione
Università degli Studi di Genova
verri@disi.unige.it

Ext. Reviewers:

Roberto Manduchi
Computer Engineering Department - University of California
manduchi@soe.uscs.edu

Emanuele Trucco
School of Engineering and Physical Sciences - Heriot-Watt University
E.Trucco@hw.ac.uk

Abstract

Understanding image content is a long standing problem of computer science. Despite decades of research in computer vision, an effective solution to this problem does not appear to be in sight. Recent advances in the theory of learning by examples indicate that devising systems which can be trained instead of programmed to solve this problem is an interesting alternative to solutions constructed from higher level image analysis and description.

In this thesis we consider a number of image understanding problems viewed as classification problems for which a certain number of input/output pairs is given. Within the statistical learning schemes we adopt (binary support vector machines and one-class support vector machines), the solution to each problem is written as a linear combination of certain functions, named kernel functions. These functions, which satisfy some specific mathematical properties, are evaluated on input pairs and encode the prior knowledge on the problem domain. Roughly speaking, kernel functions can be thought of as measuring the similarity between input pairs by extracting certain features from the raw data.

In this thesis we argue for the need of finding appropriate kernel functions for building effective trainable systems. Thus, we proceed to investigate, design, implement, and validate kernels for images, or *engineer kernels*, in this context. In the case of images, the problem of kernel engineering cannot be easily decoupled from the choice of the image description. Therefore, we make use of different descriptions depending on the nature of the considered classification problem. For classification problems like indoor/outdoor classification we choose to represent images through histograms (color, edges, co-occurrences, etc.), while for view-based object recognition we choose grey values and/or wavelets representations.

In our work we introduce and study the mathematical properties of two image kernels: the Histogram Intersection kernel and the Hausdorff kernel. The Histogram Intersection kernel, derived from a similarity measure widely used in the computer vision community for color based object recognition, is found to be a very effective kernel for describing similarities between images in high level classification problems. The Histogram Intersection kernel can be implemented efficiently and does not require the introduction of additional parameters. The Hausdorff kernel, instead, which we derive as a specialization of a larger class of kernels defined on binary strings, appears to be well suited for measuring the similarity between image patches. We show that the Hausdorff kernel can be used to boost the performances of trainable 3D object detection systems trained on positive examples only.

The obtained experimental results confirm that the choice of the appropriate kernel can make the difference for a specific application. We conclude by discussing strengths and weaknesses of the approach and outlining directions for future work.

The bourgeois prefers comfort to pleasure, convenience to liberty, and a pleasant temperature to the deathly inner consuming fire.

(Herman Hesse, "Der Steppenwolf")

Quod Huic Deest Me Torquet.

(F.Gonzaga)

Acknowledgements

Since my *Laurea* I have been thinking about these acknowledgments. Actually, I felt guilty because I did not write any in my previous thesis. This time.. I would like to apologize for it!!!

Therefore, my first "thank you" goes to Francesca, Alessandro and all the SlipGURU group: Lorenzo [the most straightforward of my friends... and this is a gift!!], Emanuele [without you.. it would have been impossible!], Francesco, Bettina, it is a privilege working with you all.

The "overflow dottorandi": Barbara, Cristiano, Gianluca, Michela, Simone.

Francesco, Alessandro, Gabriele, Roberto, Andrea and Michele,("loggionekompleto"), for supporting me, believing in me and let me feel that I am a special and lucky person. The other part of our "clan": Marta, Celotto, Matteo, Claudia, la7i, Isa.

Carlo e Aldo, for teaching me how to defend myself. (And not only in a metaphorical sense!)

The colorful tennis people..AILT! They make C.U.S. my second home. Ernesto.. (GRAZIE... in the real sense...)

Il folletto dei boschi for letting me drive a special fast car.

Some special persons that always support me: la ZiaBianca&ZioCarlo, "i Gori" , Neda, Marina, Dario (my personal opera advisor & LaTex consultant).

My gypsy friend Maxime, the fong... keep on surfing life, man!!

I'd like to say thanks to all the people I met on my way, including the adventures around the world... They definitely changed me in many ways, and widened my perspectives.

The "epitomi" group!!! Because they are a lot of fun!

The Torgnon people... Massimiliano (la mia bussola), Raffaele (salud y libertad), Sergio, Pierpaolo (from whom I stole the citation), the two most special girls i know: Esterina and Sara...

All the special friends I met in Princeton, thanks for sharing with me a great experience.

Maurice, the "Space Cowboy", Amer, constantly pulling my leg, and Michael... standing my clumsiness without freaking out...I still wonder how it was possible!!

My family... including all my good-looking cousins (Laura.. non posso.. proprio non posso...) ...

Minou&Minette,
Gino&Marisa and..
the “spectacular” nonna Maggiorina!!!

Table of Contents

Introduction

This thesis considers a long standing problem of computer vision: image understanding. So far, understanding the semantic of an image is still one of the most challenging open problems in this research area. Here, we address image classification in the statistical learning framework [Vap95, EPP00].

The goal of building systems that can adapt to their environments and learn from their experience has attracted researchers from many fields, including computer science, engineering, mathematics, physics, neuroscience, and cognitive science. Out of this research has come a wide variety of learning techniques that have the potential to transform many scientific and industrial fields. Recently, several research communities have converged on a common set of issues surrounding supervised, unsupervised, and reinforcement learning problems.

In our work we focus on supervised learning, a machine learning technique for creating a function from training data. The training data consist of pairs of input objects (typically vectors), and desired outputs. The output of the function can be a continuous value (regression), or can predict a class label of the input object (classification). The task of the supervised learner is to predict the value of the function for any valid input object after having seen only a finite number of training examples (i.e. pairs of input and target output). To achieve this, the learner has to generalize from the presented data to unseen situations in a "reasonable" way.

Supervised learning can generate models of two types. Most commonly, supervised learning generates a global model that maps input objects to desired outputs. In some cases, however, the map is implemented as a set of local models (such as in the nearest neighbour algorithm).

In order to solve a given problem of supervised learning various steps need to be considered:

1. *Determine the type of training examples.* Before doing anything else, the user should decide what kind of data is to be used as examples. For instance, this might be a single handwritten character, an entire handwritten word, or a certain kind of image.

2. *Gathering a training set.* The training set needs to be characteristic of the real-world use of the function. Thus, a set of input objects is gathered and corresponding outputs are also gathered, either from human experts or from measurements.

3. *Determine the input feature representation of the learned function.* The accuracy of the learned function depends strongly on how the input object is represented. Typically, the input object is transformed into a feature vector, which contains a number of features that are descriptive of the object. The number of features should be large enough to accurately predict the output.

4. *Determine the structure of the learned function and corresponding learning algorithm* (like SVMs, neural networks, or decision trees).

5. *Finalize.* The learning algorithm is trained on the gathered training set, the parameters of the learning algorithm may be adjusted by optimizing performance on a subset (called validation set) of the training set, or via cross-validation. After parameter adjustment and learning, the performance of the algorithm may be measured on a test set separate from the training set.

In this thesis we consider a number of image understanding problems viewed as supervised classification problems for which a certain number of input/output pairs is given. Within the statistical learning schemes we adopt (binary support vector machines and one-class support vector machines), the solution to each problem is written as a linear combination of certain functions, named kernel functions. These functions, which satisfy some specific mathematical properties, are evaluated on input pairs and encode the prior knowledge on the problem domain. Roughly speaking, kernel functions can be thought of as measuring the similarity between input pairs by extracting certain features from the raw data.

An important aspect of statistical learning approaches like Support Vector Machines is which kernel to use for which problem. A number of general purpose kernels is available in the literature but we argue for the need of finding appropriate kernel functions for building effective trainable systems. Thus, our work consisted in investigating, designing, implementing, and validating kernels for images, or *engineering kernels*, in this context. In the case of images, the problem of kernel engineering cannot be easily decoupled from the choice of the image description. Therefore, we make use of different descriptions depending on the nature of the considered classification problem. For classification problems like indoor/outdoor classification we choose to represent images through histograms (color, edges, co-occurrences [BOV02, BFOV02, BOV03a, FOV05]), while for view-based object recognition we choose grey values and/or wavelets representations.

In our work we introduce and study the mathematical properties of two image kernels: the Histogram Intersection kernel and the Hausdorff kernel.

The Histogram Intersection kernel, derived from a similarity measure widely used in the computer vision community for color based object recognition, is found to be a very effective kernel for describing similarities between images in high level classification problems. The Histogram Intersection kernel can be implemented efficiently and does not require the introduction of additional parameters.

The Hausdorff kernel, instead, which we derive as a specialization of a larger class of kernels defined on binary strings, appears to be well suited for measuring the similarity between image patches. We show that the Hausdorff kernel can be used to boost the performances of trainable 3D object detection systems trained on positive examples only.

Structure of the Thesis

This thesis is organized in 5 chapters. In Chapter 1 we review the state of the art in the topics concerning this work: Chapter 2 describes the various image representations which have been used in our work. These include gray-levels, wavelets, color and edge histograms, and co-occurrences. Chapter 3 summarizes the statistical learning parts at the basis of my work. After setting the notation and reviewing the main results, we discuss kernels functions, their definition, meaning, and mathematical properties. Finally, we discuss the two learning schemes we adopted from an optimization viewpoint: support vector machines for binary classification and one-class support vector machines. Chapter 4 deals with the kernel engineering and discuss extensively both the Histogram Intersection kernel and the Hausdorff kernel (as a specialization of a kernel for binary strings). Chapter 5 describes the experimental results obtained in different tasks of image classification. The presented results are often better than state of the art and sometimes much better. Finally we draw the conclusions, discuss strenghts and weaknesses of the proposed approach, and lay down the path for future work in the attempt to overcome the limitations of our approach.

Chapter 1

State of the Art

We are going to review the state of the art in image representation and classification. Some methods typically used to describe images will be surveyed, either based on color properties or on texture features. Furthermore, we will focus on the main algorthms used in machine learning to solve the problem of classification: nearest neighbor, discriminant functions, perceptrons, neural networks and SVMs, to name some. Then we will consider some applications that have been considered in the context of images, emphasizing the ones that we have performed in our experiments (see chapter 5).

1.1 Techniques for Image Representation

By using the term *feature* we refer to a representation of special parts of an image, usually corresponding to interesting elements of the scene. It can refer to two possible entities: a *global property of an image*, as the average gray level intensity or *a part of the image with special properties*. In the reminder of this section we are going to describe and indtroduce

Color Properties

Swain [SB91] introduced the concept of color histograms for indexing objects in image databases, proving that color can be exploited as a useful feature for rapid detection. The use of color for solving classification and recognition problems is clearly in opposition to the main philosophy, which considers geometrical cues the most reliable of object identity.

For a long time color has not been used also because of lack of good algorithms for *color constancy*, that is perceiving a stable and constant perception of color under changing light conditions. This problem has been faced and mainly solved [For90, MW86, NS90,

GJKK88], with good progress for problems with *chromaticity* and geometric effects such as *specularity*.

Texture

A wide variety of techniques for describing image texture have been proposed.

Tuceryan and Jain [JT00] divided texture analysis methods into four categories: statistical, geometrical, model-based and signal processing. Due to the extensive research on texture analysis over the past 30 years it is impossible to list all published methods, but the following description provide short introduction to each of the four categories, together with some key references. For surveys on texture analysis methods see [GGC91, Har80, HS92].

- Statistical methods

  Statistical methods analyze the spatial distribution of gray values, by computing local features at each point in the image, and deriving a set of statistics from the distributions of the local features. Depending on the number of pixels defining the local feature statistical methods can be further classified into first-order (one pixel), second-order (two pixels) and higher-order (three or more pixels) statistics. The basic difference is that first-order statistics estimate properties (e.g. average and variance) of individual pixel values, ignoring the spatial interaction between image pixels, whereas second- and higher-order statistics estimate properties of two or more pixel values occurring at specific locations relative to each other. The most widely used statistical methods are cooccurrence features [HSD73] and gray level differences [WDR76], which have inspired a variety of modifications later on.

- Geometrical methods

  Geometrical methods consider texture to be composed of texture primitives, attempting to describe the primitives and the rules governing their spatial organization.

  The primitives may be extracted by edge detection with a Laplacian-of-Gaussian or Difference-of-Gaussian filter [VP87, TJ90], by adaptive region extraction or by mathematical morphology.

  Once the primitives have been identified, the analysis is completed by computing statistics of the primitives (e.g. intensity, area, elongation, and orientation).

  The structure and organization of the primitives can also be presented using Voronoi tessellations [TJ90]. Image edges are an often used primitive element. In [DM79, DM81] are defined *generalized co-occurrence matrices*, which describe second-order statistics of edges.

- Model-based methods hypothesize the underlying texture process, constructing a parametric generative model, which could have created the observed intensity distribution. The intensity function is considered to be a combination of a function representing the known structural information on the image surface and an additive random noise sequence [MC93].

- Signal Processing methods

  Signal processing methods analyze the frequency content of the image. Spatial domain filters such as local linear transforms and various masks designed for edge detection are the most direct approach for capturing frequency information. Another class of spatial filters are moments which correspond to filtering the image with a set of spatial masks. The resulting images are then used as texture features.

  "True" frequency analysis is done in the Fourier domain. The Fourier transform describes the global frequency content of an image, without any reference to localization in the spatial domain, which results in poor performance. Spatial dependency is incorporated into the presentation with a window function, resulting in a so-called short-time Fourier transform. The squared magnitude of the two-dimensional version of the short-time Fourier transform is called *spectrogram*, which is used in [BL76] in analysing shape from texture.

  Multiresolution analysis, the so-called *wavelet transform*, is achieved by using a window function, whose width changes as the frequency changes [Mal89]. If the window function is Gaussian, the obtained transform is called the Gabor transform A two-dimensional Gabor filter is sensitive to a particular frequency and orientation. Other spatial/spatial-frequency methods include the Difference-of-Gaussians and the pseudo-Wigner distribution

  Texture description with these methods is done by filtering the image with a *bank of filters*, each filter having a specific frequency (and orientation). Texture features are then extracted from the filtered images.

  Often many scales and orientations are needed, which results in texture features with very large dimensions. Dimensionality can be reduced by considering only those bands, which have high energy [JF91]. Alternatively, redundancy can be reduced by optimizing filter design so that the frequency space is covered in a desired manner. For a detailed discussion on spatial/spatial-frequency methods see [RW90].

1.2 Classification Tools

Machine learning algorithms are organized into a taxonomy, based on the desired outcome of the algorithm. Common algorithm types include:

- Supervised learning: where the algorithm generates a function that maps inputs to desired outputs. One standard formulation of the supervised learning task is the classification problem: the learner is required to learn (to approximate the behavior of) a function which maps a vector into one of several classes by looking at several input-output examples of the function.
- Unsupervised learning: which models a set of inputs: labeled examples are not available.
- Reinforcement learning: where the algorithm learns a policy of how to act given an observation of the world. Every action has some impact in the environment, and the environment provides feedback that guides the learning algorithm.
- Transduction: similar to supervised learning, but does not explicitly construct a function: instead, tried to predict new outputs based on training inputs, training outputs, and new inputs.

Here we are briefly analyse the principle and methodologies of classic statistical learning and some new techniques of modern approach. Statistical learning main problem consists in studying different methods able to learn a solution to problems either of *classification* or *clustering*, based on a certain number of known examples, couples of objects whose class is known a priori.

Classical major techniques are based on bayesian theory and they include memory-based classifiers (as nearest neighbour) nonparametric methods, discriminant functions, perceptrons, neural networks, bagging, boosting and principal component analysis (PCA).

We are about to review some of the classic techniques, described in [Bis96, DHS00, HTF01] and then briefly introduce the modern statistical learning theory, introduced in the last decades [Vap95, Vap98].

Statistical learning is a framework which aim to solve a learning problem, that consists in find a probabilistic relation between two random variables *input* and *output*, starting from a number of examples (pairs of realizations of the random variables), called the *training set*. The problem is not trivial, since probability distributions are not known a priori.

Many classification, regression, clustering or data mining problems can be faced within the statistical learning framework. The input random variable is usually a *descriptor* of the object. Tipically, every input data is represented by a vector in some space (usually characterized by high dimensionality), while the output variable is identifying the class (in classification tasks) or the group (in case of clustering).

Probabilistic nature of the relation between data can depend on many different factors, including noise (such as grey level values in images), lack of necessary information or

natural ambiguity in classification, due to the nature of the data (i.e. a text can belong to different categories).

Nonparametric Methods: Nearest Neighbor and k-Nearest Neighbor

Given ℓ pairs of training examples:

$$\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell)\},$$

with $\mathbf{x}_i \in \mathbb{R}^n$ and $y_i = \{-1, 1\}$.

The *Nearest Neighbor* rule assigns to a new input point the class of the nearest training point. It is necessary to introduce a *metric*, in order to be able to evaluate distances. This method is really simple and asintotically this guarantees a superior limit to the average error, with an increasing number of points. This approach can be extended: we classify the new point by evaluating the k nearest points, and assigning to it the class of the majority. It canbe shown that different metrics lead to the same classification when the number of points is infinite. The good asymptotic properties are not preserved when the number of examples is realtively small. Different metrics give different classifications. Moreover, this method show its weakness in three main points: it has no theoretical results in non asymptotic case, it slowly converges to the decision function and it is computationally heavy.

Discriminant functions

A direct application of Bayes rule are discriminant functions. This method is applied when some prior knowledge is available on the probability distribution of the data.

These functions assume positive values for one class ($y = 1$) and negative in the other case ($y = -1$). Points such that the function equals zero are points belonging to the *decision surface*

An example of function implementing Bayes is:

$$g(\mathbf{x}) = \log \frac{p(\mathbf{x}|y{=}\,1)P(y{=}\,1)}{p(\mathbf{x}|y{=}\,-1)P(y{=}\,-1)}.$$

NOte that $g(\mathbf{x}) = g_1(\mathbf{x}) - g_{-1}(\mathbf{x})$ with:

$$g_m(\mathbf{x}) = \log(p(\mathbf{x}|y{=}\,m)P(y{=}\,m)), \quad m = -1, 1.$$

If we assume that the probability distribution is gaussian, then we can write some particular cases, involving the *covariance* matrix, which is defined as follows:

If $x \in \mathbb{R}$ we write

$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp^{-(x-\mu)^2/2\sigma^2}$$

as the gaussian density with average $\mu = E[x]$ and variance $\sigma^2 = E[(x-m)^2]$.

If $\mathbf{x} \in \mathbb{R}^n$ we write

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}} \exp^{-(\mathbf{x}-\boldsymbol{\mu})^\top \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})/2}$$

with $\boldsymbol{\mu} = E[\mathbf{x}] \in \mathbb{R}^n$ and

$$\Sigma = E[(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^\top]$$

as the *covariance* matrix $(n \times n)$, whose determinant is written as $|\Sigma|$.

Some problems are connected with this approach. The adopted model is usually too simple and depends on knowledge or estimate of parameters (in this case the covariance matrices). In many real problems this leads to a high dimensionality of the problem, sometimes unfeasible.

Fisher linear discriminant

Instead of choosing to estimate big quantities of parameters one can choose to reduce the dimensionality of the input data, trying to avoid intersection and superposition of the two classes.

Given a training set of ℓ pairs:

$$\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_\ell, y_\ell)\},$$

with $\mathbf{x}_i \in \mathbb{R}^n$, ℓ_1 pairs with $y_i = 1$, ℓ_{-1} pairs with $y_i = -1$, and $\ell_1 + \ell_{-1} = \ell$, We evaluate dot product:

$$t_i = \mathbf{w}^\top \mathbf{x}_i, \quad i = 1, ..., \ell$$

for some versor $\mathbf{w} \in \mathbb{R}^n$.

The idea behind is to determine the versor $\mathbf{w}$ that optimally separates the input data, using only the information stored in $t_1, ..., t_\ell$.

This approach is sub-optimal, but heavily reduces the dimensionality of the problem.

Perceptron

Let us now consider classification methods hat estimate the coefficients of a linear function directly from examples. If $\mathbf{x} \in \mathbb{R}^n$, we consider linear discriminant functions such as: $f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$ where $\mathbf{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$. The classification is baed on the sign of:

$$U(\mathbf{w}^\top \mathbf{x} + b)$$

where $U(\cdot)$ is the unitary step function.

When the f us a linear function, we are using *Perceptrons*. If the *training set* is linearly separable, there exist $\mathbf{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ such that:

$$(\mathbf{w}^\top \mathbf{x}_i + b) y_i > 0, \quad i = 1, ..., \ell.$$

In this case a very important property holds. A perceptron separating the training set properly is qbtained in a finite number of steps.

Neural Networks

Natural limits for a linear classifier can be improved using a linear superposition of perceptrons, considering functions like:

$$f(\mathbf{x}) = \sum_{i=1}^{N} c_i \sigma(\mathbf{w}_i^\top \mathbf{x} + b_i)$$

where σ is the step function or, more generally, a sigmoid function. The parameters of this network are determined from the examples of training set, solving a minimization problem. This is a very though task, due to the existence of many local minima.

Neural Networks have been successfully used in some specific tasks (such as character recognition), even if they are strongly dependent on a big number of parameters, giving to this technique a strongly empirical nature.

Principal Component Analysis

This technique is used to reduce dimensionality of input data, indipendently from the class they belong to. The basic idea behind is that the training set $\mathbf{x}_1, \ldots, \mathbf{x}_\ell \in \mathbb{R}^n$, can be represented in a more compact way if it elements approximatively are extracted from a subspace with dimension $m << n$. Basically we assume that there is an *affinity* on the data even if they belong to two different classes. PCA describes each input point using m numbers instead of n, evaluating how good is the approximation done.

Its limit consists in this assumption, meaning that if the data do not belong to a homogeneous subset of the input space, then dimensionality reduction is done by reducing the necessary information for a correct classification.

Now we focus on some modern approaches due to the statistical learning theory developed in the last decades [Vap98, EPP00, CS02].

Given two sets of random variables $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$ and $y \in \mathcal{Y} \subseteq \mathbb{R}$ and an unknown fixed joined probability distribution $p(\mathbf{x}, y)$, ℓ pairs are *i.i.d.* sampled from it:

$$D_\ell \equiv \{(\mathbf{x}_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}_{i=1}^{\ell}.$$

Learning means looking for a function $f_0 : \mathcal{X} \to \mathcal{Y}$ that, given a $\mathbf{x}$ is able to evaluate the correspondent label y.

We will review SLT in detail in the remaining of this thesis. Here we briefly scketch some algorithms commonly used.

- Support Vector Machines

  SVMs are a binary classification tool that have been applied successfully to a large number of different categorization binary problems. They can perform binary either classification (pattern recognition) or real valued function approximation (regression estimation) tasks. They are the main example of *kernel methods*, using the idea of kernel substitution to identify the decision surface (which is the goal of the learning problem) by solving a linear problem in some *feature space*. SVMs and Kernel Methods have been studied and discussed widely in the last decade [CV95, BV96, EOG, PV, Cam00, CST00] and sistematically motivated in the framework of statistical learning [Vap95, Vap98].

  Support Vector Machines in Vision

  Schölkopf [Sch] was the first to apply SVMs to recognize 3D objects from 2D images and has demonstrated the potential of this approach in visual learning. Pontil and Verri [PV] also used SVMs for 3D object recognition and experimented with a subset of the COIL-100 dataset. A subset of the COIL-100 has been used by also Roobaert and Van Hulle [RH99] to compare the performance of SVMs with different pixel-based input representations. Recently, SVMs have been applied to several visual learning tasks including face detection [HP00, EOG], pedestrian detection [MOP, CP], and gender classification [MY].

  In some cases it is difficult to collect examples of both classes, while it is easy to list items belonging to one class. In these cases it is useful applying one-class classification techniques [Tax01].

- One-class methods

Several models have been proposed for one-class classification. A complete survey can be found in [Tax01].

Most often the methods focus on outlier detection. Conceptually the most simple solution for outlier detection is to generate outlier data around the target set. Then an ordinary classifier is trained to distinguish between the target data and outliers [RS].

Koch [KF95] used ART-2A and a Multi-Layered Perceptron for the detection of (partially obscured) objects in an automatic target recognition system.

The methods scale very poorly in high dimensional problems, especially when the near-target data has to be created and is not readily available.

In classification or regression problems a more advanced Bayesian approach can be used for detecting outliers [Bis94, Mac92].Instead of using the most probable weights for a classifier (or regressor) to compute the output, the output is weighted by the probability that the weights for the classifier or regressor is correct, given the data. T he classifier outputs are automatically moderated for objects remote from the training domain. These methods are not optimized for outlier detection and they require a classification (or regression) task to be solved.

When just a set of objects is available, these methods cannot be used. They also tend to be computationally expensive.

Another possible approach is to use a density method which directly estimates the density of the target objects $p(x|\omega_T)$ [BL84]. By assuming a uniform outlier distribution and by the application of Bayes rule the description of the target class is obtained. Only the prior probabilities of the target and outlier class, $p(\omega_T)$ and $p(\omega_O)$ should be chosen beforehand. This directly influences the choice where the probability $p(\omega_T|x)$ should be thresholded to obtain a target and an outlier region. For instance, in [Bis94] the density is estimated by a Parzen density estimator.

In some cases, prior knowledge might be available and the generating process for the objects can be modeled. When it is possible to encode an object x in the model and to reconstruct the measurements from this encoded object, the reconstruction error can be used to measure the fit of the object to the model.

It is assumed that the smaller the reconstruction error, the better the object fits to the model and the more likely that it is not an outlier. These methods will therefore be called the *reconstruction methods*.

Finally, *boundary methods* have been developed which only focus on the boundary of the data. They try to avoid the estimation of the complete density of the data (which might be impossible from small sample sizes) and therefore also work with an uncharacteristic training data set. These pure one-class classification methods are relatively new and are completely focused on the problem of one-class classification

(the other methods were mainly used in conventional classification problems). The main advantage of these methods is that they avoid the estimation of the complete probability density. This not only gives an advantage when just a limited sample is available, it is even possible to learn from data when the exact target density distribution is unknown. Although in principle the boundary methods are more efficient than the density estimation, it is not directly clear how one should define a boundary around a target set, how to define the resemblance of an object x to a target set and where to put the threshold.

Kernel Methods for One-class categorization tasks

In order to solve problems where only one class of elements is easily describable, some kernel methods have been proposed.

In [TD99, TYD] the basic idea is to find a hypersphere with a minimal radius R and centre $\mathbf{x}_0$ which contains most of the data. This problem can be algorithmically solved by quadratic programming problem.

An alternative approach has been proposed [SPST$^+$01], restricting the attention to RBF kernels. The goal consists in finding the hyperplane maximally distant from the origin and with all the input points lying on the opposite side from the origin.

A simpler version of the previous one has been proposed in [CB01] and can be solved by linear programming techniques. Instead of looking for the hyperplane maximally distant from the origin (which represents in a way the "negative class"), the idea is finding the hyperplane closer to the cetre of the positive data distribution.

1.3 Image Classification

Many different applications have been considered in the context of images. What follows concentrate on those problems that most have attracted our attention.

- Indoor/Outdoor

  Indoor/outdoor classification is difficult problem. A number of attempts have been made to cassify the images by mapping low-level features to highlevel semantics.

  Szummer et al. [SP98] proposed an algorithm for indoor/outdoor classification based on the k-NN classifiers and three types of features, one each for color, texture and frequency information. Vailaya et al. [AVZ] formalized the classification problem into the Bayesian framework using vector quantization and proposed the hierarchical classification to classify the images into indoor/outdoor classes at the highest

level. Luo ans Savakis [LA]use low-level features such as color in a Bayesian network, integrating knowledge from low-level and midlevel features.

- Cityscape/Landscape

  This problem has been faced [AVZ98, GP] by using color and texture features on the entire image or on image subsections. One of the issues when dealing with a diverse set of features is how to integrate them into a classification engine. The solution proposed in [AVZ98, VZ] has been used to independently classify image subsections and obtain a final result using a majority classifier. In [RYH] SVM ensembles are proposed to address the rare class problem. Various classifier combination strategies are investigated, including majority voting, sum rule, neural network gater and hierarchical SVMs.

- Face detection/ 3D object detection

  Face recognition has become a popular area of research in computer vision and one of the most successful applications of image analysis and understanding. Because of the nature of the problem, not only computer science researchers are interested in it, but neuroscientists and psychologists also. In [BHP00, HSPP01, HHP01] a trainable system for detecting frontal and near-frontal views of faces in still gray images using Support Vector Machines (SVMs).

  In [JHH] a novel approach to view and pose invariant face recognition is presented. It combines two recent advances in the computer vision field: component-based recognition and 3D morphable models. More generally, in [SK] a statistical method for 3D object detection is proposed. The statistics of both object appearance and "non-object" appearance using a product of histograms. Each histogram represents the joint statistics of a subset of wavelet coefficients and their position on the object. The approach is to use many such histograms representing a wide variety of visual attributes. Using this method, the algorithm can detect human faces with out-of-plane rotation and the first algorithm that can reliably detect passenger cars over a wide range of viewpoints.

1.4 Content Based Image Retrieval Systems

Finally, we briefly review the final and natural goal of learning applied to images, analyzing the typical structure of a *content based image retrieval system.*

There has been a growing demand for image and video data in applications due to the significant improvement in the processing technology, network subsystem and availability of large storage systems. This demand for visual data has spurred a significant interest

in the research community to develop methods to archive, query and retrieve this data based on their content. Such systems employ many *pattern recognition* methods developed over the years. The term *pattern recognition methods* refer to their applicability in feature extraction, feature clustering, generation of database indexes and determining similarity in content of the query and database elements.

Current indexing practice for images relies largely on text descriptors or classification codes, supported in some cases by text retrieval packages designed or adapted specially to handle images. Again, remarkably little evidence on the effectiveness of such systems has been published. User satisfaction with such systems appears to vary considerably.

A *Content Based Image Retrieval* (CBIR) system operates on a totally different principle from keyword indexing. Primitive features characterizing image content, such as colour, texture and shape, are computed for both stored and query images, and used to identify the stored images most closely matching the query. Semantic features such as the type of object present in the image are harder to extract, though this remains an active research topic. Video retrieval is a topic of increasing importance here, CBIR techniques are also used to break up long videos into individual shots, extract still keyframes summarizing the content of each shot, and search for video clips containing specified types of movement.

The effectiveness of current CBIR systems [VAF97] is that they can operate only at the primitive feature level. None of them can search effectively for, say, a photo of a dog though some semantic queries can be handled by specifying them in terms of primitives. A beach scene, for example, can be retrieved by specifying large areas of blue at the top of the image, and yellow at the bottom. There is evidence that combining primitive image features with text keywords or hyperlinks can overcome some of these problems, though little is known about how such features can best be combined for retrieval.

Standards development relevant to CBIR can be grouped under three headings image compression, query specification and metadata description.

To analyze in depth what kinds of query are usually required in an image database requires a detailed knowledge of user needs, why users seek images, what use they make of them, and how they judge the utility of the images they retrieve.

Common sense evidence suggests that still images are required for a variety of reasons, including:

- illustration of text articles, conveying information or emotions difficult to describe in words,
- display of detailed data (such as radiology images) for analysis, formal recording of design data (such as architectural plans) for later use.

Access to a desired image from a repository might thus involve a search for images depicting

specific types of object or scene, evoking a particular mood, or simply containing a specific texture or pattern. Potentially, images have many types of attribute which could be used for retrieval, including:

- the presence of a particular combination of colour, texture or shape features (e.g. green stars);
- the presence or arrangement of specific types of object (e.g. chairs around a table);
- the depiction of a particular type of event (e.g. a football match); the presence of named individuals, locations, or events (e.g. the Queen greeting a crowd);
- subjective emotions one might associate with the image (e.g. happiness);
- metadata such as who created the image, where and when.

Each listed query type (with the exception of the last) represents a higher level of abstraction than its predecessor, and each is more difficult to answer without reference to some body of external knowledge. This leads naturally on to a classification of query types into three levels of increasing complexity [Eak96, Eak].

- Level 1 comprises retrieval by primitive features such as colour, texture, shape or the spatial location of image elements. Examples of such queries might include "find pictures with long thin dark objects in the top left-hand corner", "find images containing yellow stars arranged in a ring" or most commonly "find me more pictures that look like this".

  This level of retrieval uses features (such as a given shade of yellow) which are both objective, and directly derivable from the images themselves, without the need to refer to any external knowledge base. Its use is largely limited to specialist applications such as trademark registration, identification of drawings in a design archive, or colour matching of fashion accessories.

- Level 2 comprises retrieval by derived (sometimes known as logical) features, involving some degree of logical inference about the identity of the objects depicted in the image. It can usefully be divided further into:

  1. retrieval of objects of a given type
  2. retrieval of individual objects or persons

  To answer queries at this level, reference to some outside store of knowledge is normally required.

- Level 3 comprises retrieval by abstract attributes, involving a significant amount of high-level reasoning about the meaning and purpose of the objects or scenes depicted. Again, this level of retrieval can usefully be subdivided into:
  - retrieval of named events or types of activity
  - retrieval of pictures with emotional or religious significance

Success in answering queries at this level can require some sophistication on the part of the searcher. Complex reasoning, and often subjective judgement, can be required to make the link between image content and the abstract concepts it is required to illustrate. Queries at this level, though perhaps less common than level 2, are often encountered in both newspaper and art libraries.

This classification of query types can be useful in illustrating the strengths and limitations of different image retrieval techniques. The most significant gap at present lies between levels 1 and 2.

Many authors [NV, NV95] refer to levels 2 and 3 together as semantic image retrieval, and hence the gap between levels 1 and 2 as the semantic gap . Note that this classification ignores a further type of image query retrieval by associated metadata such as who created the image, where and when. This is not because such retrieval is unimportant. It is because (at least at present) such metadata is exclusively textual, and its management is primarily a text retrieval issue.

Despite the shortcomings of current CBIR technology, several image retrieval systems are now available as commercial packages, with demonstration versions of many others available on the Web.

Chapter 2

Representing Visual Information

In dealing with image classification and characterization problems, data can be represented in many different ways. The simplest way to represent image information is to consider the image bitmap representation. Alternatively, one can consider a list of features based either on global or local properties. Tipically, we expect that a good representation for a certain task is the best compromise between loss of information and enhancement of image characteristics.

Some representation methods describe the image in a "punctual" way. This means that the description does not lose information and that it would be possible to rebuild the image from it. In this category we include gray level vectors and wavelets. In the first case a gray-level image is represented by a vector with its gray values as entries. Wavelets, on the other side, describe the image in terms of its behavior in spatial frequency.

Image description can also be achieved by "statistical methods", as histograms and co-occurrence matrixes, which do not preserve all the information contained. In particular, an histogram is a first order statistic on the image, carrying the informaion on values frequency and, therefore, characterized by the bin number. Co-occurrences matrixes are a statistic of the second order. This means that they are a measure of occurrence of value pairs all over the image. Moreover, the information brought out is usually evaluated through measures such as average, variance and entropy.

In this chapter we summarize the image representations adopted in our work, mostly based on color and texture information.

2.1 Visual Cues

2.1.1 Color Information

Color images can be represented by three matrices, one for each primary color component: as an example, in the RGB space, red (R), green (G) and blue (B). Using these three primary colors one can represent most of the visible colors.

However, the human eye is much more sensitive to luminance differences than to color differences. One often uses a different decomposition, similar to the color encoding scheme used in the color television system. Now a color image is represented by the following three matrices:

- Luminance matrix Y: This is the luminance of the image, a weighted mean of the three primary color components,
- Chrominance matrix U: This is the red difference, i.e. the difference of the red color component and the luminance,
- Chrominance matrix V: This is the blue difference, i.e. the difference of the blue color component and the luminance.

Thus the luminance matrix Y is a grey scale version of the original color image, while the two chrominance matrices U and V contain the color information. The choice of the red and blue difference (and not the green difference) as the chrominance matrices was influenced by the higher sensitivity of the human eye to green light. Other linear combinations of the 3 primary color components are also in common use. The original primary color components R, G and B can be easily reconstructed from the luminance and chrominance matrices.

To make a more general statement, a digital image contains one or more matrices (channels) of integer values, one for each color component. Note that channels are not limited to the traditional three color components. They can contain any spectral data. Each channel can have a different size, to accommodate for subsampled chrominance channels, which are commonly used in e.g. broadcast applications.

Other color representations are also possible. Non-linear color representations take into account the non-linearity in the response of the human eye to light.

Color Spaces

In the case of color images, different color spaces (RGB, CMYK, HSV, etc.) give rise to different image representations that enhance or attenuate certain color features. In spite

of their simplicity, color histograms are very efficient for many practical tasks, as they are stable to occlusions and change of view [SB91]: in building color histograms all information regarding spatial features and spatial correlation is discarded. This leads to invariance to rotation and tolerance to translation, but, on the other hand, limits their use if spatial correlation is an important cue.

The visual experience of the normal human eye is not limited to gray scale, therefore *color* is an extremely important aspect of digital imaging. In a very general sense, color conveys a variety of rich information that describes the quality of objects. The perception of color is allowed by the color-sensitive neurons known as *cones* that are located in the retina of the eye. The cones are responsive to normal light and are distributed with greatest density near the center of the retia, known as *fovea*. The *rods* are neurons that are sensitive at low-light levels and are not capable to distinguish color wavelengths. They are distributed with greates density around the perpiphery of the fovea, with very low density near the line of sight. In the normal human eye, color are sensed as near-linear combinations of long, medium and shot wavelengths, which roughly correspond to the three *primary colors* that are used in the standard video camera systems: Red (R), Green (G) and Blue(B). The way in which visible-ligth wavelengths map to RGB camera color coordinates is a complicated topic, although standard tables have been devised for extensive experiments. A number of other color coordinate systems are also used in image processing, printing and display systems, such as the one used for broadcast television: YIQ (luminance, in-phase cromatic, quadratic chrmoatic) or the HSV (hue, saturation, value). Most of the theory and algorithms in computer vision are developed for single-valued images, but they are used as well onto color images, by applying them separately on the main color channels and recombining the results.

The RGB Model

In the RGB model, an image consists of three independent image planes, one in each of the primary colours: red, green and blue. Specifying a particular colour is by specifying the amount of each of the primary components present. Figure 2.1 shows the geometry of the RGB colour model for specifying colours using a Cartesian coordinate system. The greyscale spectrum, i.e. those colours made from equal amounts of each primary, lies on the line joining the black and white vertices.

This is an additive model, i.e. the colours present in the light add to form new colours, and is appropriate for the mixing of coloured light for example. The additive mixing of the primaries red, green and blue form the three secondary colours yellow (red + green), cyan (blue + green) and magenta (red + blue), and white ((red + green + blue).

The RGB model is used for colour monitors and most video cameras.

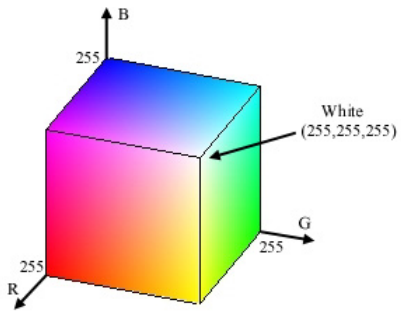

Figure 2.1: RGB color space.

The CMY Model

The CMY (cyan-magenta-yellow) model is a subtractive model appropriate to absorption of colours, for example due to pigments in paints. Whereas the RGB model asks what is added to black to get a particular colour, the CMY model asks what is subtracted from white. In this case, the primaries are cyan, magenta and yellow, with red, green and blue as secondary colours (see fig 2.2).

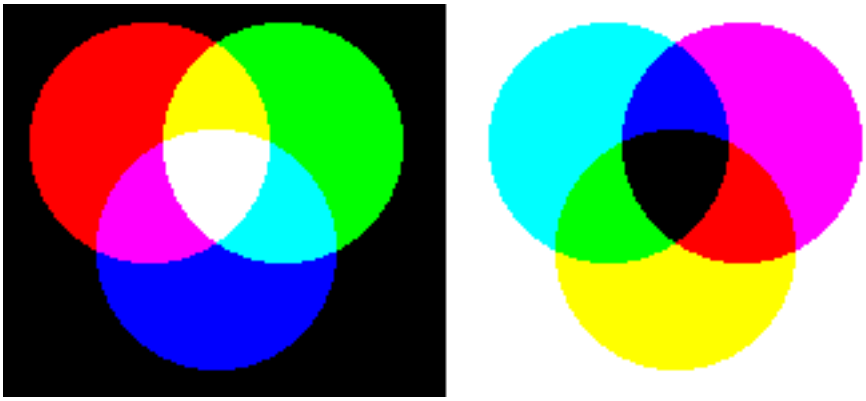

Figure 2.2: Subtractive and additive models.

When a surface coated with cyan pigment is illuminated by white light, no red light is reflected, and similarly for magenta and green, and yellow and blue.

The HSV Model

The Hue-Saturation-Value (HSV) model consists in a circular cone to describe the space. Pure black lies at tht tip of the cone, pure white at the center of the base, and pure hues around the perimeter of the base.

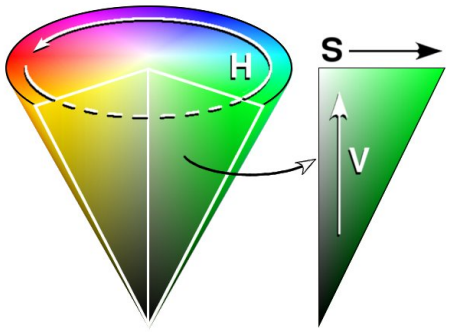

Figure 2.3: HSV color cone.

Sometimes the HSV color model is preferred over alternative models such as RGB or CMYK, because of its similarities to the way humans tend to perceive color. RGB and CMYK are additive and subtractive models, respectively, defining color in terms of the combination of primary colors, whereas HSV encapsulates information about a color in terms that are more familiar to humans: color, vibration (saturation) and brightness.

The HSV (tristimulus space) does not technically support a one-to-one mapping to physical power spectrum as measured in radiometry. Thus it is not generally advisable to try to make direct comparisons between HSV coordinates and physical light properties such as wavelength or amplitude. However, if physical intuitions are indispensable, it is possible to translate HSV coordinates into pseudo-physical properties using the psychophysical terminology of colorimetry as follows:

- Hue specifies the dominant wavelength of the color, except in the range between red and indigo (somewhere between 240 and 360 degrees) where the Hue denotes a position along the line of pure purples.
- The value is roughly analogous to the total power of the spectrum, or the maximum amplitude of the light waveform. However, it should be obvious from the equations below that value is actually closer to the power of the greatest spectral component.

2.1.2 Edges

We use the Canny edge detector to extract edges and estimate their orientation. Histograms were obtained dividing the range between 0° and 180° degrees in 10 equally space intervals and adding a further bin for non-edge pixels (cityscape/landscape). In other cases we use histogram of gradient magnitude (see figure 2.4).

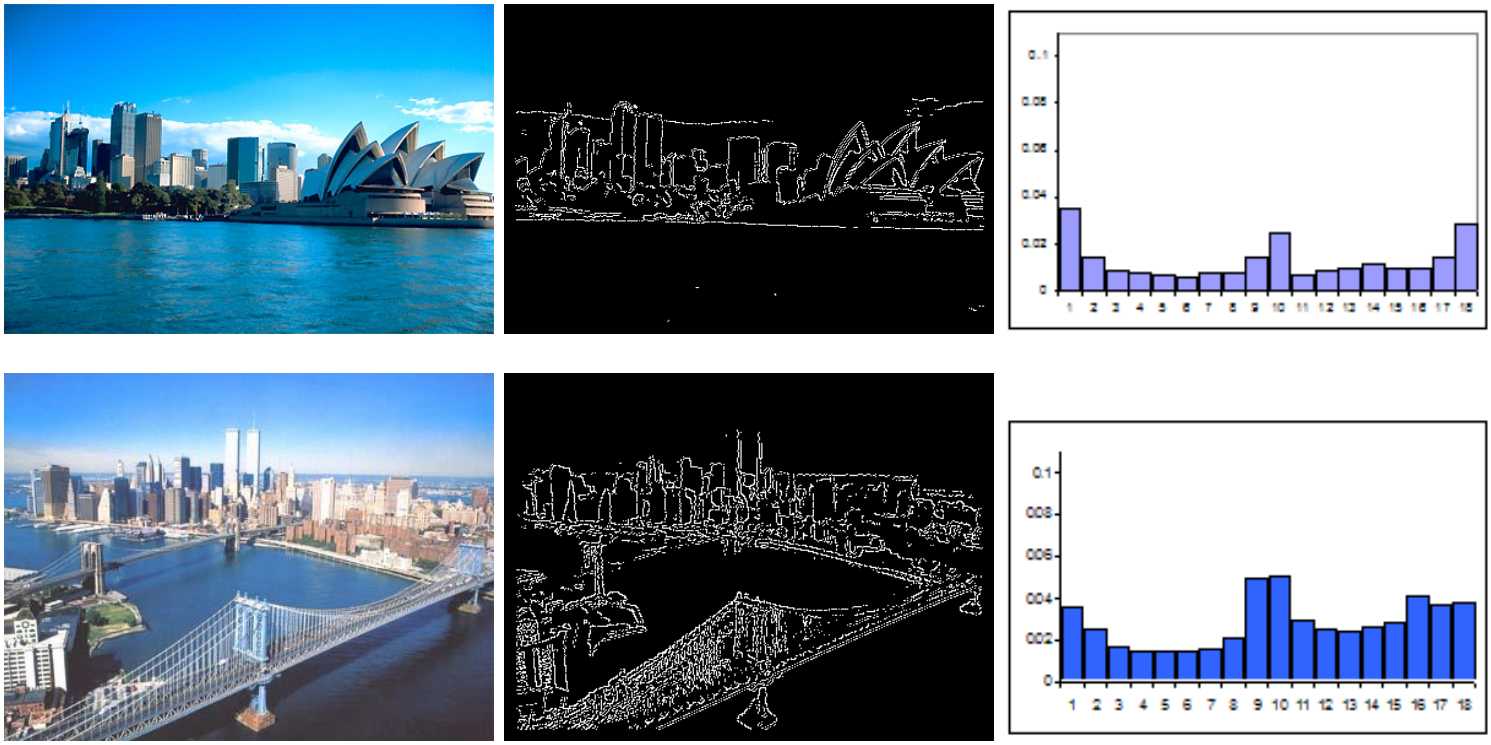

Figure 2.4: Color images, binary images after edge extraction and the corresponding edge direction histograms.

Canny Edge Detector

The Canny edge detector [Can86] arises from the earlier work of Marr and Hildreth [MH79], who were concerned with modelling the early stages of human visual perception. In designing his edge detector he concentrated an ideal step edge, represented as a Sign function in one dimension, corrupted by an assumed Gaussian noise process. In practice this is not an exact model but it represents an approximation to the effects of sensor noise, sampling

and quantisation. The approach is based strongly on convolution of the image function with Gaussian operators and their derivatives, so we shall consider these initially.

The goals of the Canny Operator were stated explicitly:

- Good Detection: the ability to locate and mark all real edges.
- Good Localisation: minimal distance between the detected edge and real edge.
- Clear Response: only one response per edge.

To fulfil these objectives, the edge detection process can be described by the following stages.

1. Image Smoothing

   The image data is smoothed by a two dimensional Gaussian function of width specified by a user parameter. In practice, two dimensional convolution with large Gaussians takes a long time, so that in practice it is common to approximate this by two one dimensional Gaussians, one aligned with the x-axis, the other with the y-axis. This produces two (rather than one) values at each pixel.

2. Differentiation

   The smoothed image data is differentiated with respect to the x and y directions. It is possible to compute the gradient of the smooth surface of the convolved image function in any direction from the known gradient in any two directions.

   From the computed x and y gradient values, the magnitude and angle of the slope can be calculated.

3. Non-maximum Suppression

   Having found the rate of intensity change at each point in the image, edges must now be placed at the points of maxima, or rather non-maxima must be suppressed. A local maximum occurs at a peak in the gradient function, or alternatively where the derivative of the gradient function is set to zero. However, in this case we wish to surpress non-maxima perpendicular to the edge direction, rather than parallel to (along) the edge direction, since we expect continuity of edge strength along an extended contour.

   If the pixel under consideration is not greater than these two values (i.e. non-maximum), it is suppressed.

4. Edge Thresholding

   The thresholder used in the Canny operator uses a method called "hysteresis". Hysteresis thresholders set an upper and lower edge value limit. Considering a line segment, if a value lies above the upper threshold limit it is immediately accepted. If the value lies below the low threshold it is immediately rejected. Points which lie between the two limits are accepted if they are connected to pixels which exhibit strong response.

   It is recommended that the ratio of high to low limit be in the range two or three to one, based on predicted signal-to-noise ratios.

2.1.3 Texture

Texture is a phenomenon that it is easy to recognize and difficult to define. It is usually not depending on the scale at which the image is viewed. A rough definition is usually given by thinking of views of large numbers of small objects. As examples see figure 2.5. In general, many surfaces are characterized by ordered occurrences of patterns. (Think of spots or stripes of animals like leopards or zebras).

Three standard problems arise with texture:

- Segmentation

  An image can be divided into components whithin the texture is homogeneous. Textue segmentation involves both representing a texture and determining the basis on which segment boundaries are to be determined.

- Synthesis

  It consists in the opposite problem, constructng large region of texture from small example images.

- Shape from Texture

  This means recovering surface orientation or surface shape from image texture.

Image textures consist of organized patterns of quite regular subelements (called *textons*). One intuitive way to represent textures is to find the textons and then describe the way in which they are laid out. The difficulty with this approach is that there is no known canonical set of textons, meaning that it is not clear what one should look for.

One possible technique is convolving an image with a linear filter, obtaining a representation of the given image on a different basis. This process gives a clear interpretation of

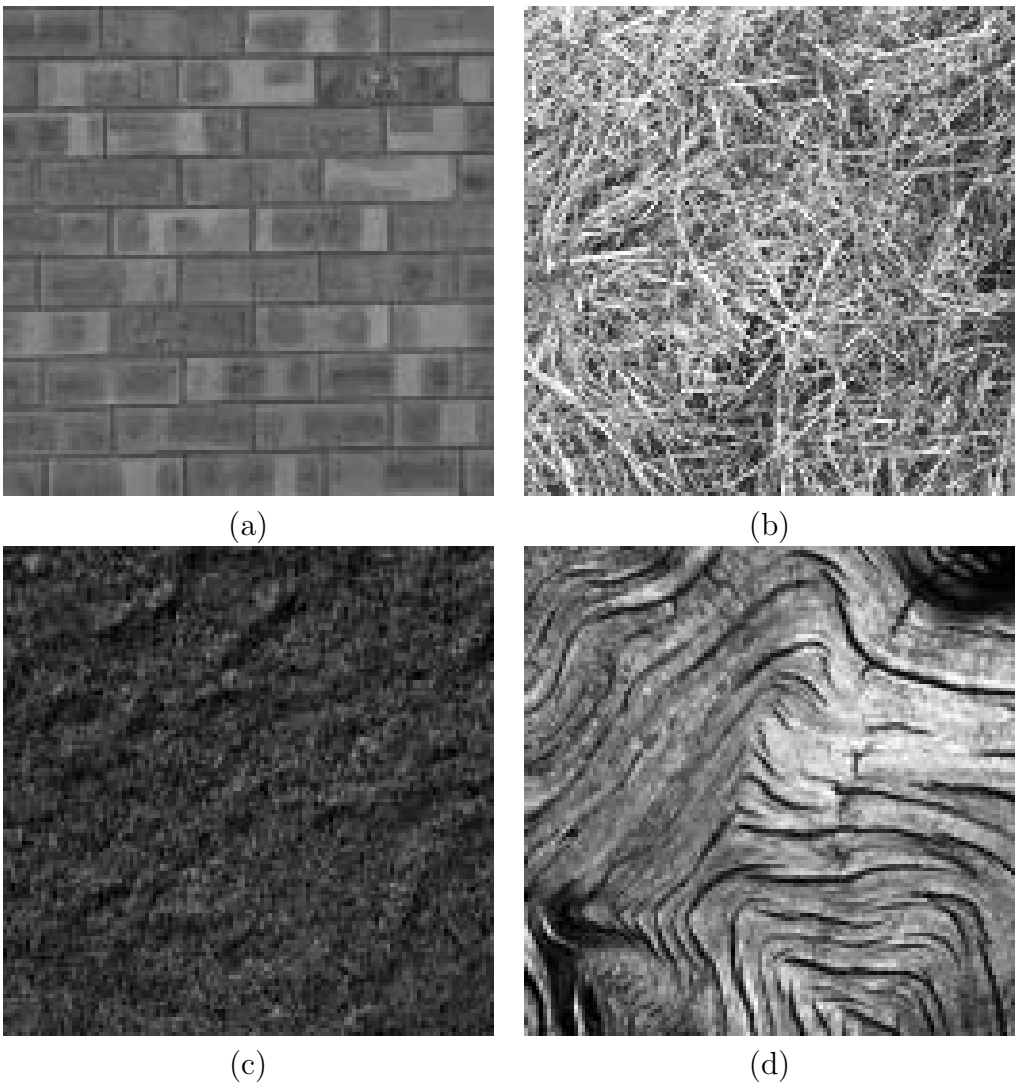

Figure 2.5: Some images characterized by different textures

the local structure of the image. There is a strong response when the image pattern in a neighborhood looks similar to the filter kernel and a weak response when it does not. Thus, image textures can be represented in terms of the response of a collection of filters. The collection of different filters would consist in a series of patterns at different scales. Usually spot filters are used in combination with bar filters, to ehance non oriented structures (with strong differences compared with the neighborhood) and oriented ones.

Many filters have been used for this purpose. In analogy with the human visual cortex it is usual to use at least one spot filter and a collection of oriented bar filters at a different

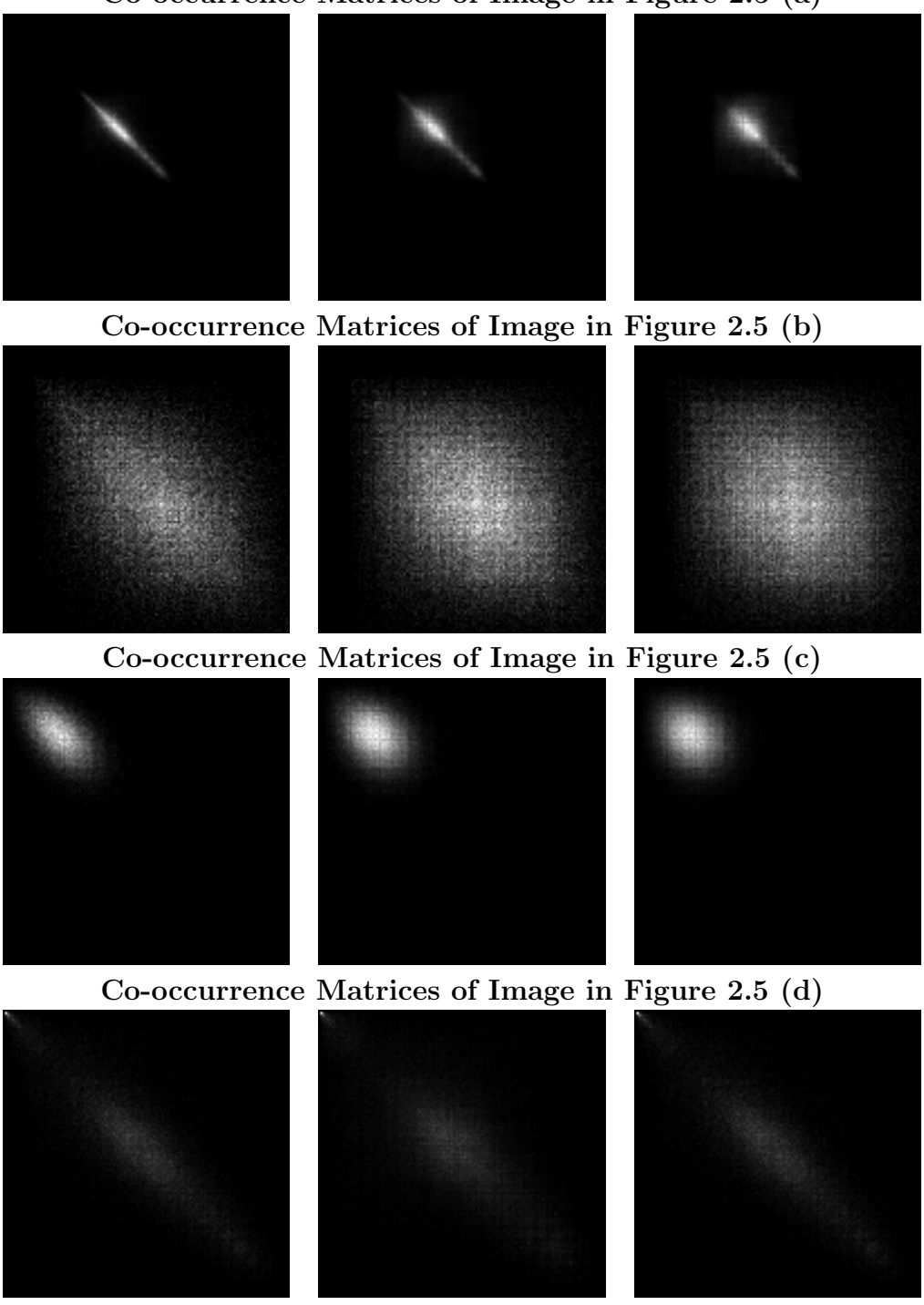

Figure 2.6: Examples of co-occurrence matrices computed on the images of Figure 2.5. First raw contains the co-occurrence matrices computed on image (a) of the table using the maps in table tbl:CoocExampleMaps. The same for second raw but on image in (b), and the third raw on image in (c) and last raw on image in (d).

orientations, scales and phases. The optimal number of filters is generally not known. Many studies have been performed on this topic, see [For03].

A set of filtered images, in itself, is not a representation of texture. A statistics on the overall distribution of texture element is needed. The question of what statistics is needed depends on what is intended to be represented. Supposing that the scale has been set, then one strategy is computing the mean of the squared response of some filters outputs for a range of filters [MP89].

It is possible to represent the content of an image in many different ways, including grey-level and color-level vectors, color histograms, edge direction histograms. Here we list some of the main features of each representation.

2.2 Histograms

As said before, a color is represented by a three dimensional vector corresponding to a position in a color space.This lets us to select the color space and the quantization steps in this color space. In most applications we choose the hue-saturation-value (HSV) space as a color space, which is in bijection with the Red Green Blue (RGB) space. The reason for the choice of HSV is that it is widely used in the literature.

HSV is attractive in theory. It is considered more suitable since it separates the color components (HS) from the luminance component (V) and is less sensitive to illumination changes. Note also that distances in the HSV space correspond to perceptual differences in color in a more consistent way than in the RGB space.

In spite of the fact that the color histogram technique is a very simple and low-level method, it has shown good results in practice [SB91] especially for image indexing and retrieval tasks, where feature extraction has to be as simple and as fast as possible. Spatial features are lost, meaning that spatial relations between parts of an image cannot be used and that the geometrical information is not considered. This also ensures full translation and rotation invariance.

The histogram is easy to compute and is insensitive to small changes in viewing positions. A histogram is a coarse characterization of an image, however, and images with very different appearances can have similar histograms. This must be taken in account if dealing with large databases of images.

The effectiveness of histogram representation is influenced by the number of bins. This parameter has to be an optimal compromise, meaning that too many bins lead to a weak characterization, while few bins imply a very coarse description.

There are many examples in nature where color is closely related to the identity of an object

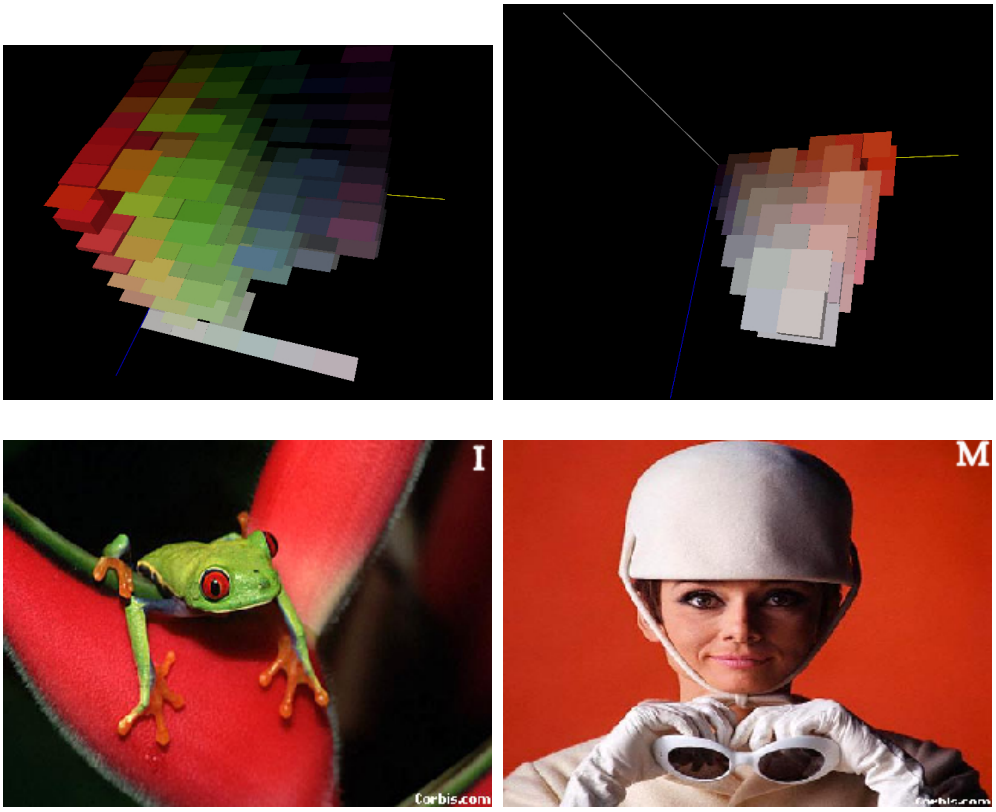

Figure 2.7: Different images with similar color histograms in HSV space $8 \times 8 \times 8$ bins

[SB91] or of a class of objects; for this reason histograms are a suitable image representation for many indexing and categorization tasks – like sorting images in subclasses [CHV99, SB91] and indoor-outdoor classification [SP98], just to name a few.

In the attempt to reduce the dependence on the RGB space actually implemented in each given camera, in what follows we concentrate on HSV color histograms.

2.3 Co-Occurrence Matrixes

Histograms are simple and efficient representations of image content but discard any information about pixels' mutual positions and distances. In some applications this information, which we loosely refer to as *texture*, can be very important. A step beyond histograms, both in terms of complexity and richness of description, is provided by gray-level co-occurrence matrices, well known statistical tools for extracting visual (in particular, texture) information from images [HSD73, LJW97]. A co-occurrence matrix is the 2-D histogram of

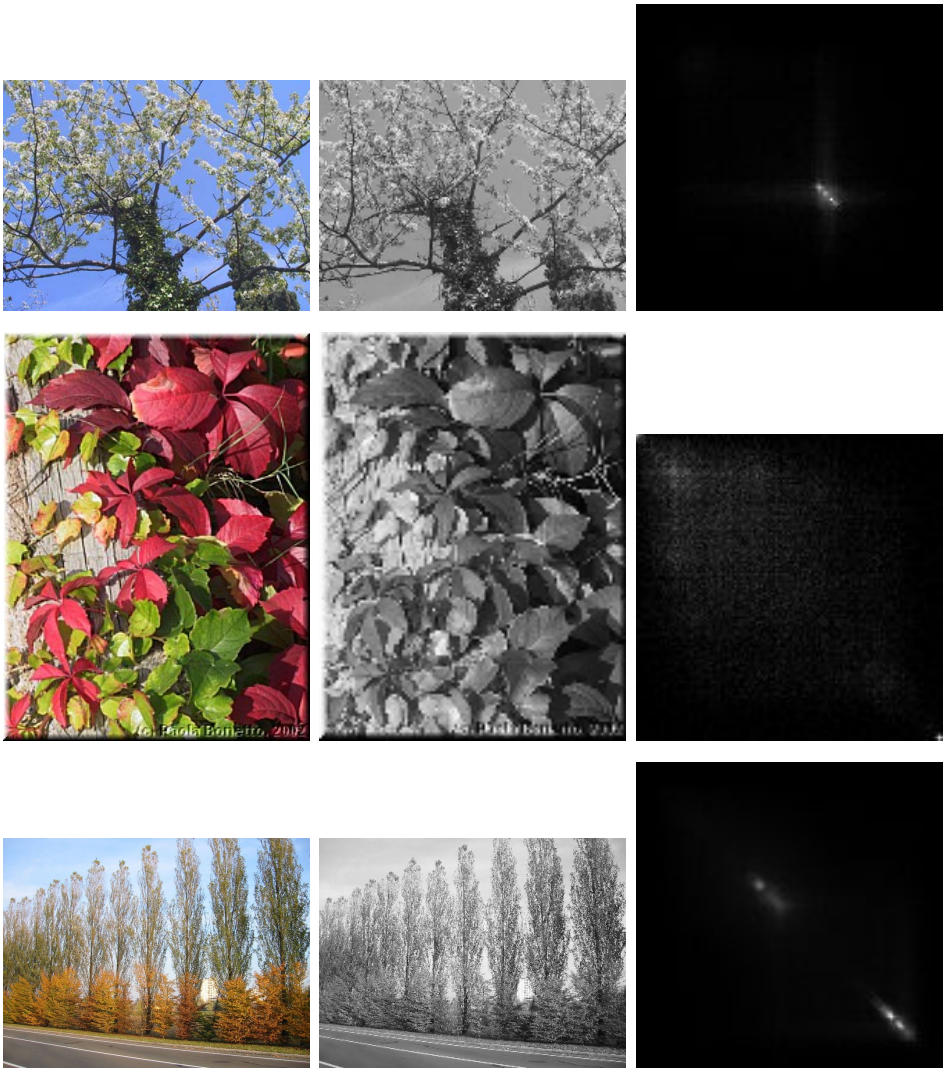

Figure 2.8: Color Images, the greyscale version and the corresponding co-occurrence matrices.

gray-level pairs at some fixed relative position in an image. Likewise image histograms, which can be thought of as an estimate of the probability distribution of gray values or color levels in an image, a co-occurrence matrix is an estimate of the joint probability distribution of pairs of gray-levels at some fixed relative position in the image.

Typically, the information stored in a co-occurrence matrix is sparse. Also, it is often useful to consider a number of co-occurrence matrices, one for each relative position of interest, in order to grasp different textual cues or the same textual cue at different scale. In order to obtain efficient descriptors, the information contained in co-occurrence matrices

is traditionally condensed in a few statistical features (see [HSD73] for a quite exhaustive list) including average, variance, entropy, energy, and correlation.

We let $P_{\mathbf{d}}(i,j)$ be the $(i,j)-th$ element of a normalized symmetric co-occurrence matrix relative to a displacement $\mathbf{d}$ (symmetric because the displacement $-\mathbf{d}$ is also considered). If N is the number of *bins* then one can consider, for example,

- **average** $\mu = \sum_{i,j \in N} i \;\; P_{\mathbf{d}}(i,j)$
- **variance** $\sigma = \sum_{i,j \in N} (i-\mu)^2 P_{\mathbf{d}}(i,j)$
- **entropy** $-\sum_{i,j \in N} P_{\mathbf{d}}(i,j) log P_{\mathbf{d}}(i,j)$
- **energy** $\sum_{i,j \in N} P_{\mathbf{d}}(i,j)^2$
- **correlation** $\sum_{i,j \in N} (i-\mu)(j-\mu) P_{\mathbf{d}}(i,j)$

These measures have been used with success in a number of applications [CTH84, HSD73, WJL95, PCGV02]. They are good descriptors of texture information, but originate very concise image descriptions. Inspired by the success of SVM in addressing classification problems with very sparse representation, like *bag of words* for text classification, we decided to use color histograms and co-occurrence matrices as they are (saving storage space as suggested in [KV98]).

Color Correlograms

Since histograms do not include any spatial information, recently several approaches have attempted to incorporate spatial information with color [Hsu, Stricker, Smith]. These methods, however, lose many of the advantages of color histograms.

A method that computes the spatial correlation of pairs of colors as a function of the distance between pixels is a feature called *color correlogram*, which basically consists in a co-occurrence matrix applied to the three color channels.

A color correlogram expresses how the spatial correlation of pairs of colors changes with distance. Informally, a correlogram for an image is a table indexed by color pairs, where the d-th entry for row (i,j) specifies the probability of finding a pixel of color j at a distance d from a pixel of color i in this image. Here d is chosen from a set of distance values D [JHZ97]. An autocorrelogram captures spatial correlation between identical colors only. This information is a subset of the correlogram and consists of rows of the form (i,j) only.

Since local correlations between colors are more significant than global correlations in an image, a small value of d is sufficient to capture the spatial correlation.

2.4 Pointwise Representations

With "pointwise representation" we mean those description that are lossless. All the information held in the image is stored in the correspondent representation.

2.4.1 Pixel values

The simplest way to represent an image is using a vector with its intensity values. This is naturally a lossless way to represent images, but it has the typical weaknesses of iconic representations. This means that it is strongly noise dependent, thus changes due to noise can lead to very different vectors. A possible solution can be equalization, even if this involves a preprocessing step on the data, influencing the learning phase.

2.4.2 Wavelets

Wavelets are a mathematical tool for hierarchically decomposing functions. They allow a function to be described in terms of a coarse overall shape, plus details that range from broad to narrow. Regardless of whether the function of interest is an image, a curve, or a surface, wavelets offer an elegant technique for representing the levels of detail present. Although wavelets have their roots on approximation theory and signal processing, they have recently been applied to many problems in computer graphics, in many applications including image editing, image compression, image querying.

The main idea behind wavelet analysis is to decompose a signal f into a basis of functions Ψ_i:

$$f = \sum_i a_i \Psi_i.$$

To have an efficient representation of the signal f using only a few coefficients a_i, it is very important to use a suitable family of functions ϕ_i. The functions ϕ_i should match the features of the data we want to represent.

Real-world signals usually have the following features: they are both limited in time (time-limited)[1] and limited in frequency (band-limited). Time-limited signals can be represented efficiently using a basis of block functions (Dirac delta functions for infinitesimal small blocks). But block signals are not limited in frequency. Band-limited signals can be represented efficiently using a Fourier basis, but sines and cosines are not limited in time. What we need is a compromise between the pure time-limited and band-limited basis functions, a compromise that combines the best of both worlds: wavelets ("small waves").

[1] Or space limited in case of images

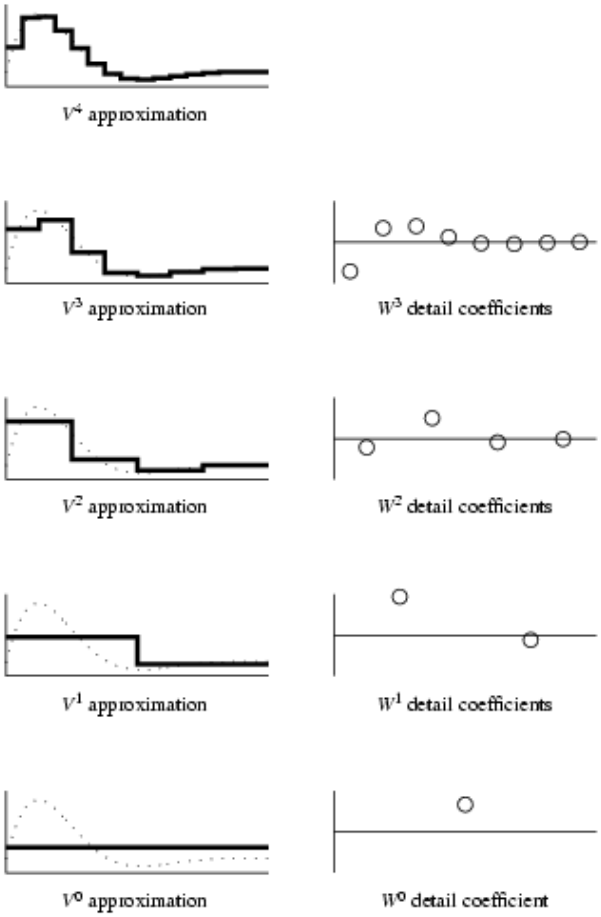

Figure 2.9: A sequence of decreasing-resolution approximations to a function (left) along with the detail coefficients required to recapture the finest approximation (right)

Wavelets are :

- Localized in both the space/time and scale/frequency domains. Hence they can easily detect local features in a signal.
- Based on a multi-resolution analysis. A wavelet decomposition allows to analyze a signal at different resolution levels (scales).
- Smooth, which can be characterized by their number of vanishing moments. A function defined on the interval $[a, b]$ has n vanishing moments if

$$\int_a^b f(x)x_i dx = 0, \ for \ i = 0, 1, \ldots, n-1.$$

The higher the number of vanishing moments, the better smooth signals can be approximated in a wavelet basis. Furthermore there exist fast ($O(n)$) and stable (wavelets can be orthogonal or biorthogonal) algorithms to calculate the discrete wavelet transform and its inverse.

More in detail, wavelets have their roots in approximation theory and signal processing. Recently, they have been succesfully applied to many problems in *computer graphics*.

Many image processing operations like noise reduction and image scaling can be done on wavelet transformed images. Furthermore wavelets can also be used for image compression, giving better performance (higher signal-to-noise ratios, better visual quality and higher compression rates) than other compression methods. This combination of image processing and compression is an important feature of wavelet-based processing.

Wavelets are computed by averaging and differencing coefficients recursively. One advantage of the wavelet is that often a large number of the detail coefficients turn out to be very small. Truncating or removing these coefficients introduces small errors in the reconstructed signal giving a form of *"lossy"* compression [G.K, ED, BP].

Let V^j be the space of all piecewise constant functions on $[0, 1]$ with the interval divided in 2^j pieces.

A 2^j pixel image is an element of V^j, such that:

$$V^0 \subset V^1 \subset ... \subset V^j \subset ...$$

The basis functions for V^j are *scaling functions*

$$\Phi_i^j = \Phi(2^j x - i) \qquad i = 0, 1, ..., 2^j - 1$$

with

$$\Phi(x) = \begin{cases} 1 & 0 \leq x \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

a basis for V^2 is shown as an example in figure 2.10.

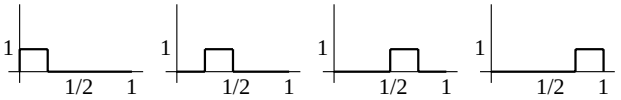

Figure 2.10: Haar basis for V^2: $\Phi_0^2, \Phi_1^2, \Phi_2^2$ and Φ_3^2

In order to deal with image analysis it is necessary to generalise Haar Wavelets in two dimensions. Two main techniques are used, depending on the algorithm used to decompone the two-dimensional signal.

- Standard Decomposition:

  first apply 1D-wavelet to each row, next apply 1D-wavelet to each column treating the transformed rows as if they where 1D-images (see figure 2.11).

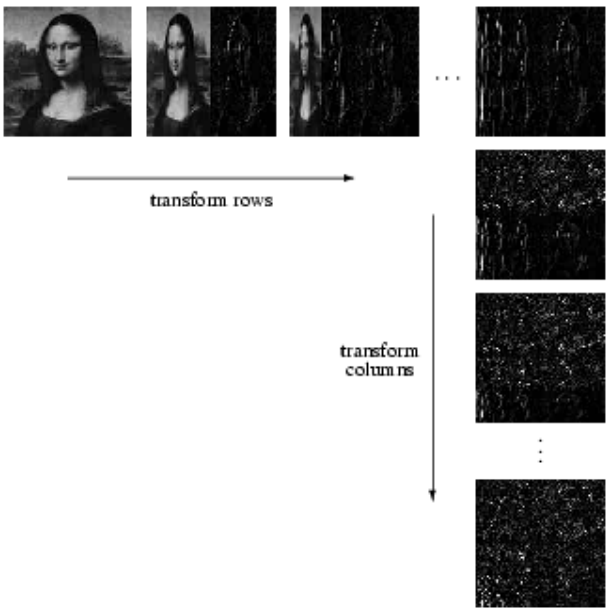

Figure 2.11: Standard Decomposition

- Non-Standard Decomposition:

  operations on rows and columns are alternated (see figure 2.12).

Figures 2.13 and 2.14 represent the two dimensional basis in the same cases.

Figure 2.12: Non-standard Decomposition

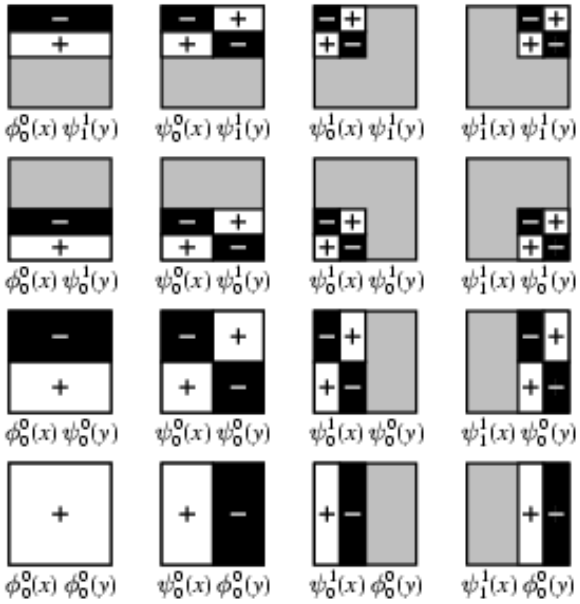

Figure 2.13: Two-dimensional Haar wavelet basis for V^2 Standard Construction

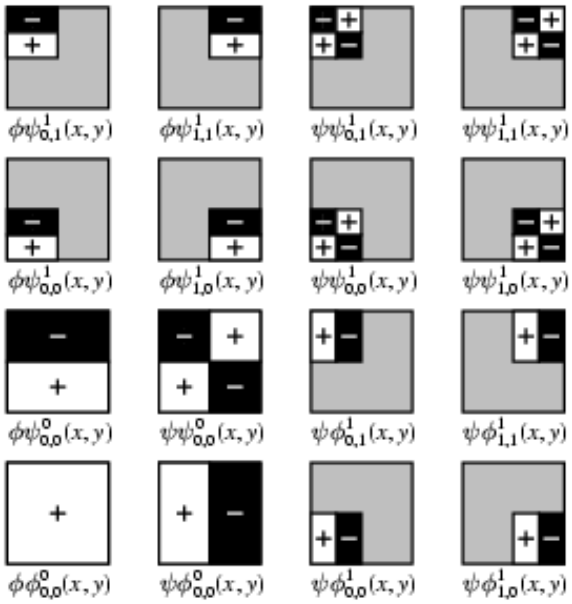

Figure 2.14: Two-dimensional Haar wavelet basis for V^2 Non-standard Construction

Chapter 3

Statistical Learning

Statistical learning is the theoretical and algorithmic backbone of the presented work. In this chapter we set the notation and main concepts used throughout the thesis in Section 3.1, discuss the notion and role of kernel functions in Section 3.2, and illustrate the kernel methods, or learning algorithms, which we used in our work, from the optimization viewpoint in Section 3.3.

3.1 Setting the scene

The problem of learning from examples can be seen as the problem of estimating an unknown functional dependency given only a finite (possibly small) number of instances. The seminal work of Vapnik [Vap98] shows that the key to effectively solve this problem is by controlling the complexity of the solution. In the context of statistical learning this leads to techniques known as *regularization networks* [EPP00] or *regularized kernel methods* [Vap98, CST00, SS02].

3.1.1 Input and output spaces

We consider two sets of random variables $\mathbf{x} \in X \subseteq R^d$ and $y \in Y \subseteq R$ related by a probabilistic relationship. The relationship is probabilistic because generally an element of X does not determine uniquely an element of Y, but rather a probability distribution on Y. This can be formalized assuming that an unknown probability distribution $P(\mathbf{x}, y)$ is defined over the set $X \times Y$. We are provided with *examples* of this probabilistic relationship, that is with a data set $D_\ell \equiv \{(\mathbf{x}_i, y_i) \in X \times Y\}_{i=1}^{\ell}$ called *training set*, obtained by sampling ℓ times the set $X \times Y$ according to $P(\mathbf{x}, y)$.

The "problem of learning" consists in, given the data set D_ℓ, providing an *estimator*, that is a function $f : X \to Y$, that can be used, given any value of $\mathbf{x} \in X$, to predict a value y. For example X could be the set of all possible images, Y the set $\{-1, 1\}$, and $f(\mathbf{x})$ an *indicator function* which specifies whether image $\mathbf{x}$ contains a certain object ($y = 1$), or not ($y = -1$). Another example is the case where $\mathbf{x}$ is a set of parameters, such as pose or facial expressions, y is a motion field relative to a particular reference image of a face, and $f(\mathbf{x})$ is a regression function which maps parameters to motion.

3.1.2 Learning functionals

The standard way to deal with the learning problem consists in defining a *risk functional*, which measures the average amount of error or risk associated with an estimator, and then looking for the estimator with the lowest risk. If $V(y, f(\mathbf{x}))$ is the loss function measuring the error we make when we predict y by $f(\mathbf{x})$, then the average error, the so called *expected risk*, is:

$$I[f] \equiv \int_{X,Y} V(y, f(\mathbf{x}))P(\mathbf{x}, y)\, d\mathbf{x}dy$$

In the classification setting the loss function V usually depends on its arguments through the product $yf(\mathbf{x})$. This particular dependency rests on the implicit assumption that false negatives ($y = +1$ and $f(\mathbf{x}) < 0$) and false positives ($y = -1$ and $f(\mathbf{x}) > 0$) are equivalent. More general situations have been considered in the literature (see for example [LLW02]) leading to describe a loss function as

$$V(y, f(\mathbf{x})) = L(y)V(yf(\mathbf{x})). \tag{3.1}$$

Different loss functions lead to different learning algorithms [EPP00]. Common choices are

- the square loss $V(y, w) = (w - y)^2 = (1 - wy)^2$,
- the hinge loss $V(y, w) = \max\{1 - wy, 0\} =: |1 - wy|_+$,
- the truncated square loss $V(y, w) = \max\{1 - wy, 0\}^2 =: |1 - wy|_+^2$.

Given a training set, a possible way to estimate $I[f]$ is to evaluate the *empirical risk*

$$I_{\text{emp}}^{D_\ell}[f] = \frac{1}{\ell}\sum_{i=1}^{\ell} V(y_i, f(\mathbf{x}_i)).$$

The problem of learning is to find, given the training set, an *estimator* f effectively predicting the label of a new point. This translates in finding a function f such that its expected risk is small with high probability.

Straightforward minimization of the empirical risk is an ill posed problem, since the solution is not unique. A possible way to efficiently solve the learning problem is provided by *regularized kernel methods* which amounts to solving a problem of functional minimization as

$$\min_{f\in\mathcal{H},b\in\mathbb{R}}\{\frac{1}{\ell}\sum_{i=1}^{\ell}V(y_i,f(\mathbf{x}_i)+b)+\lambda(\|f\|^2_{\mathcal{H}}\ +b^c)\} \tag{3.2}$$

where b is an offset term and c a parameter which can take on the values 0, 1, and 2. The second term is a *smoothness* or a *complexity* term measuring the norm of the function f in a suitable Hilbert space $\mathcal{H}$. The minimization takes place in the *hypothesis space* $\mathcal{H}\times\mathcal{B}$.

3.1.3 Hypothesis space and representer theorem

First of all, we recall the definition of reproducing kernel Hilbert space. A RKHS $\mathcal{H}$ on X with kernel $K : X \times X \to \mathbb{R}$ is defined as the unique Hilbert space of real valued functions on X such that, for all $f \in \mathcal{H}$,

$$f(\mathbf{x}) = \langle f, K_{\mathbf{x}}\rangle_{\mathcal{H}} \qquad \forall \mathbf{x} \in X, \tag{3.3}$$

where $K_{\mathbf{x}}$ is the function on X defined by $K_{\mathbf{x}}(\mathbf{s}) = K(\mathbf{x}, \mathbf{s})$.

Regularized kernel methods, or regularization networks [EPP00], are algorithms for minimizing functionals like 3.2 in RKHS. For a comprehensive discussion of the mathematical properties of regularized kernel methods see [DVRC$^+$04]. Here it suffices to say that the solution belongs to a RKHS and, for the representer theorem (and ignoring the offset term), can always be written as

$$f(\mathbf{x}) = \sum_{i=1}^{\ell}\alpha_i K(\mathbf{x}, \mathbf{x}_i).$$

where the coefficient α_i depend on the loss function, on the choice of the kernel and on the regularization parameter λ. Given the form of the solution, it is apparent the relevance of the choice of the kernel function for determining the shape of the solution. For a discussion and a proof of the representer theorem see again [DVRC$^+$04].

3.2 Kernel Functions

We now focus on kernel functions. The use of kernel functions provides a powerful way for detecting nonlinear relations between data, by using linear algorithms in suitable feature space. Regardless of which pattern analyis algorithm is being used, the theoretical properties of a given kernel are the same. This leads to a great flexibility of the approach, since the specifications of the feature space are decoupled from the design of the algorithm.

We are going to present and formalize the characteristics of a kernel function, focusing on the importance of the *kernel matrix*, the matrix containing the evaluation of the kernel function on all pairs of given data points. This matrix, also called the Gram matrix, contains all the information available for the algorithm.

3.2.1 The Gram Matrix

Given a set of vectors $S = \{\mathbf{x}_1, \ldots, \mathbf{x}_\ell\}$ in a vector space X endowed with an *inner product*, the corresponding Gram matrix is defined as the $\ell \times \ell$ matrix $\mathbf{G}$ whose entries are $G_{ij} = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$.

If we are using a kernel function K to evaluate the kernel products in a feature space with feature map ϕ, then the associated Gram matrix has entries:

$$\mathbf{G}_{ij} = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle = K(\mathbf{x}_i, \mathbf{x}_j).$$

In this case the matrix is referred to as *kernel matrix*.

The Gram matrix contains all the information about the data set available to the algorithm. For example, the matrix does not hold any information about the orientation of the original vectors with respect to the origin of the axes, since the matrix of inner products is invariant to rotations about origin. Moreover, the matrix representation loses all the information about any possible alignment between the points and the axes. In essence, all the information available for the pattern analysis algorithm is contained in the kernel matrix and in the labeling information. This reinforces the view that the kernel matrix has a central role in all kernel-based algorithms. It is therefore natural to study the properties of these matrices, the way they are built and how they can be tuned to the task being addressed.

3.2.2 Properties of matrices

The Courant-Fisher theorem characterises the eigenvalues of a matrix, considering the largest eigenvalue given by the Raileigh quotient. It considers maximising or minimising the quotient in a subspace T of specified dimension, and then choosing the subspace either to minimise the maximum or maximise the minimum. The largest eigenvalue corresponds to taking the dimension of T to be that the whole space and then maximising:

Theorem 1 (Courant-Fisher) *If $\mathbf{A} \in \mathbb{R}^{n \times n}$ is symmetric, then for $k = 1, \ldots, n$ the kth eigenvalure $\lambda_k(\mathbf{A})$ of the matrix $\mathbf{A}$ satisfies:*

$$\lambda_k(\mathbf{A}) = \max_{dim(T)=k} \min_{0 \neq \mathbf{v} \in T} \frac{\mathbf{v}'\mathbf{A}\mathbf{v}}{\mathbf{v}'\mathbf{v}} = \min_{dim(T)=n-k+1} \max_{0 \neq \mathbf{v} \in T} \frac{\mathbf{v}'\mathbf{A}\mathbf{v}}{\mathbf{v}'\mathbf{v}},$$

with the extrema achieved by the corresponding eigenvector.

Positive semi-definite matrices A symmetric matrix is *positive semi-definite* if its eigenvalues are all non-negative. By Theorem 1 this holds if and only if:

$$\mathbf{v}'\mathbf{A}\mathbf{v} \geq 0$$

for all vectors $\mathbf{v}$, since the minimal eigenvalue satisfies:

$$\lambda_m(\mathbf{A}) = \min_{0 \neq \mathbf{v} \in \mathbb{R}^n} \frac{\mathbf{v}'\mathbf{A}\mathbf{v}}{\mathbf{v}'\mathbf{v}}.$$

It holds that Gram and kernel matrices are positive semi-definite. Moreover, it holds the:

Proposition 2 *A matrix* $\mathbf{A}$ *is positive semi-definite if and only if* $\mathbf{A} = \mathbf{B}'\mathbf{B}$ *for some real matrix* $\mathbf{B}$

and the:

Proposition 3 *A matrix* $\mathbf{A}$ *is positive semi-definite if and only if all of its principal minors are positive semi-definite.*

3.2.3 Characterizing Kernels

Recall that a kernel function computes the inner product of the images:

$$K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle.$$

As stated before, a matrix formed by evaluating a kernel function over pairs of data is positive semi-definite. Moreover, a kernel function implicitly defines a feature space that does not need to be constructed explicitly. Hence, if we are able to use a proper measure that is able to evaluate the degree of similarity between input points, we can use that as a kernel, verifying if it has all the requirements for being a kernel.

Definition 4 *A function*

$$K : X \times X \to \mathbb{R}$$

satisfies the finitely positive semi-definite property if it is a symmetric function for which the matrices formed by restriction to **any** *finite subset of the space* X *are positive semi-definite.*

3.2.3.1 Mercer's Theorem

In this section we will introduce Mercer's theorem, which fully chatacterizes *kernel functions*.

Theorem 5 (Mercer) *Let X be a compact subset of $\mathbb{R}^n$. Suppse K is a continuous symmetric function such that the integral operator $T_K : L_2(X) \rightarrow L_2(X)$,*

$$(T_K F)(\cdot) = \int_X K(\cdot, f(\mathbf{x}) d\mathbf{x},$$

is positive, that is:

$$\int_{X \times X} K(\mathbf{x}, \mathbf{z}) f(\mathbf{x}, \mathbf{z}) d\mathbf{x} d\mathbf{z} \geq 0$$

for all $f \in L_2(X)$. Then we can expand $K(\mathbf{x}, \mathbf{z})$ in a uniformly convergent series (on $X \times X$) in terms of T_K's eigen-functions $\phi_j \in L_2(X)$, normalised in such a way that $\|\phi_j\|_{L_2} = 1$, and positive associated eigenvalues $\lambda_j \geq 0$,

$$K(\mathbf{x}, \mathbf{z}) = \sum_{j=1}^{\infty} \lambda_j \phi_j(\mathbf{x}) \phi_j(\mathbf{z}).$$

3.2.4 Off-the-Shelf Kernels

Two important examples of *off-the-shelf* kernels are the *polynomial* and the *Gaussian* kernels. Both kernels are computationally efficient (with respect to the *feature mapping*).

Proposition 6 *Let $K_1(\mathbf{x}, \mathbf{z})$ be a kernel over $X \times X$, where $\mathbf{x}, \mathbf{z} \in X$ and $p(x)$ a polynomial with positive coefficients. Then the following functions are also kernels:*

$$K(\mathbf{x}, \mathbf{z}) = p(K_1(\mathbf{x}, \mathbf{z})) \tag{3.4}$$
$$K(\mathbf{x}, \mathbf{z}) = exp(K_1(\mathbf{x}, \mathbf{z})) \tag{3.5}$$
$$K(\mathbf{x}, \mathbf{z}) = exp(-\|\mathbf{x} - \mathbf{z}\|^2/(2\sigma^2)) \tag{3.6}$$

- Polynomial Kernel

  Usually, the definition *polynomial kernel* refers to a special case of 3.5:

  $$K_d(\mathbf{x}, \mathbf{z}) = (\langle \mathbf{x}, \mathbf{z} \rangle + R)^d$$

  defined over a vector space X of dimension n, where R and d are parameters.

- Gaussian Kernel

  The final kernel of proposition 6 is known as *Gaussian kernel*. They are the most widely used kernels for many different tasks with different purposes. For the Gaussian kernel, the images of all points have norm 1 in the resulting feature space as $K(\mathbf{x}, \mathbf{x}) = exp(0) = 1$. The parameter σ controls the "flexibility" of the kernel in a similar way to the degree d in the polynominal kernel. Small values of σ correspond to large values of d since, for example, they allow classifiers to fit any labels, inducing overfitting. In these cases kernel matrix becomes close to the identity matrix. On the other hand, large values of σ gradually reduce the kernel to a constant function, making the learning process impossible in any non-trivial case. The feature space in this case has infinite dimension, so the rank of the kernel matrix will be full anyway, but the points will lie in a low-dimensional subspace.

The kernel contains all the information available to the learning machine about the relative positions of the inputs in the feature space. This implies that if there is a particular structure in the data, then this structure must be enhanced in the kernel matrix. If the kernel function is too general, then it does not give importance to the actual similarity existing between certain input points. This means that the kernel weights the same way any pair of input data, both similar or dissimilar. Therefore, the kernel matrix will be similar to the identity matrix, with many close-to-zero values off the diagonal and values close to 1 in the diagonal. The topic of designing *ad-hoc* functions incorporating the prior knowledge of the problem will be dealt with in Chapter 4.

3.3 Support Vector Methods

We consider two special cases of regularization networks, Support Vector Machines for binary classification and a Support Vector Method for one-class classification.

3.3.1 Binary SVM

In the case of binary classification [Vap95] we have $y_i \in \{-1, 1\}$ for $i = 1, ..., \ell$. The idea is to find a hyperplane in the feature space associated with the kernel K classifying correctly most of the training points. The hyperplane is found as the optimal trade off between making the smallest number of misclassifications and maximizing the *margin*, or the distance of the closest points from the hyperplane. In the primal formulation the problem reduces to minimizing the functional

$$L = \frac{1}{2}\mathbf{w} \cdot \mathbf{w} + C \sum_{i=1}^{\ell} \xi_i - \sum_{i=1}^{\ell} (\alpha_i (y_i \mathbf{w} \cdot \mathbf{x}_i + b) - 1 + \xi_i) - \sum_{i=1}^{\ell} \beta_i \xi_i \tag{3.7}$$

with $\alpha_i \geq 0$, and $\beta_i \geq 0$. Imposing stationarity with respect to $\mathbf{w}$, ξ, and b one obtains

$$\begin{aligned}
\frac{\partial L}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{\ell} y_i \alpha_i \mathbf{x}_i = 0 \\
\frac{\partial L}{\partial \xi_i} &= C - \alpha_i - \beta_i = 0 \\
\frac{\partial L}{\partial b} &= \sum_{i=1}^{\ell} \alpha_i y_i = 0
\end{aligned}$$

Therefore, eliminating b, $\boldsymbol{\xi}$ and $\boldsymbol{\beta}$ in the primal formulation, we can write the dual problem:

$$\begin{aligned}
\max_{\alpha_i} \quad & \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} y_i y_j \alpha_i \alpha_j \mathbf{x}_i \cdot \mathbf{x}_j \\
\text{subject to} \quad & \sum_{i=1}^{\ell} y_i \alpha_i = 0, \\
& 0 \leq \alpha_i \leq C
\end{aligned} \tag{3.8}$$

The solution vector α for each $i = 1, ..., \ell$ satisfies the Kuhn Tuker conditions

$$\alpha_i \left(y_i \mathbf{w} \cdot \mathbf{x}_i + b) - 1 + \xi_i \right) = 0$$

and

$$\xi_i (\alpha_i - C) = 0$$

From these conditions it follows that the solution is found by solving a quadratic programming problem with box constraints (the α_i are always less than C in absolute value) and one equality constraint. The points for which $\alpha_i \neq 0$ are called *support vectors*.

A new point $\mathbf{x}$ is classified according to the sign of the expression

$$\sum_{i=1}^{\ell} y_i \alpha_i \mathbf{x} \cdot \mathbf{x}_i + b, \tag{3.9}$$

where the coefficient b is also determined from the Kuhn Tuker conditions for any $0 < \alpha_i < C$.

In the nonlinear case nothing changes provided that the inner product between input points is viewed as inner product in feature space the value of which is given in terms of the kernel function $K(\mathbf{x}, \mathbf{x}_i)$

3.3.2 One-class SVM

One-class classification techniques were originally developed to cope with binary classification problems in which statistics for one of the two classes was virtually absent ([TD99, SPST$^+$01]). In this setting the component of the training set $(\mathbf{x}_i, y_i)_{i=1}^{\ell}$ labeled according to the minority class ($y = -1$ in the following) is intentionally removed, generating the reduced one-class training set $(\mathbf{x}_i)_{i=1}^{\ell_+}$.

Intuitively the idea behind one-class SVM algorithm is to look for the smallest sphere enclosing the examples in data space. Hence, the training procedure amounts to the solution of the following constrained minimization problem with respect to balls of center $\mathbf{a}$ and radius R in input space

$$\min_{R,\xi_i}\{C\sum_{i=1}^{\ell_+}\xi_i + R^2\}, \tag{3.10}$$

s.t. the existence of a vector $\mathbf{a}$ for which $\forall\ i \leq \ell_+$

$$(\mathbf{x}_i - \mathbf{a}) \cdot (\mathbf{x}_i - \mathbf{a})^\top \leq R^2 + \xi_i, \quad \xi_i \geq 0.$$

A non-linear extension of the previous algorithm can be directly achieved by substituting scalar products with kernel functions. From a more geometrical point of view we consider balls in a suitable feature space. It can be shown (see for example [CS02]) that the centers $\mathbf{a}$ can be mapped one to one with the functions f of the RKHS $\mathcal{H}$ of kernel $K(\mathbf{x}, \mathbf{s}) = \mathbf{x} \cdot \mathbf{s}$. This correspondence is such that $\mathbf{x} \cdot \mathbf{a} = f(\mathbf{x})$, so that the constraints of the problem (3.10) can be rewritten as the existence of a function $f \in \mathcal{H}$ for which $\forall\ i \leq \ell_+$

$$K(\mathbf{x}_i, \mathbf{x}_i) - 2f(\mathbf{x}_i) + \|f\|^2_{\mathcal{H}} \leq R^2 + \xi_i, \quad \xi_i \geq 0.$$

In the minimization problems above the relevant complexity measure R^2 runs over non negative real values. In order to interpret the above problems from a regularization point of view it results convenient slightly modify the problem allowing the penalty to run over unconstrained reals. This is obtained simply by replacing R^2 with the real number ρ. The two problems are essentially equivalent since it is straightforward to verify that whenever $C > \frac{1}{\ell_+}$ they have the same solution. On the other hand if $C < \frac{1}{\ell_+}$ both problems are trivial: for any solution of (3.10) we have $R = 0$ while no solution exists if R^2 is replaced by ρ.

Introducing the offset b and the fixed bias function $\mathcal{D}(\mathbf{x})$, and suitably rescaling the parameter C and the kernel K, problem (3.11) in which R^2 is replaced by ρ becomes

$$\min_{f,b,\xi_i}\{\widetilde{C}\sum_{i=1}^{\ell_+}\xi_i + \frac{1}{2}(\|f\|^2_{\widetilde{\mathcal{H}}} + b)\}, \tag{3.11}$$

s.t. $\forall\ i \leq \ell_+$

$$\mathcal{D}(\mathbf{x}_i) + f(\mathbf{x}_i) + b \geq -\xi_i, \quad \xi_i \geq 0,$$

with

$$\begin{aligned} b &= \frac{1}{2}(\rho - \|f\|^2_{\mathcal{H}} - K(\mathbf{0},\mathbf{0}), \\ \mathcal{D}(\mathbf{x}) &= \frac{1}{2}(K(\mathbf{x},\mathbf{x}) - K(\mathbf{0},\mathbf{0}), \\ \widetilde{C} &= \frac{1}{4}C, \\ \widetilde{K} &= 2K. \end{aligned}$$

Finally, problem (3.11) considering the loss function

$$V(y,w) = \theta(y) \cdot |-wy|_+, \tag{3.12}$$

which matches the general form of 3.1, can be rewritten as

$$\min_{f\in\widetilde{\mathcal{H}}, b\in\mathbb{R}} \{\frac{1}{\ell}\sum_{i=1}^{\ell} V(y_i, \mathcal{D}(\mathbf{x}_i) + f(\mathbf{x}_i) + b) + \lambda(\|f\|^2_{\widetilde{\mathcal{H}}} + b)\}. \tag{3.13}$$

For translation invariant kernel functions (e.g. the Gaussian kernel), the fixed bias function $\mathcal{D}(\mathbf{x})$ vanishes and problem (3.13) fits the general regularization network form (3.2) with $c = 1$.

In the case of novelty detection, described in [TD01], for all training points we have $y_i = 1$ — i.e. all the training examples describe only one class. The intuition is to determine a sphere in the feature space associated with the kernel K containing most of the training points. The solution is a sphere found as the optimal trade off between having the smallest possible radius and allowing for the smallest amount of outliers. In this case the optimization problem reduces to

$$\begin{aligned} \max_{\alpha_i} \quad & \sum_{i=1}^{\ell} \alpha_i K(\mathbf{x}_i,\mathbf{x}_i) - \sum_{i=1}^{\ell}\sum_{j=1}^{\ell} \alpha_i\alpha_j K(\mathbf{x}_i,\mathbf{x}_j) \\ \text{subject to} \quad & \sum_{i=1}^{\ell} \alpha_i = 1 \\ & 0 \leq \alpha_i \leq C. \end{aligned} \tag{3.14}$$

If $K(\mathbf{x},\mathbf{x})$ is constant over the domain X, a novelty is detected if the inequality

$$\sum_{i=1}^{\ell} \alpha_i K(\mathbf{x},\mathbf{x}_i) \geq \tau \tag{3.15}$$

is violated, where $\tau > 0$ is determined from the solution of problem (3.14).

Chapter 4

Kernel Engineering

A main issue of statistical learning approaches like Support Vector Machines or other kernel methods is which kernel to use for which problem. In Chapter 3 we have described a number of general purpose kernels but, especially in the case in which only a relatively small number of examples is available, there is little doubt that the use of an appropriate kernel function can lead to a substantial increase in the generalization ability of the developed learning system.

Building kernel functions presents two difficulties. The first is that finding an appropriate kernel function might be complex owing to the mathematical requirements a function needs to satisfy in order to be safely used as a kernel. The second is that a specific kernel should be motivated in terms of prior knowledge of the application domain.

In this chapter we start discussing kernels based on the manipulation of binary vectors [FOV05]. Since the information content of images and other type of signals can be represented with binary vectors, the presented analysis is relevant to a number of application domains.

We show that two kernels derived from suitable manipulations of binary vectors are Mercer's kernels. Both these kernels are motivated by computer vision considerations: the first, histogram intersection, has been successfully used as a tool for object recognition while the second is obained by the notion of Hausdorff distance between two image patches [FOA01].

4.1 Ad Hoc Kernels

As we pointed out in Chapter 3, kernels have a central role in the effectiveness of the learning process. When data are images the input space is often rather big, while the

number of examples tends to be relatively small, and this translates in the need of using as much of the prior knowledge available in order to find appropriate similarity measures. The topic of kernels for images is relatively new [STC04], but has been studied quite extensively in the last few years.

At first, number of studies have been reported about the use of general purpose kernels for image-based classification problems [PV97, CP, GLK00, JMKL00, HHP01, MPP01]. Then, one of the first attempts to introduce prior knowledge on the correlation properties of images in the design of an image kernel can be found in [SSSV98]; in [CHV99], instead, a family of functions which seem to be better suited than Gaussian kernels for dealing with image histograms has been studied.

In essence the kernels described in [CHV99] ensure heavier tails than Gaussian kernels in the attempt to contrast the well known phenomenon of diagonally dominant kernel matrices in the case of high dimensional inputs. In [JLZZ03] a RBF kernel is engineered in order to use the Earth Mover's Distance as a dissimilarity measure.

Recently a new family of kernels, referred to as *string kernels*, has been proposed. They are engineered for dealing with a string data. These kernels consider the degree of presence of a certain subsequence of elements to ba a function of how many gaps are interspersed within it. two different strings are weighted similar in mean of how many similar subsequences they contain. A complete survey of these kernels can be found in [STC04].

Fisher kernels [JH98, JH99] attempt to analiyze more information from a single generative model than the only output probability. This means that they try to investigate on the internal representation on data items within the model.

4.2 Kernel for Binary Strings

In this section we deal with the topic of finding Mercer's kernel for binary strings — both bitwise operations, and measures capturing local spatial correlation are considered. In sections 4.3 and 4.4 we will see how this will lead to image kernels. We start by setting the notation.

4.2.1 Notation

Throughout the following we denote strings with upper case letters like A and B and bits with lower case letters like a and b. A binary string of fixed length is a sequence of bits like 0010 or $a_1a_2...a_P$ with $a_p \in \{0, 1\}$ for $p = 1, ..., P$. A string A of length P can also be written as a P-dimensional vector $\mathbf{A} = (a_1, ..., a_P)$. If $\mathbf{A}$ is an ordinary P-dimensional

vector we write $\mathbf{A} = (A_1, ..., A_P)$. Without risk of confusion we do not distinguish between the *truth* values "0" and "1" and the *numerical* values "0" and "1". Finally, we write $\mathbf{A} \cdot \mathbf{B}$ for the standard inner product between $\mathbf{A}$ and $\mathbf{B}$.

4.2.2 Making kernels with logical connectives

We first deal with the case in which all binary strings have the same length and the similarity between strings is computed bitwise. As we will see in the next section this case is relevant, for example, to histogram-like representations.

The question we address is which two-place logical operator acting bitwise on pairs of binary strings A and B defines a Mercer's kernel. Given the four different bit pairs – $(0,0)$, $(0,1)$, $(1,0)$, and $(1,1)$ – we have only to consider 2^4 different logical operators. Following standard notation we use $\wedge$, $\vee$, and $\leftrightarrow$ for the *conjunction*, *disjunction*, and *if-and-only-if* two-place logical operators, and $\neg$ for the *negate* one-place operator. We start with the conjunction, the AND operator, and define

$$K_{\wedge}(A, B) = \sum_{p=1}^{P} a_p \wedge b_p. \tag{4.1}$$

It is easy to see that $K_{\wedge}$ is the linear kernel acting on the vectors $\mathbf{A}$ and $\mathbf{B}$ since

$$\sum_{p=1}^{P} a_p \wedge b_p = \sum_{p=1}^{P} a_p b_p = \mathbf{A} \cdot \mathbf{B}.$$

Of the remaining 15 cases only the negate of the disjunction, the NOR operator, and the if-and-only-if operator define Mercer's kernels. Indeed, using the identity

$$\neg (a_p \vee b_p) \equiv (\neg a_p) \wedge (\neg b_p)$$

we find that

$$K_{\neg\vee}(A, B) = \sum_{p=1}^{P} \neg (a_p \vee b_p) = \sum_{p=1}^{P} (1 - a_p)(1 - b_p) = (\neg \mathbf{A}) \cdot (\neg \mathbf{B}),$$

from which we see that $K_{\neg\vee}$ is the standard inner product between the binary vectors obtained by negating each component of the original vectors $\mathbf{A}$ and $\mathbf{B}$ respectively.

From the fact that the sum of kernels is a kernel, instead, we find that

$$K_{\leftrightarrow}(A, B) = \sum_{p=1}^{P} a_p \leftrightarrow b_p$$

is a Mercer's kernel since

$$K_{\leftrightarrow}(A,B) = K_{\wedge}(A,B) + K_{\neg\vee}(A,B).$$

Through simple counterexamples it can be immediately verified that the other 13 two-place logical operators do not define Mercer's kernels. Let us consider for example $\otimes$, the *or exclusive* operator:

$$K_{\otimes}(A,B) = \sum_{p=1}^{P} a_p \otimes b_p$$

If we consider the two binary strings $\mathbf{A}^1 = (1,0)$ and $\mathbf{A}^2 = (0,1)$, and compute explicitly the entries of the 2×2 matrix K, $K_{ij} = \mathbf{A}^i \otimes \mathbf{A}^j$, we obtain

$$K = \begin{bmatrix} 0 & 2 \\ 2 & 0 \end{bmatrix}$$

which is clearly not positive semidefinite. Notice that this operator corresponds to the computation of the L^1 distance between binary strings since

$$\mathbf{A} \otimes \mathbf{B} = \sum_{p=1}^{P} |a_p - b_p| = \|\mathbf{A} - \mathbf{B}\|_{L^1}$$

To conclude, if we restrict our attention to the case in which the binary strings have the same number N of bits equal to 1, it is easy to show that for each possible pairs of strings $\mathbf{A}$ and $\mathbf{B}$ we have

$$2\mathbf{A} \cdot \mathbf{B} + \mathbf{A} \otimes \mathbf{B} \equiv 2N. \tag{4.2}$$

Identity (4.2) follows trivially from the fact that for two strings of length P with N 1s each, $\hat{N}$ of which at the same location, we have $\mathbf{A} \cdot \mathbf{B} = \hat{N}$ and $\mathbf{A} \otimes \mathbf{B} = 2(N - \hat{N})$.

4.2.3 Capturing Spatial Correlations

Up to now we have discussed operators acting bitwise on binary strings. If nearby bits in a string are correlated we aim at finding Mercer's kernels able to capture short range correlations. These kernels are expected to be particularly useful when studying images and signals in the original domain.

The key idea is to replace the binary vector $\mathbf{A}$ with the non-binary vector $\mathbf{A}^S$. The vector $\mathbf{A}^S$ can be thought of as the result of convolving $\mathbf{A}$ with the stencil $\mathbf{S}$,

$$\mathbf{A}^S = \mathbf{A} * \mathbf{S}.$$

For example, if $\mathbf{S} = (1, 1, 1)$ we have $A_p^S = \sum_{i=-1}^{1} a_{p+i}$, where, for simplicity, we considered the last component, a_P, adjacent to the first, a_1 (equivalently, all summations on subscripts are taken modulo P). The vector $\mathbf{A}^S$ makes explicit the existence of short range spatial correlations between the components of $\mathbf{A}$ nearby a non zero component, while the convolution operation "$*$" can be interpreted as a feature mapping. We thus have a class of Mercer's kernels indexed by the possible stencils and defined as

$$K_S(A, B) = \mathbf{A}^S \cdot \mathbf{B}^S.$$

Extending the convolution operation with the stencil $\mathbf{S}$ to non binary vectors $\mathbf{V}$ in the natural way, we now show that for a specific class of stencils one can identify a particular subset of Mercer's kernels. These kernels enjoy an interesting property which will be useful in Section 4.3.

Theorem: Let $\mathbf{S}$ be defined as

$$\mathbf{S} = \left(\underbrace{1/(2m), ..., 1/(2m)}_{m}, 1, \underbrace{1/(2m), ..., 1/(2m)}_{m} \right), \tag{4.3}$$

then there exists a feature mapping $\phi : \mathbb{R}^P \to \mathbb{R}^{2m \times P}$ defined as $\mathbf{W} = \phi(\mathbf{V})$ such that for all vectors $\mathbf{V} \in \mathbb{R}^P$

$$\mathbf{V}^S \cdot \mathbf{V} = \mathbf{W} \cdot \mathbf{W}.$$

Proof: Using the stencil $\mathbf{S}$ of (4.3), for the components $p = 1, \ldots, P$ of $\mathbf{V}^S$ we find

$$V_p^S = V_p + \frac{1}{2m} \sum_{\substack{i=-m \\ i \neq 0}}^{m} V_{p+i}.$$

Taking the inner product between $\mathbf{V}^S$ and $\mathbf{V}$ gives

$$\mathbf{V}^S \cdot \mathbf{V} = \frac{1}{4m} \left(4m \sum_{p=1}^{P} V_p^2 + 2 \sum_{p=1}^{P} \sum_{\substack{i=-m \\ i \neq 0}}^{m} V_p V_{p+i} \right).$$

Expanding and rearranging the terms of the above sums we obtain

$$\begin{aligned} 4m\mathbf{V}^S \cdot \mathbf{V} &= \underbrace{V_1^2 + ... + V_1^2}_{4m} + ... + \underbrace{V_P^2 + ... + V_P^2}_{4m} \\ &+ 2V_1(\underbrace{V_{1-m} + ... + V_{1+m}}_{2m}) + ... + 2V_P(\underbrace{V_{P-m} + ... + V_{P+m}}_{2m}). \end{aligned}$$

Finally, grouping the squares and the corresponding mix products gives

$$\mathbf{V}^S \cdot \mathbf{V} = \frac{1}{4m} \left(\sum_{\substack{i=-m \\ i\neq 0}}^{m} (V_1 + V_{1+i})^2 + ... + \sum_{\substack{i=-m \\ i\neq 0}}^{m} (V_P + V_{P+i})^2 \right). \tag{4.4}$$

Notice that $\mathbf{V}^S \cdot \mathbf{V}$ is the sum of $2m \times P$ squares and can thus be regarded as the square of the norm of a $2m \times P$-dimensional vector $\mathbf{W}$ the components of which are of the type

$$\frac{1}{2\sqrt{m}}(V_p + V_{p+i})$$

for $p = 1, \ldots P$ and $i = -m, \ldots, m$ $(i \neq 0)$.

Moreover, notice that this stencil is just one of many. It is not important that the stencil has this shape, but that verifies requirements in order to build a Mercer's kernel.

Simple counting arguments can be advocated to extend this theorem to other stencils subject to the condition that the central entry is 1 and the remaining $\mathcal{M}$ entries w_i can be written as $w_i = n_i/\mathcal{N}$ with $\mathcal{N} = \sum_{j=1}^{\mathcal{M}} n_j$. For the specific stencil $\mathbf{S}$ in the theorem every component appears twice in the r.h.s. of (4.4) and the mapping could be more parsimoniously defined in a feature space of dimensionality $m \times P$.

We are now in a position to study under what conditions a number of similarity measures, well known to the computer vision community, can be used as kernels in statistical learning methods. In the following section we show that, by means of appropriate modifications, a similarity measure for grey level images based on the notion of Hausdorff distance is a Mercer's kernel.

4.3 Kernel based on the Hausdorff distance

The basis of our methodology is a measure based on the notion on Hausdorff distance which is a distance measure between two arbitrary point sets.

The directed Hausdorff distance between two (finite) point sets A and B is defined as:

$$h(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\|.$$

$h(A, B)$, will be small when every point of A is near some point of B. The directed Hausdorff distance is not symmetric. To restore symmetry, for instance, one could take the maximum or the averag between $h(A, B)$ and $h(B, A)$. This leads to the Hausdorff distance.

This measure has been extensively used to compare binary signals, such as binary maps. More recently a modification designed to deal with gray-level images has been proposed. This modification is the starting point for the kernel for images that we will describe in the remainder of the section.

4.3.1 Matching grey-level images with Hausdorff distance

Here we introduce a method for measuring the degree of similarity between two gray-level images. This technique is inspired by the notion of Hausdorff distance and can be interpreted as a grey level equivalent to a matching method for the binary case developed in [HKR93].

Let us start describing the essence of the similarity measure proposed in [FOA01]. Without loss of generality and for ease of notation we consider the case of 1-D images.

Suppose we have two N pixel grey-level images, $\mathcal{A}$ and $\mathcal{B}$, of which we want to compute the degree of similarity. In order to compensate for small level changes or local transformations we define a neighborhood O of each pixel $i = 1, ..., N$ and evaluate the expression

$$h_{\mathcal{A}}(\mathcal{B}) = \sum_{i=1}^{N} U(\epsilon - |\mathcal{A}[i] - \mathcal{B}[s_i]|) \tag{4.5}$$

where U is the unit step function, and s_i denotes the pixel of $\mathcal{B}$ most similar to $\mathcal{A}[i]$ in the neighborhood $O(i)$ of i. For each pixel i the corresponding term in the sum (4.5) equals 1 if $|\mathcal{A}[i] - \mathcal{B}[s_i]| \leq \epsilon$ (that is, if in the neighborhood $O(i)$ there exists a pixel s_i in which $\mathcal{B}$ differs by no more than ϵ from $\mathcal{A}[i]$) and 0 otherwise. The rationale behind (4.5) is to evaluate the degree of overlap of two images bypassing pixel-wise correspondences that do not take into account the effects of small variations and acquisition noise.

Unless the set O consists of the only element i, $h_{\mathcal{A}}(\mathcal{B}) \neq h_{\mathcal{B}}(\mathcal{A})$. Symmetry can be restored, for example, by taking the average

$$H(\mathcal{A}, \mathcal{B}) = \frac{1}{2} \left(h_{\mathcal{A}}(\mathcal{B}) + h_{\mathcal{B}}(\mathcal{A}) \right).$$

The quantity $H(\mathcal{A}, \mathcal{B})$ can be computed in three steps:

1. Expand the two images $\mathcal{A}$ and $\mathcal{B}$ into 2-D binary matrices A and B respectively, the second dimension being the grey value. For example, for A we write

$$A[i, j] = \begin{cases} 1 & \text{if } \mathcal{A}[i] = j; \\ 0 & \text{otherwise.} \end{cases}$$

2. Dilate both matrices by growing their nonzero entries by a fixed amount $\epsilon/2$ in the grey value dimension and half the linear size of the neighborhood O in the spatial dimension. Let $\tilde{A}$ and $\tilde{B}$ be the resulting 2-D dilated binary matrices.

3. To obtain $H(\mathcal{A}, \mathcal{B})$ compute the size of the intersections between A and $\tilde{B}$, and B and $\tilde{A}$, and take the average of the two values. Thinking of the 2-D binary matrices as (binary) vectors in obvious notation we could also write

$$H(\mathcal{A}, \mathcal{B}) = \frac{1}{2}\left(\mathbf{A} \cdot \tilde{\mathbf{B}} + \mathbf{B} \cdot \tilde{\mathbf{A}}\right). \tag{4.6}$$

Under appropriate conditions on the dilation sizes h is closely related to the partial directed Hausdorff distance between the binary matrices A and B thought of as 2-D point sets [HKR93, FOA01]. As a similarity measure, the function H in (4.6) has been shown to have several interesting properties, like tolerance to small changes affecting images due to geometric transformations, viewing conditions, and noise, and robustness to occlusions [FOA01].

4.3.2 Is it a Mercer's kernel?

In this section we ask whether, or not, the similarity measure of above can be adopted as a kernel in statistical learning methods.

For a fixed training set of images, $\{\mathcal{A}^1, ..., \mathcal{A}^\ell\}$, an empirical answer to this question can be simply provided by checking the positive semidefiniteness of the $\ell \times \ell$ matrix

$$K_{ij} = H(\mathcal{A}^i, \mathcal{A}^j) = \frac{1}{2}\left(\mathbf{A}^i \cdot \tilde{\mathbf{A}}^j + \mathbf{A}^j \cdot \tilde{\mathbf{A}}^i\right).$$

In [BOV02] we show experimental results indicating that in many practical applications this appears to be the case. By contrast, a simple counterexample shows that $H(\mathcal{A}, \mathcal{B})$ is not a Mercer's kernel. Consider the three 1-pixel images $\mathcal{A}^1 = 1$, $\mathcal{A}^2 = 2$, $\mathcal{A}^3 = 3$, and $\epsilon = 1$. Explicit computation of the entries of the 3×3 matrix K gives

$$K = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix},$$

which, having negative determinant, is clearly not positive semidefinite. As a rule of thumb, we found that if the dilation size is small compared to the image size (a few units against a few hundreds) the matrix K_{ij} is very likely to be positive semidefinite.

Notice that H is not a Mercer's kernel since the dilation stage makes the off diagonal elements of the "kernel" matrix too large. Using the theorem of Section 4.2.3, we see that

if we redefine the dilation stage as a convolution with a stencil of the type in expression (4.3), the newly defined quantity H_s,

$$H_s(\mathbf{A}, \mathbf{B}) = 1/2(\mathbf{A}^s \cdot \mathbf{B} + \mathbf{B}^s \cdot \mathbf{A}),$$

is an inner product. Symmetry and linearity hold trivially, while positivity for all *possible* vectors $\mathbf{V}$ follows from the theorem. It is interesting to remark that this result is made possible by the *reduced amount* of dilation built in in the convolution stage as opposed to the original dilation stage.

4.4 Exploiting global representations

In this section we discuss a number of similarity measures based on histograms and show under what conditions they can be employed as kernels for statistical learning. In particular we draw our attention toward histogram intersection, a technique made popular in [SB91] for color indexing with application to object recognition.
As we have seen in Chapter 2, histograms provide a crude but very effective basic representation of images. In the case of color images, histograms are very useful for addressing difficult problems like 3D object recognition and classification. Depending on the task, different color spaces (RGB, HSV, HSI, etc.) might turn out to be more appropriate. From the results reported in [SB91] and subsequent works, for example, we know that color histogram intersection is an effective similarity measure which makes it possible to build reasonably efficient color-based recognition systems.

4.4.1 Histogram intersection

We start off by defining histogram intersection in the simpler case of 1-D images of N pixels. We denote with A and B the M-bin histograms of images $\mathcal{A}$ and $\mathcal{B}$ respectively and define the *histogram intersection*, H_{int}, as

$$H_{int}(A, B) = \sum_{m=1}^{M} \min\{A_m, B_m\} \tag{4.7}$$

with A_m and B_m the m-th bin for $m = 1, ..., M$ and $\sum_{m=1}^{M} A_m = \sum_{m=1}^{M} B_m = N$. Intuitively, histogram intersection measures the degree of similarity between two histograms and we clearly have $0 \leq H_{int} \leq N$. Unlike other similarity measures, like L^2 distance for example, this similarity measure is not only well suited to deal with color and scale changes, but can also be successfully used in the presence of non-segmented objects. Furthermore, H_{int} can be computed efficiently and adapted to search for partially occluded objects in images.

4.4.2 Is it a Mercer's kernel?

The effectiveness of histogram intersection as a similarity measure for color indexing raises the question of whether, or not, this measure can be adopted in kernel based methods. We answer this question by finding explicitly the feature mapping after which histogram intersection is an inner product.

If we represent an histogram A with the P-dimensional binary vector $\mathbf{A}$ (with $P = N \times M$) defined as

$$\mathbf{A} = \left(\underbrace{\overbrace{1,1,...,1}^{A_1}\overbrace{0,...,0}^{N-A_1}}_{\text{1-st bin}}, \underbrace{\overbrace{1,1,...,1}^{A_2}\overbrace{0,0,...,0}^{N-A_2}}_{\text{2-nd bin}}, ..., \underbrace{\overbrace{1,1,...,1}^{A_M}\overbrace{0,0,...,0}^{N-A_M}}_{\text{M-th bin}} \right), \tag{4.8}$$

and similarly B with $\mathbf{B}$, it can be readily seen that $H_{int}(A, B)$ in (4.7) is equal to the standard inner product between the two corresponding vectors $\mathbf{A}$ and $\mathbf{B}$, or in the notation of the previous section

$$H_{int}(A, B) = K_{\wedge}(A, B).$$

We thus have that H_{int} is a Mercer's kernel and that the binary vector in (4.8) describes explicitly the mapping $\mathbf{A} = \phi(A)$ between input and feature space.

A few comments are in order. First we notice the redundant representation of the information content in (4.8). The P components of $\mathbf{A}$ are dependent *features* since the original histogram A is uniquely determined by the rightmost 1 in each of the M N-tuplets of components in (4.8).

Second, the assumption of dealing with images with the same number of pixels is unnecessary. All what we just said holds true for images with different number of pixels provided that P is taken large enough (for example equal to $\hat{N} \times M$ with $\hat{N}$ equal to the number of pixels of the largest image in the considered set. If needed, in the case of images of different size one can normalize the histogram areas, and repeat the same construction of above on the resulting normalized histograms.

Third, the generalization of this result to higher dimension like 2-D images and 3-D color space representations is straightforward.

In [BOV03b] we report experimental results showing that the histogram intersection kernel (which is parameter free) leads to the best recognition rates in an indoor/outdoor classification problem when compared with other kernels.

Note that histogram intersection kernel can be seen as the L^1 distance between *cumulative distribution function* evaluated over the two vectors:

$$H_{int}(A, B) = \| cdf(\mathbf{A}) - cdf(\mathbf{B}) \|_{L^1}.$$

We conclude observing that, as noted in [SB91], if all images have the same number N of pixels, histogram intersection can be written in terms of the L^1 bin-wise distance. Using (4.2) and the fact that the L^1 distance in input space is equal to the L^1 bit-wise distance in feature space, $\sum_{p=1}^{P} |a_p - b_p| = \sum_{m=1}^{M} |A_m - B_m|$, we have

$$2H_{int}(A, B) = 2N - \|\mathbf{A} - \mathbf{B}\|_{L^1}.$$

Chapter 5

Image Classification

In this chapter we are going to show the experiments performed on some tyipical image classification problems. Most of the images used in our experiments have been downloaded from the web, by randomly crawling websites containing images relevant to the problems of above. A small percentage is composed by holiday photos of members of our research group. The images used have different sizes and resolutions. While performing the experiments we noticed that some images had been artificially retouched and we descarded them from our dataset.

5.1 Indoor/Outdoor Classification

Deciding whether an image is an indoor or an outdoor view is a typical example of understanding the"semantics" of the image content [SP98]. Its practical use ranges from scene understanding, to automatic image adjustment and intelligent film development.

An important cue for this problem is color distribution, as reported in [SP98].

We choose this application as a case study for the effectiveness of histogram intersection kernel as opposed to general purpose ones.

For the sake of comparison, we have selected a few experiments and used the RGB space instead of the HSV space, while keeping the other conditions identical: the impact of the choice of the color space on performance was found to be minimal compared to the impacts of the other experimental conditions (choice of the kernel, remapping of the input). An explanation for this fact is that, after quantization into bins, no information about the color space is used by the classifier.

Subsampling the image too much results in significant losses in performance. This may be

Figure 5.1: Samples of indoor images used in our experiments.

explained by the fact that by subsampling, the histogram loses its sharp peaks, as pixel colors turn into averages (aliasing).

The problem can be naturally seen as a binary classification problem; we therefore use a Support Vector Machine for binary classification.

For our experiments we used a collection of 1392 outdoor images and 681 indoor images; the two sets are unbalanced since indoor images are more difficult to collect and are often either too dark or deteriorated by saturation effects produced by artificial illumination.

Figures 5.1 and 5.2 show examples of indoor and outdoor images respectively, taken from our dataset, while figure 5.3 shows examples of ambiguous images, either indoor images with natural outdoor light effects (often caused by the presence of windows), outdoor images taken at the sunset, or close-ups.

Figure 5.2: Samples of outdoor images used in our experiments.

Figure 5.3: Example of images with ambiguous labeling.

The images of the dataset are not homogeneous in size, format, or acquisition device. Their typical sizes are in the range of $10^4 \div 10^5$ pixels.

We randomly selected 4 different sets of training and test data from the original dataset,and each time we trained an SVM for binary classification on the training sets. The dimension of all training sets are 400 images for the indoor group and 400 images for the outdoor

Kernel type	**Recognition rate** (%)		
	$8 \times 8 \times 8$	$15 \times 15 \times 15$	$20 \times 20 \times 20$
histogram intersection	69.63 ± 3.1	**89.73** ± 0.5	71.98 ± 5.4
linear	70.02 ± 4.2	82.71 ± 2.2	68.15 ± 8.4
2-nd deg polynomial	69.90 ± 2.9	82.83± 1.9	68.65 ± 7.7
4-th deg polynomial	67.17 ± 3.1	81.88 ± 2.1	69.00 ± 7.2
Gaussian RBF kernel ($\sigma = 0.3$)	66.52 ± 2.6	81.40± 1.6	66.64 ± 6.2
Gaussian RBF kernel ($\sigma = 0.5$)	67.35 ± 3.0	81.98 ± 1.9	68.08 ± 7.2
Gaussian RBF kernel ($\sigma = 0.7$)	69.08 ± 3.1	83.51 ± 2.1	68.94 ± 7.7
Gaussian RBF kernel ($\sigma = 0.9$)	69.03 ± 3.5	83.16 ± 2.4	68.61 ± 8.3

Table 5.1: Recognition rates (*r.r.*) for SVMs with different kernels on the indoor-outdoor classification problem in the HSV color space.

Kernel type	**Recognition rate** (%)		
	$8 \times 8 \times 8$	$15 \times 15 \times 15$	$20 \times 20 \times 20$
histogram intersection	58.69 ± 4.1	**88.44** ± 0.7	88.71 ± 0.8
linear	55.61 ± 2.1	82.57 ± 1.6	82.48 ± 2.0
2-nd deg polynomial	53.30 ± 1.5	83.06 ± 1.6	83.56 ± 1.6
4-th deg polynomial	50.46 ± 2.6	84.26 ± 1.2	84.48 ± 1.2
Gaussian RBF kernel ($\sigma = 0.3$)	46.30 ± 2.0	82.24 ± 1.0	84.44 ± 1.1
Gaussian RBF kernel ($\sigma = 0.5$)	50.77 ± 1.8	85.82 ± 2.0	84.34 ± 1.1
Gaussian RBF kernel ($\sigma = 0.7$)	54.05 ± 1.2	85.21 ± 0.8	84.40 ± 1.5
Gaussian RBF kernel ($\sigma = 0.9$)	53.96 ± 1.8	80.13 ± 0.2	83.67 ± 1.8

Table 5.2: Recognition rates (*r.r.*) for SVMs with different kernels on the indoor-outdoor classification problem in the RGB color space.

group. The validation sets contain at each random extraction the images excluded from the training sets. The dimensions of the validation sets is 992 outdoor images and 281 indoor images.

Each image is represented by means of a single global color histogram. We worked both on the RGB and on the HSV space and perform our experiments on different samplings of the color spaces. The experiments reported in this section have been obtained using $8 \times 8 \times 8$, $15 \times 15 \times 15$, $20 \times 20 \times 20$ bins (this corresponds to work in spaces of 512, 3375, and 8000 dimensions respectively).

We compared the performance obtained using histogram intersection with the ones of common off-the-shelf kernels (see the average recognition rated in Tables 5.1 and 5.2 and

Kernel type	**Recognition rate** (%) $4 \times 15 \times 15$
histogram intersection	**87.06** $\pm$ 0.8
linear	81.31 $\pm$ 1.5
2-nd deg polynomial	81.38 $\pm$ 1.3
4-th deg polynomial	81.63 $\pm$ 1.4
Gaussian RBF kernel ($\sigma = 0.3$)	81.65 $\pm$ 0.9
Gaussian RBF kernel ($\sigma = 0.5$)	81.84 $\pm$ 1.3
Gaussian RBF kernel ($\sigma = 0.7$)	82.01 $\pm$ 1.4
Gaussian RBF kernel ($\sigma = 0.9$)	81.91 $\pm$ 1.4

Table 5.3: Results obtained for a $4 \times 15 \times 15$ sampling.

Kernel type	**Recognition rate** (%)		
	$8 \times 8 \times 8$	$15 \times 15 \times 15$	$20 \times 20 \times 20$
Histogram intersection	69.63 $\pm$ 3.1	89.73 $\pm$ 0.5	71.98 $\pm$ 5.4
Laplacian RBF kernel ($\sigma = 1.0$)	71.23 $\pm$ 3.6	**90.75** $\pm$ 0.6	73.24 $\pm$ 6.5
Laplacian RBF kernel ($\sigma = 0.9$)	70.61 $\pm$ 2.8	90.00 $\pm$ 0.7	72.60 $\pm$ 5.9
Sublinear RBF kernel ($\sigma = 30.0$)	71.04 $\pm$ 3.4	90.19 $\pm$ 0.3	75.37 $\pm$ 5.4
Sublinear RBF kernel ($\sigma = 50.0$)	73.51 $\pm$ 5.0	89.67 $\pm$ 0.5	72.99 $\pm$ 6.0
Sublinear RBF kernel ($\sigma = 70.0$)	72.61 $\pm$ 5.6	89.62 $\pm$ 0.3	72.86 $\pm$ 6.0

Table 5.4: Recognition rates (*r.r.*) for SVMs with heavy-tailed kernels in the HSV color space. Laplacian RBF corresponds to the choice of a=1 and b=1 , sublinear RBF are such that a=1 and b=0.5.

the comments further down the section). We also made accurate comparisons with the heavy-tailed RBF kernels presented in [CHV99] (see Tables 5.4 and 5.5).

During the training stage, for all kernels we obtain a perfect separation a number of support vectors in the range of $42\% \div 55\%$ of the training data (see Table 5.6 for a illustrative list of the number of support vector); the number of support vectors for the HSV representation is always slightly higher.

The choice of the color space does not seem to be crucial (in the experiments presented HSV performs only slightly better than RGB); given a color space the choice of the bin sampling is more important, as reported in [CHV99]. In HSV a sampling of about 15 bins per channel is a good compromise between efficiency and computational cost. In RGB finer samplings lead to higher recognition rates.

Kernel type	**Recognition rate** (%)		
	$8 \times 8 \times 8$	$15 \times 15 \times 15$	$20 \times 20 \times 20$
Histogram intersection	58.69 ± 4.1	88.44 ± 0.7	88.71 ± 0.8
Laplacian RBF kernel ($\sigma = 1.0$)	49.09 ± 2.0	**90.35** ± 0.7	89.76 ± 0.9
Laplacian RBF kernel ($\sigma = 0.8$)	46.39 ± 2.8	89.67 ± 0.6	89.53 ± 0.9
Sublinear RBF kernel ($\sigma = 30.0$)	48.60 ± 1.4	89.67 ± 0.6	89.21 ± 0.6
Sublinear RBF kernel ($\sigma = 50.0$)	52.09 ± 2.4	88.84 ± 0.6	88.94 ± 0.6
Sublinear RBF kernel ($\sigma = 70.0$)	52.67 ± 3.4	89.19 ± 1.5	88.67 ± 0.6

Table 5.5: Recognition rates (*r.r.*) for SVMs with heavy-tailed kernels in the RGB color space. Laplacian RBF corresponds to the choice of a=1 and b=1 , sublinear RBF are such that a=1 and b=0.5.

Kernel type	**RGB**	HSV
histogram int.	436	532
linear	372	419
polynomial (deg. 2)	355	395
polynomial (deg. 4)	342	370
Gaussian ($\sigma = 0.3$)	337	378
Gaussian ($\sigma = 0.5$)	341	379
Gaussian ($\sigma = 0.7$)	364	396
Gaussian ($\sigma = 0.9$)	369	416

Table 5.6: The number of support vectors obtained for diffent kernels, a training set of 800 items, two different color spaces sampled with 15 bins for each color field.

In the case of HSV we also performed experiments with an unbalanced sampling (different number of bins per channel). In the case of indoor/outdoor classification the quantity of light and saturation seemed to be more determinant that the shade of colors; for this reason we performed a few experiments with a lower sampling on the hue. Table 5.3 shows results obtained for a $4 \times 15 \times 15$ sampling. The recognition rates drop consistently of about 1% for all kernels with respect to the $15 \times 15 \times 15$ case, but this choice is an interesting compromise that allow us to achieve still good results (compared to the $8 \times 8 \times 8$ sampling) in a relatively small input space (900-dimensions as opposed to the 3375 of the balanced 15 bins sampling).

Histogram intersection performs on average worse than heavy-tailed RBF kernels with carefully selected parameters.

Interestingly, the choice of a and b that produced the best results for image categorization

in [CHV99] (i.e., $a = 0.25$, $b = 1$) does not give good results on the indoor-outdoor problem of our database (the recognition rate obtained is 85.2%). An important interesting point, though, is that histogram intersection always performs better than general purpose kernels, and performs comparably to heavy-tailed kernels, but does not require the tuning of parameters, allowing us to design a system which is easier to tune.

Overall, the good recognition rates obtained by all kernels are presumably due to the fact that the color histogram representation captures an essential aspect of color appearance for this problem.

5.2 Dominant Color Detection

In this section we show that it is possible to train a learning machine to understand if an image would give to a human observer the impression of a dominant color. It is not clear how this issue of perceived color could be addressed using a straightforward algorithmic approach and the nature of the problem may in fact be rather different. We carried out a first set of experiments on the Artchive database (http://www.artchive.com), that contains thousands of images of paintings. A second set of experiments has been carried out on photo images both downloaded from the Web or acquired for the purpose.

The idea is to train a machine to "simulate" the perception of color of one or more people. We obtained results which are very promising by training a system on a dataset manually extracted by two different people.

Dominant colors in paintings
Here we present the results obtained on representing the perception of a green dominant on images of paintings. To this purpose we trained a one class SV-based classifier on a dataset of 51 positive examples. We tested the system on a test set of 53 positive examples and 164 negative examples. Positive examples test the robustness to false negatives, negative examples to false positives. The results obtained, in terms of *equal error rate*, i.e., the error obtained for the best compromise between false positives and false negatives, are summarized in Table 5.7. The performance obtained with histogram intersection kernel is well above standard kernels and comparable to a Heavy-tailed kernel with suitable parameters.

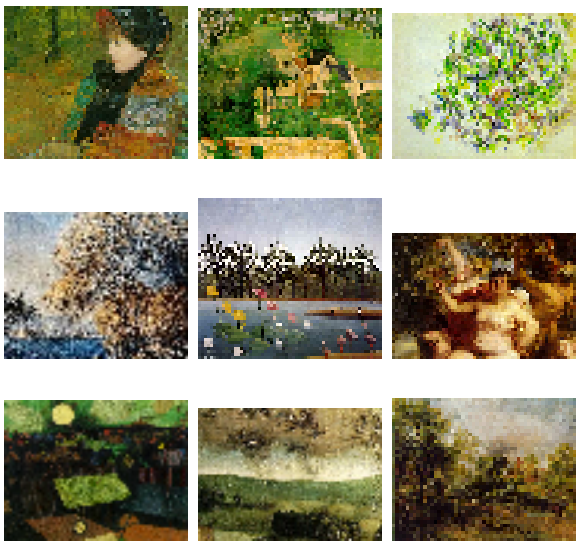

Figure 5.4: Images used in the experiments on the perception of dominant green. From top to bottom: 3 examples of training data, and examples used for testing (3 neg. and 3 pos.).

5.3 Cityscape Retrieval

Cityscapes can be described as images of a city taken from a distance sufficient to obtain a panoramic view [1]. A well known example of cityscapes are skylines.

In the problem of finding cityscapes an important cue is the geometry of the scene: a possible approach is to look for specific shapes such as lines or polygons.

This problem allow us to apply the histogram intersection kernel to data representations describing textures.

Again, we use a binary SVM classifier. In this case, while the first class is given by example images of cityscapes, the choice of representatives of the second class is less obvious: we collected images of various subjects, ranging from the cathegory of man-made objects to natural scenes.

In [BOV03b] we presented preliminary results on cityscape retrieval that we obtained with a representation based on co-occurrence matrices. We reasoned on the fact that cities could be seen as a random distribution of buildings, windows, roofs, at different scales, that generate a well defined sort of texture: the texture of cities.

[1] The relevance of this problem has been brought to our attention by Reuters. It is a typical task for content based image retrieval.

Kernel type	**HSV** (%)	**RGB** (%)
histogram intersection	85	86
2-nd deg polynomial	74	70
3-rd deg polynomial	69	70
Gaussian RBF kernel ($\sigma = 1.0$)	71	68
Gaussian RBF kernel ($\sigma = 1.5$)	70	71
Laplacian RBF kernel ($\sigma = 2$)	85	85
RBF kernel a=1 b=1.5 ($\sigma = 2$)	72	72

Table 5.7: E.E.R. for SVMs with different kernels, and for RBG and HSV representations, on the problem of "simulating" the perception of a dominant color.

Figure 5.5: Samples of cityscapes

We trained 19 binary SVMs with the same training set and 19 different representation of images (given by 19 different choices of displacements for the co-occurrence matrices). In the validation stage we tested each test image with all the 19 classifiers and applied a voting method to obtain the final solution.

Figure 5.6: Examples of non-cityscapes.

Then, we obtained better results [FOV05] (the recognition rate raised of about 10%) obtained describing the cityscape texture by means of *edge orientation histograms* similarly to [JV96].

We used the Canny edge detector to extract edges and estimate their orientation, and then compute an histogram.

The range bewteen 0 and 180 degrees is sampled in 10 intervals, adding a further bin for non-edge pixels. For this set of experiments we collected a dataset of 803 positive examples consisting of open city views and skylines (some examples of which are shown in Figure 5.5). Somewhat arbitrarily, we decided to sample the space of *non cityscapes* gathering 849 examples of city details (like street corners, buildings, etc.), different sorts of man-made objects, people and landscapes (see Figure 5.6 for some examples).

We randomly separated our dataset in training sets of 400 images of cityscapes and 400 images of negative exampes and perform a training step on these, and a validation step on

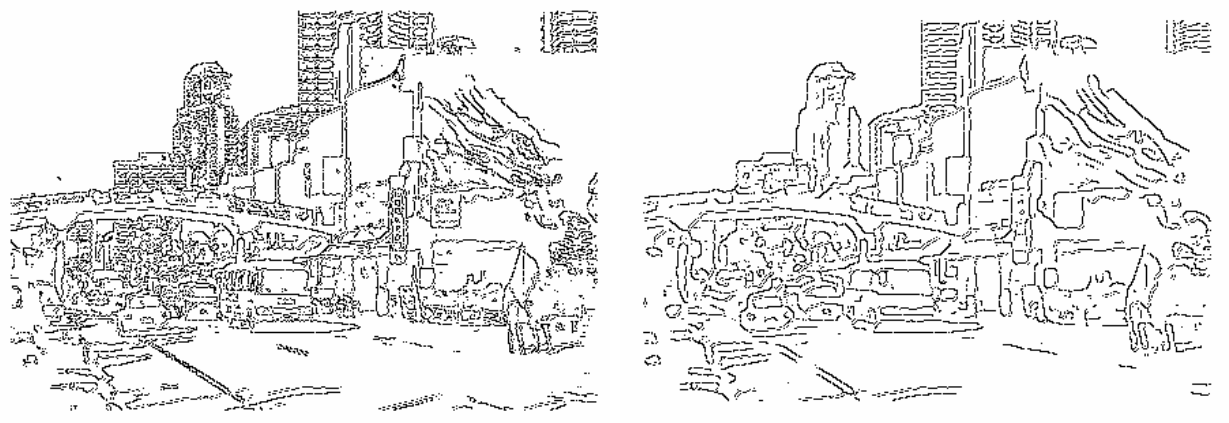

Figure 5.7: Edge detection with two different values of σ.

	$\sigma_e = 1.0$		$\sigma_e = 2.0$	
Kernel type	**Precision**	**Recall**	**Precision**	**Recall**
histogram intersection	90.13	85.05	87.10	82.23
linear	82.75	85.98	80.23	83.63
2-nd deg polynomial	85.9	84.21	83.83	83.41
4-rd deg polynomial	90.03	82.25	84.23	82.57
Gaussian RBF ($\sigma = 1.0$)	79.70	83.52	79.53	83.95
Laplacian RBF ($\sigma = 3$)	83.00	84.42	82.32	83. 10

Table 5.8: Accuracy of the cityscape retrieval system, for different kernels and two different image filterings. In the case of RBF kernels only the better recognition rate for a standard deviation σ varying between 0.5 and 2.5 is reported.

the remaining data. We repeated the operation of 5 different separations of the data to test the repeatibility of the results.

We compared the results obtain with the histogram intersection kernel with the ones achieved using standard kernels.

We obtained a perfect separation and a number of support vector that is always below the 50% of the training data, for $C = 100$. Table 5.8 shows the recognition rates of the system in terms of *recall* (how many cityscapes are retrieved by the system) and *precision* (how often a retrieval is false), for the different kernels used and two different filterings of the image

(Figure 5.7 shows the effects of a variation of the parameter on the edge detector output).

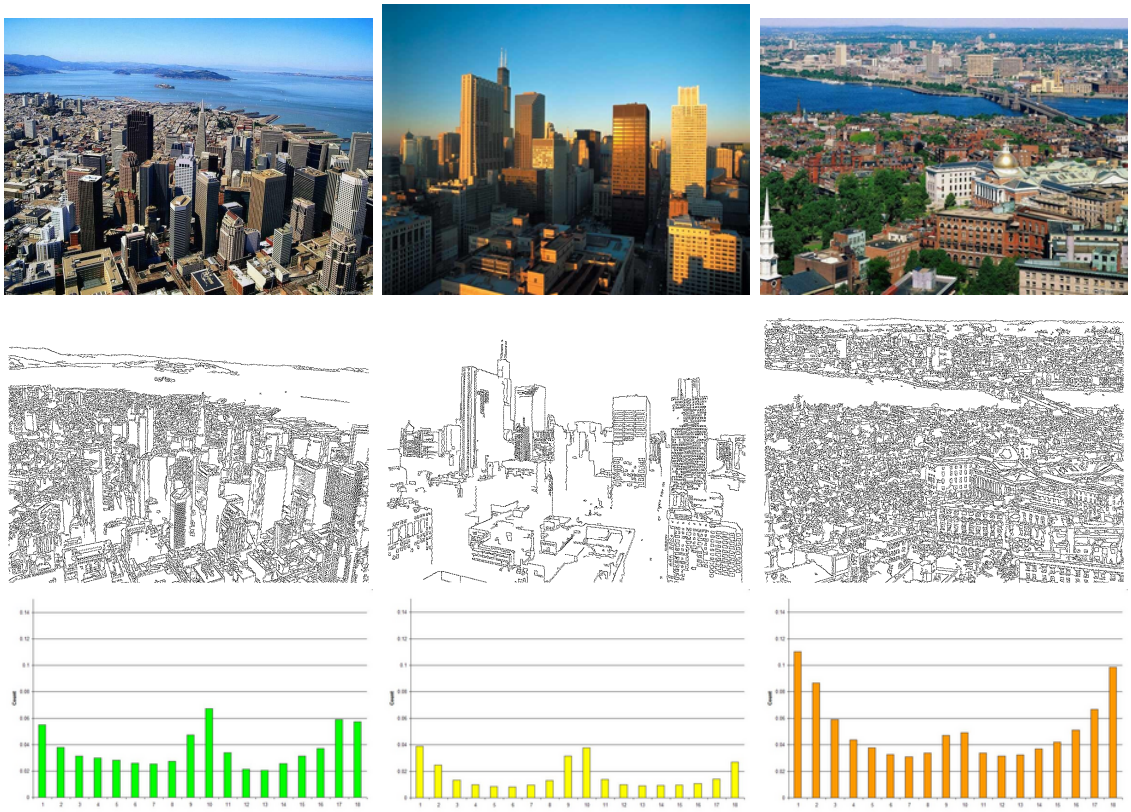

Figure 5.8: Samples of cityscapes (top), the detected edge points (middle) and the histograms of edge directions (bottom).

5.4 Color or Monochrome

This analysis can be performed at two levels, either by checking the information in the image header or, in case no such information is available, by studying the color distribution. We also trained a simple image classifier, to take into account borderline images (sepia color, drawings with few color details).

All the images that do not carry information about the color mode (color, gray) are analyzed with a simple SVM classifier, trained with a small training set of data (50 images per class):

- class 1 is made of color images of various nature, mostly natural images and a few drawings,
- class 2 is made of gray level, sepia images.

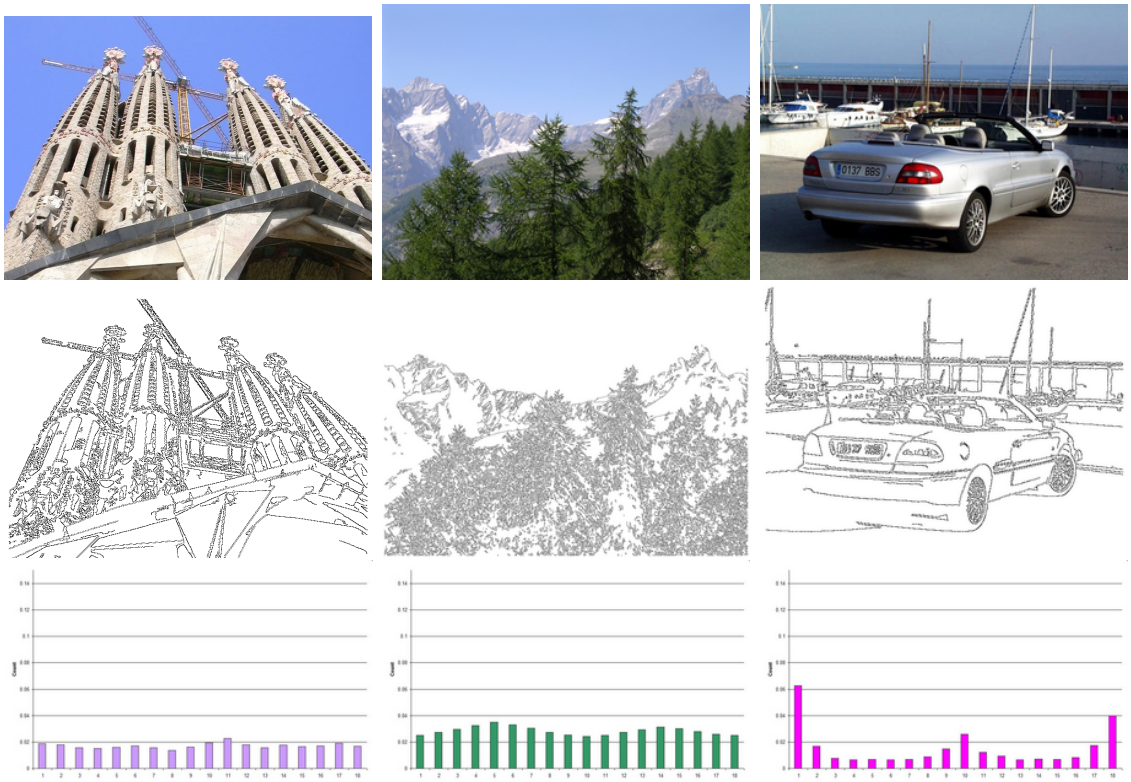

Figure 5.9: Samples of non cityscapes (top), the detected edge points (middle) and the histograms of edge directions (bottom).

We represent images with normalized color histograms both in HSV and RGB color space. We use a validation set of 400 color images, containing both photos and graphics, and 120 gray-level images. Four different training and validation sets are obtained by randomly sampling a data set made of 450 color images and 170 gray-level images. Table 5.9 shows recognition rates for the two color spaces and different bins. Results are shown as average values plus standard deviation, as the experiments have been repeated for different training and validations sets. From the experiments there is no evidence that one color space is better suited than the other to the problem in exam. The choice of the bins number is much more crucial, it is better to choose a finer description. We choose the 16 16 16 sampling, as it is the best compromise between performance and efficiency among the ones tried. In the case of HSV color space we also tried odd sampling of hue, saturation and value, but the results, not reported here, are in line with the ones of Table 5.9. We conclude this section with a remark on the errors obtained with the best performing classifier. Most of the monochrome images classified as color are professional photos and it is not clear whether they have been retouched or filtered, in one case there was a colored detail. All color images except one had a very poor color depth.

# of bins	**RGB** (r.r.)	**HSV** (r.r)
$4 \times 4 \times 4$	78.3 ± 0.6	78.6 ± 0.8
$8 \times 8 \times 8$	81.6 ± 0.8	78.6 ± 0.9
$16 \times 16 \times 16$	98.3 ± 0.5	98.5 ± 0.5
$20 \times 20 \times 20$	98.2 ± 0.4	98.9 ± 0.2

Table 5.9: Recognition rates (r.r.) of an SVM equipped with a histogram intersection kernel on the problem of gray-level against color images (see text).

5.5 Photos or Graphics

The problem of distinguishing real images from graphics and other synthetic pictures is known and studied in the literature (see, for instance, [VAF97, SC97, SPY02]) and it finds its main application in the domain of image search within the Web. The problem is more tricky than it seems especially because the class of non-photos is quite vast and it includes scanned hand drawings, computer generated graphics, plots, maps, cartoons, synthetically generated images, photo-realistic artworks. In the last two cases, the purpose of the designer was to produce art as similar as possible to real images. Also, mixed data are very popular on the Web and they can contain photos, maps, graphical elements, all inside the same image. Similarly to [VAF97, SC97, SPY02] we use low level feature describing color distributions and edges, starting from the simplistic assumption that natural images usually are more complex and detailed, while artificial images and graphics have sharper edges and more saturated colors.

As in [VAF97], we do not use mixed images for training and validation, since our approach is based on a global analysis over the whole image. At run time the effect of our filters over this sort of images is somewhat unpredictable and, for the moment, is taken care manually. We collected a set of 420 real color images and 420 images including cartoons, drawings, maps, synthetic images, adverts and logos. Again, we resampled the sets four times in order to obtain four different pairs of train and validation sets. We trained an SVM classifier with histogram intersection kernel, on a number of different image representations. Since small images and thumbnails are quite popular on the Web, we also tested the system on a data set of about 100 small images and graphics, the size of which is lower than 100×100 pixels.

The best results have been obtained with a combination of RGB color histograms and histograms of the gradient magnitude in correspondence of edge points. This supports the intuition that artificial images are populated of colors that are not common in real images, and are not well represented in coarse color representations (such as color histograms with a small number of bins, or color averages); also edges of artificial images tend to be sharper.

Figure 5.10: Examples of natural photos (top) and graphical images (bottom). Top, from left to right: a sunset with bright colors and few details, a panorama with a highly distinctive texture, a textured detail, an underexposed photo with few details and sharp edges. Bottom, from left to right: a digital map, an advert mixing a real image with graphical details, a computer-designed logo, a scanned watercolor painting.

image representation	**medium size** (r.r.)	**small size** (r.r)
HSV color hist. (4x4x4)	66.1 ± 0.4	76.3
RGB color hist. (4x4x4)	67.3 ± 0.5	76.8
HSV color hist. (16x16x16)	86.1 ± 0.2	86.8
RGB color hist. (16x16x16)	88.4 ± 0.3	92.3
histog. of edge strength	89.4 ± 0.4	90.1
histog. of edge direction	49.4 ± 0.4	48.3
histog. of edge chain length	84.1 ± 0.5	81.1
HSV hist + edge strength hist	93.2 ± 0.3	94.5
RGB hist + edge strength hist	93.8 ± 0.2	93.8
HSV hist + edge length hist	93.2 ± 0.3	92.9

Table 5.10: Recognition rates (r.r.) of an SVM equipped with histogram intersection kernel on the problem of color images against graphics; the experiments are performed on two validation sets of images of different size, and a number of different image representations (see text).

Other edge features, such as edge direction, gave poor results, while the histogram of chain edge lengths, normalized over the lateral size of the image are quite good descriptor for the problem in exam but suffer from image manipulation and rescaling. The results obtained are in line with state-of-the-art methods [VAF97, SC97, SPY02]. A more precise comparison can not be made as we all use different databases. Again, we conclude this section with a remark on the errors obtained with the best performing classifier. Most of the images classified as graphics looked as if they were manually retouched, and all the graphics classified as real images were photo-realistic artworks.

Experiments on Image retrieval

We implemented a system for automatically downloading images from the web [FOV] to the purpose of building training sets. The system is implemented by combining a "spider-like" algorithm, that visits randomly the web seeking images related to a subject of interest[2] and a part of filters and classifies implemented to the purpose of discarding small images, low quality images, black and white images and banners. We have run the system on three different text as seeds. Each run takes about two hours depending of the state of the network. We have kept the default values (i.e., minimum byte size equal to 10 Kb, minimum pixel size 100×100, JPG and GIF image formats), we do not check the aspect ratio. After discarding images that looked similar or alike, and manually eliminating 310 mixed images, we are left with 817 images.

[2]This is achieved by using the Google search libraries for text retrieval

We achieved recognition rates of 86.9% on graphics and 94.6% on photos. Many false positives on graphics are images retouched, filtered or otherwise manipulated.

5.6 Iconic Representation (3D object identification)

To validate the effectiveness of Hausdorff Kernels we pick some experiments out of a view-based identification system that we are building. The idea is to train a system to identify a 3D object from a single image, after it has been trained with only positive examples. Following an appearance-based approach, the training set of positive examples consists of a collection of views of a certain object taken by a camera moving around it. At run time a new image is considered and the system must answer the question of whether this image, or a part of it, is a different view of the same object. The actual identification is performed by means of classifiers that learn from positive examples. In what follows we report experiments obtained with a one-class Support Vector classifier and, for comparison, results achieved with Nearest Neighbours.

In this section we report experiments on the identification of a single object (a toy duck) in a cluttered environment. Figure 5.11 shows example frames belonging to the training sequence. In the described approach we keep preprocessing minimal, we only take care of discarding most of the background by performing a semi-automatic object tracking procedure based on Equation (4.6) with a tolerance on the gray-levels $\epsilon = 3$ and a neighbourhood $N_p = 3 \times 3$ (notice that most of the image is background and that the object of interest is not always in the same part of the image — therefore we can not identify a region of interest). A region containing the object of interest is extracted from frame 0 and compared with the succeeding frames in order to locate the best match. After several experiments we concluded that keeping the same template for as long as possible is better than updating the template at each step, in terms of localization precision.

Figure 5.12 shows a subsequence containing the results of tracking, where the template used appears in the first frame. When the matching results fall under a fixed threshold a new template is selected. Figure 5.13, for instance, shows the succeeding sequence.

The images obtained after the segmentation are 137×93; they are then resized to 69×47, therefore our data after preprocessing live in a space of 3243 dimensions.

In our experiments we train a one-class SV system on 460 images taken from a single image sequence. We tested the system on two different image sequences of the same object (a total of 660 images) and 1360 images of negative examples. The positive test data have been acquired in different days from different people, the negative test data contain other views of the same environment including similar objects (e.g., other toys). The results of the experiments performed are shown by means of Receiver Operating Characteristic

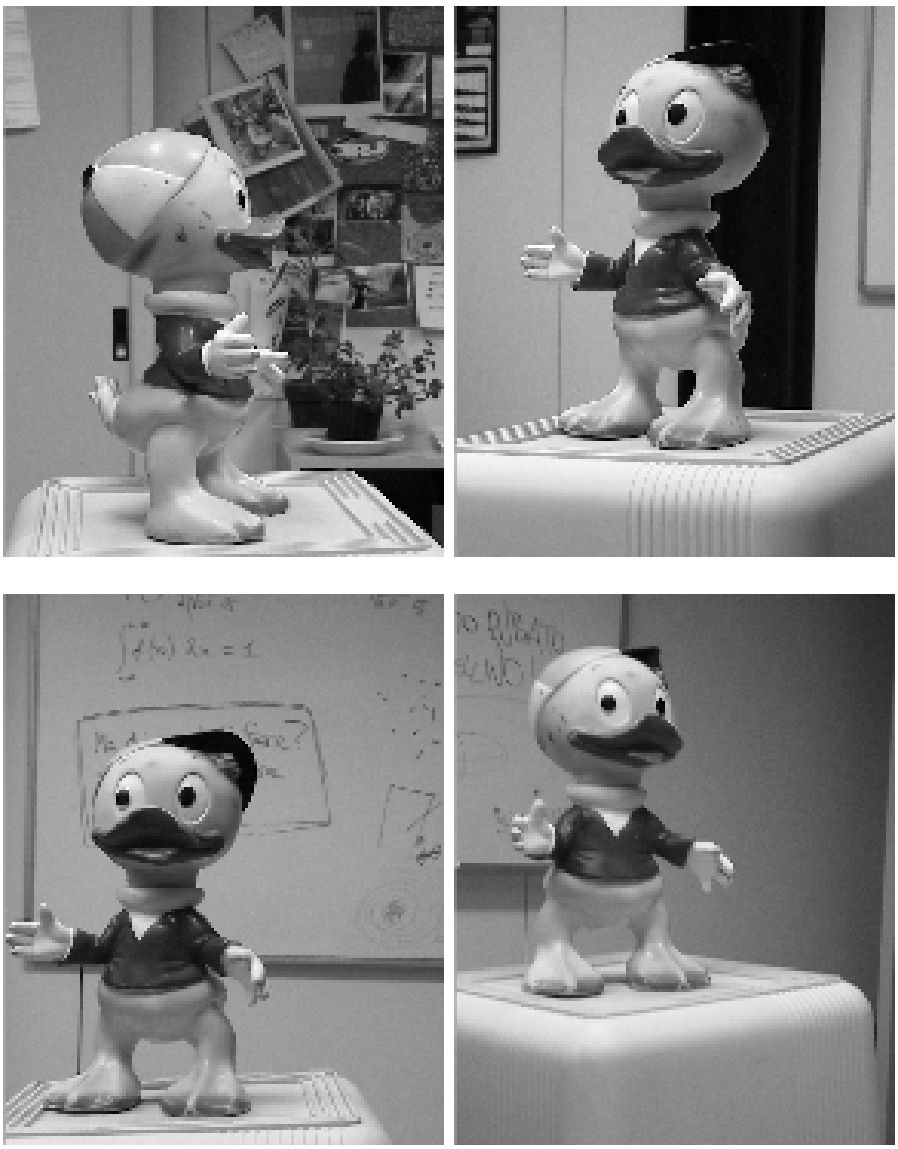

Figure 5.11: Example frames from the original training sequence

Figure 5.12: A sampling of the first tracking. The first image (top left) is the template which is used to track the toy throughout the sequence

.

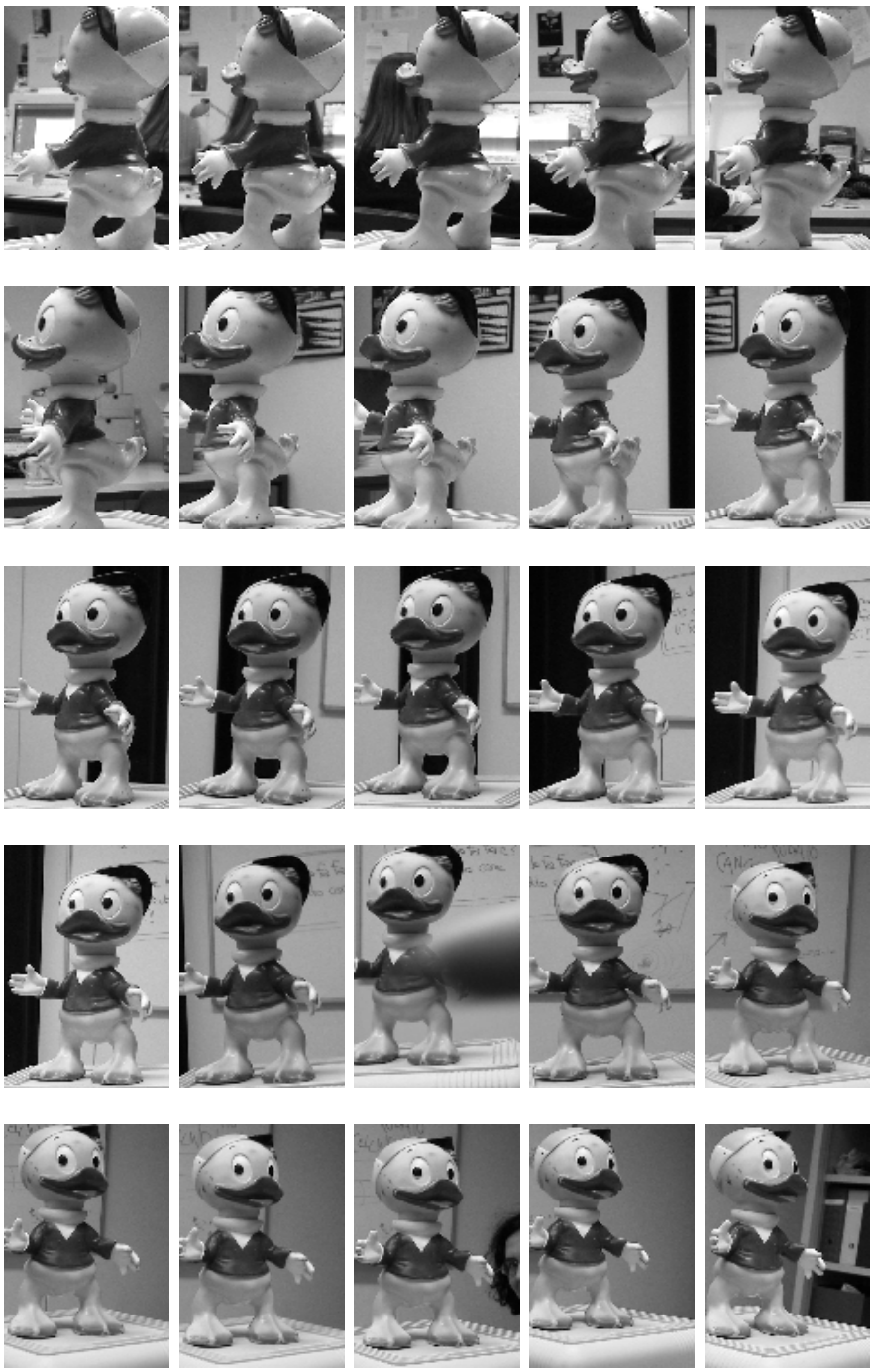

Figure 5.13: A sampling of the second tracking. The first image (top left) is the template which is used to track the toy throughout the remaining part of the sequence
.

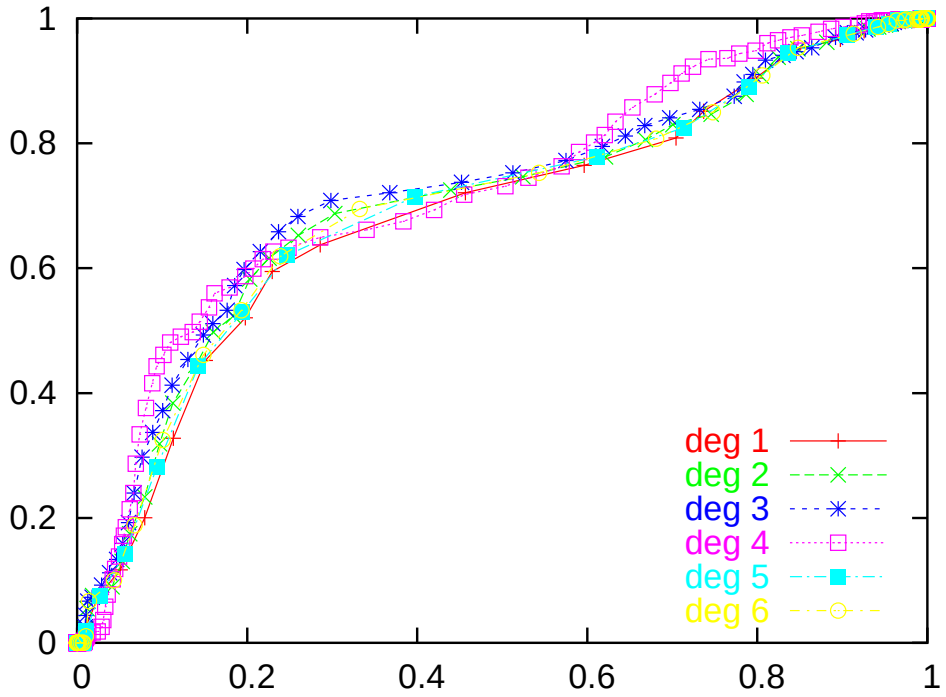

Figure 5.14: R.O.C. curves obtained with polynomial kernels (different degree)

(ROC) curves. Each point of a ROC curve represents a pair consisting of the false-alarm rate and the hit-rate of the system obtained by varying the radius of the sphere in feature space within a range. The system efficiency can be evaluated by the growth rate of its ROC curve, and for a given false-alarm rate, the better system will be the one with a higher hit probability. The overall performance of a system can be measured by the equal error rate (e.e.r.) representing the best compromise between false alarms and hit rate. The e.e.r. cab be computed by finding the intersection between the ROC curve and the function $f(x) = 1 - x$.

Since the results obtained seem hardly affected by changing the regularization parameter C of SVMs in the range $0.1 - 0.9$, we fix $C = 0.2$.

Figures 5.14, 5.15 and 5.16 show the performances obtained for various kernels on the identification problem described above. Figure 5.14 compares the results of polynomial kernels of various degrees. It shows immediately that more complex polynomials do not raise recognition accuracy.

Figure 5.15 shows the results obtained with Gaussian RBF kernels of different σs. As illustrated by the figure it is not easy to find a range of σ that leads to accettable results.

Figure 5.16 compares the performances of the modified Hausdorff kernel for different dila-

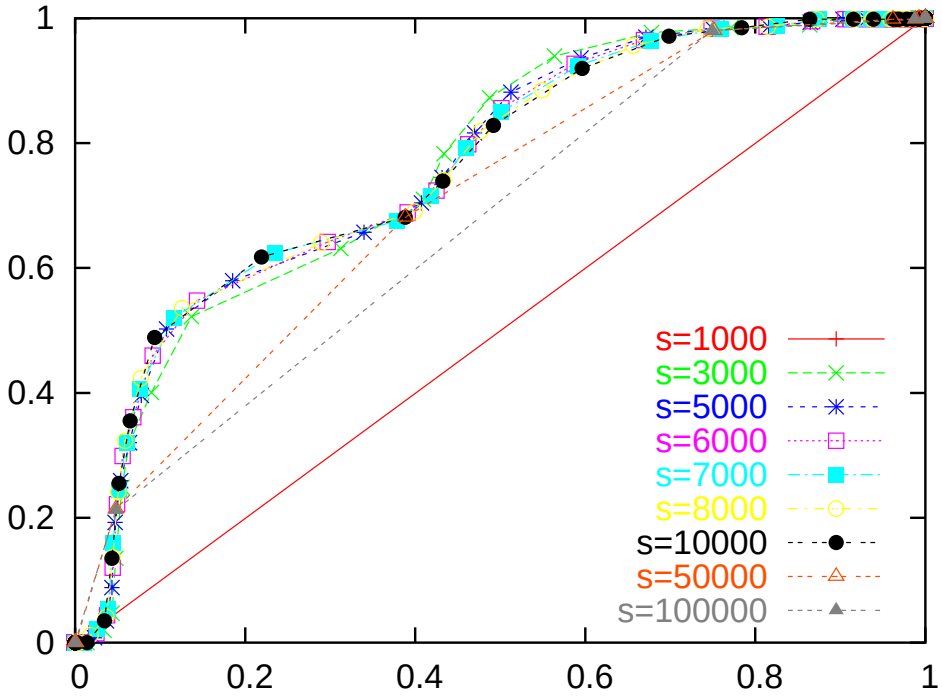

Figure 5.15: R.O.C. curves obtained with RBF Kernels (different sigma

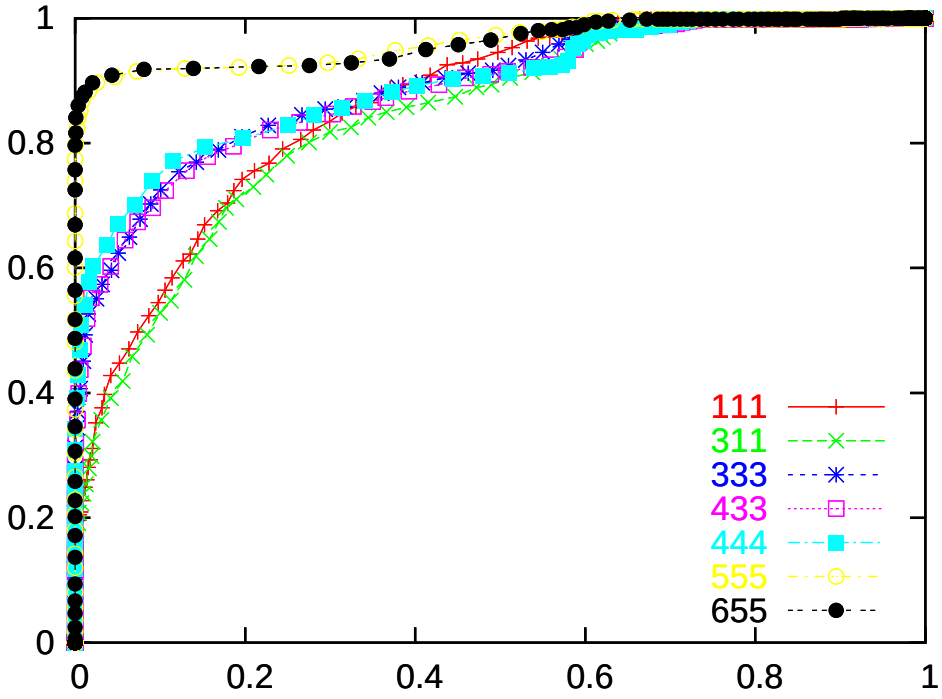

Figure 5.16: R.O.C. curves obtained with Hausdorff Kernel (different dilations

tions. The figure illustrates samples of possible dilations ranging from $(1, 1, 1)$ to $(5, 5, 5)$.

Conclusions

In this thesis we argue for the need of finding appropriate kernel functions, *kernel engineering*, for building effective trainable systems in the statistical learning framework. In the case of images, the problem of kernel engineering cannot be easily decoupled from the choice of the image description. Therefore, we make use of different descriptions depending on the nature of the considered classification problem. For classification problems like indoor/outdoor classification we choose to represent images through histograms (color, edges, co-occurrences, etc.), while for view-based object recognition we choose grey values and/or wavelets representations.

In our work we introduced and studied the mathematical properties of two image kernels: the Histogram Intersection kernel and the Hausdorff kernel. The Histogram Intersection kernel, derived from a similarity measure widely used in the computer vision community for color based object recognition, is found to be a very effective kernel for describing similarities between images in high level classification problems. The Histogram Intersection kernel can be implemented efficiently and does not require the introduction of additional parameters. The Hausdorff kernel, instead, which we derive as a specialization of a larger class of kernels defined on binary strings, appears to be well suited for measuring the similarity between image patches. We show that the Hausdorff kernel can be used to boost the performances of trainable 3D object detection systems trained on positive examples only.

From the experimental viewpoint we considered many classification problems: indoor/outdoor, cityscape vs non cityscape, photographs/graphics, main color detection; for each of this problems we collected training and tests sets and we tested each task obtaining very good performances in detection rates.

From our theoretical analysis and the obtained experimental results we conclude that the choice of the appropriate kernel can make the difference for a specific application.

This approach has demonstrated to be effective and has shown excellent results in each task we faced. Undoubtely, putting all the effort in choosing an appropriate representation and developing a suitable kernel results to be the strenght of this approach. On the other side, this can be considered as a weak point as the approach is, of course, not general. Our

primary goal is achieving a *general purpose classifier*, able to understand and describe the *semantic* of an image, a possible solution consists in implementing a *dictionary* of simple classifiers, each of those specified on a single task.

Our current work is focusing on building classifiers specified on different properties: *global properties*, such as day/night, indoor/outdoor, drawing/pictures; *objects*, such as presence of faces, people, cars, traffic lights, doors, roofs and *stuff*, intended as homogeneous part of the image: sky, rocks, grass, roads, snow.

One possible approach to address this problem is using a system able to automatically select the salient features for each category. First it would be necessary to list as many features as possible for each datum, then by a *feature selection* step [Fra05], find the best representation for each class of input data.

Bibliography

[AVZ] A.K. Jain A. Vailaya, M. Figueiredo and H.J. Zhang. Content-based hierarchical classification of vacation images. In *IEEE Multimedia Computing and Systems*.

[AVZ98] A. Jain A. Vailaya and H. J. Zhang. Image classification: City images vs landscapes. *Pattern Recognition*, 1998.

[BFOV02] A. Barla, E. Franceschi, F. Odone, and A. Verri. Image kernels. *Lecture Notes in Computer Science*, 2388:83–??, 2002.

[BHP00] T. Poggio B. Heisele and M. Pontil. Face detection in still gray images. Technical report, Artificial Intelligence Laboratory, MIT, 2000.

[Bis94] Christopher M. Bishop. Novelty detection and neural network validation. *IEE Proceedings — Vision, Image and Signal Processing*, 141(4):217–222, aug 1994. Document No. 19941330.

[Bis96] C. Bishop. *Neural Networks for Pattern Recognition*. 1996.

[BL76] Ruzena Bajcsy and Lawrence Lieberman. Texture gradient as a depth cue. *Computer Graphics and Image Processing*, 5(1):52–67, March 1976. BAJESY76a.

[BL84] V. Barnett and T. Lewis. *Outliers in statistical data*. Wiley, 1984.

[BOV02] A. Barla, F. Odone, and A. Verri. Hausdorff kernel for 3D object acquisition and detection. *Lecture Notes in Computer Science*, 2353:20–??, 2002.

[BOV03a] A. Barla, F. Odone, and A. Verri. Histogram intersection kernel for image classification. *Lecture Notes in Computer Science*, 2003.

[BOV03b] A. Barla, F. Odone, and A. Verri. Old fashioned state-of-the-art image classification. In *IEEE International Conference on Image Analyisi and Processing*, 2003.

[BP] B.Vidakovic and P.Mueller. Wavelets for kids. Technical report, Duke University.

[BV96] T. Poggio B. Schoelkopf, K. Sung, C. Burges, F. Girosi, P. Niyogi and V. Vapnik. Comparing support vector machines with gaussian kernels to radial basis function classifiers. Technical Report CBCL-142, MIT Artificial Intelligence Laboratory, December 2 1996.

[Cam00] C. Campbell. Kernel methods: A survey of current techniques. Technical report, Department of Engineering Mathematics, Bristol University, Bristol BS8 1TR, United Kingdom, 2000.

[Can86] F J Canny. A computational approach to edge detection. *IEEE Trans PAMI*, 8(6):679–698, 1986.

[CB01] C. Campbell and K. P. Bennett. A linear programming approach to novelty detection. *Advances in Neural Information Processing Systems*, 13, 2001.

[CHV99] O. Chapelle, P. Haffner, and V. Vapnik. Svms for histogram-based image classification. *IEEE Transactions on Neural Networks, special issue on Support Vectors*, 1999.

[CP] C.Papageorgiou and T. Poggio. A trainable system for object detection.

[CS02] F. Cucker and S. Smale. On the mathematical foundation of learning. *Bull. A.M.S.*, 2002.

[CST00] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge Univ. Press, 2000.

[CTH84] R. W. Conners, M. M. Trivedi, and C. A. Harlow. Segmentation of a high resolution urban scene using texture operators. *Computer Vision, Graphics, and Image Processing*, 25:273–310, 1984.

[CV95] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:1–25, 1995.

[DHS00] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley-Interscience, 2000.

[DM79] L. S. Davis and A. Mitiche. Edge detection in texture. Technical Report CS-TR-79-111, University of Texas at Austin, Department of Computer Sciences, October 1 1979. Tue, 10 Aug 104 18:46:51 GMT.

[DM81] Larry S. Davis and Amar Mitiche. Edge detection in textures — maxima selection. *Computer Graphics and Image Processing*, 16(2):158–165, June 1981.

[DVRC+04] E. De Vito, L. Rosasco, A. Caponnetto, M. Piana, and A. Verri. Some properties of regularized kernel methods. *Journal of Machine Learning Research*, 2004.

[Eak] J.P. Eakins. Techniques for image retrieval. *Library and Information Briefings*.

[Eak96] J.P. Eakins. Automatic image content retrieval are we getting anywhere? In *Proceedings of Third International Conference on Electronic Library and Visual Information Research (ELVIRA3)*, pages 123–135, De Montfort University,Milton, 1996.

[ED] T.D.DeRose E.J.Stollnitz and D.H.Salesin. Wavelets for computer graphics: A primer. Technical report, University of Washington.

[EOG] R. Freund E. Osuna and F. Girosi. Training support vector machines: an application to face detection. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[EPP00] T. Evgeniou, M. Pontil, and T. Poggio. Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13:1–50, 2000.

[FOA01] E. Trucco F. Odone and A.Verri. General purpose matching of grey level arbitrary images. *Lecture Notes in Computer Science*, 2059:573–??, 2001.

[For90] D. A. Forsyth. A novel algorithm for color constancy. *Intern. J. Computer Vision*, 5:5–35, 1990.

[For03] *Computer Vision - A modern approach*. Prentice Hall, 2003.

[FOV] E. Franceschi F. Odone, A. Barla and A. Verri. Web tools to support image classifiaction.

[FOV05] A. Barla F. Odone and A. Verri. Building kernels from binary strings for image matching. *IEEE Transactions on Image Processing*, 2005.

[Fra05] E. Franceschi. *Advanced hypothesis testing techniques and their application to image classification*. PhD thesis, DISI - Dipartimento di Informatica , Universitá di Genova, 2005.

[GGC91] H. Greenspan, R. Goodman, and R. Chellappa. Texture analysis via unsupervised and supervised learning. In *IEEE International Joint Conference on Neural Networks (5th IJCNN'91)*, volume I, pages I–639–I–644, Seattle, WA, 1991. IEEE.

[GJKK88] S. A. Shafer G. J. Klinker and T. Kanade. The measurement of highlights in color images. *Intern. J. Computer Vision*, 2:7 – 32, 1988.

[G.K] G.Kaiser. *A friendly guide to Wavelets*. Birkhäuser.

[GLK00] G. Guodong, S. Li, and C. Kapluk. Face recognition by support vector machines. In *Proc. IEEE International Conference on Automatic Face and Gesture Recognition*, pages 196–201, 2000.

[GP] M. Gorkani and R. W. Picard. Texture orientation for sorting photos at a glance. In *IEEE Conference on Pattern Recognition*.

[Har80] R M Haralick. Edge and regional analysis for digital image data. *CGIP*, 12:60–73, 1980. HARALICK80.

[HHP01] B. Heisele, P. Ho, , and T. Poggio. Face recognition with support vector machines: global versus component-based approach. In *ICCV*, pages 688–694, 2001.

[HKR93] D. Huttenlocher, G. Klanderman, and W. Rucklidge. Comparing images using the hausdorff distance. *Trans. on PAMI*, 15(9), 1993.

[HP00] Tomaso Poggio Bernd Heisele and Massimiliano Pontil. Face detection in still gray images. Technical Report CBCL-187, MIT Artificial Intelligence Laboratory, May 7 2000.

[HS92] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision*. Addison-Wesley, 1992.

[HSD73] R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *IEEE Transactions of Systems, Man and Cybernetics*, 3:610–621, 1973.

[HSPP01] B. Heisele, T. Serre, M. Pontil, and T. Poggio. Component-based face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 657–662, 2001.

[HTF01] T. Hastie, R. Tibshirani, and J.H. Friedman. *The Elements of Statistical Learning*. Springer Verlag, 2001.

[JF91] Anil K. Jain and Farshid Farrokhnia. Unsupervised texture segmentation using Gabor filters. *Pattern Recognition*, 24(12):1167–1186, 1991.

[JH98] T. S. Jaakkola and D. Haussler. Exploiting generatie models in discriminative classifiers. In *Advances in Neural Information Processing Systems*, volume 11, 1998.

[JH99] T. S. Jaakkola and D. Haussler. Probabilistic kernel regression models. In *Proc. of the 1999 Conference on AI and Statistics*, 1999.

[JHH] V.Blanz J. Huang and B. Heisele. Face recognition with support vector machines and 3d head models.

[JHZ97] M. Mitra W.-J. Zhu J. Huang, S. Ravi Kumar and R. Zabih. Image indexing using color correlograms. In *IEEE Conference on Computer Vision and Pattern RecognitionP*, pages 762–768, 1997.

[JLZZ03] F. Jing, M. Li, H.J. Zhang, and B. Zhang. Support Vector Machines for region-based image retrieval. In *Proc. IEEE International Conference on Multimedia and Expo*, 2003.

[JMKL00] K. Jonsson, J. Matas, J. Kittler, and Y. Li. Learning Support Vectors for face verification and recognition. In *Proc. IEEE Int Conf on Automatic Face and Gesture Recognition*, 2000.

[JT00] A. K. Jain and M. Tuceryan. Texture analysis, August 22 2000.

[JV96] A. K. Jain and A. Vailaya. Image retrieval using color and shape. *Pattern Recognition*, 29:1233–1244, august 1996.

[KF95] Moya M. Hostetler L. Koch, M. and R. Fogler. Cueing, feature discovery and one-class learning for synthetic aperture radar auomatic target recognition. *Neural Networks*, 1995.

[KV98] V. Kovalev and S. Volmer. Color co-occurrence descriptor for querying-by-example. In *Multimedia Modeling*, page 32ff, Lausanne, Switzerland, 1998.

[LA] J. Luo and A.Savakis. Indoor vs outdoor classification of consumer photographs using low-level and semantic features. In *ICIP - International Conference on Image Processing*.

[LJW97] I. D. Longstaff, P. T. Jackway, and R. F. Walker. Recent developments in the use of the co-occurrence matrix for texture recognition, July 22 1997.

[LLW02] Y. Lin, Y. Lee, and G. Wahba. Support vector machines for classification in nonstandard situations. *Machine Learning*, 2002.

[Mac92] David J. C. MacKay. *Bayesian Modeling and Neural Networks*. PhD thesis, Dept. of Computation and Neural Systems, CalTech, 1992.

[Mal89] S. Mallat. Multiresolution approximation and wavelets. *Trans. of American Math. Soc.*, 315:69–88, 1989.

[MC93] B. S. Manjunath and Rama Chellappa. A unified approach to boundary perception: Edges, textures and illusory contours. *IEEE Transactions on Neural Networks*, 4(1):96–108, january 1993.

[MH79] D. Marr and E. Hildreth. Theory of edge detection. *Proceedings Royal Society of London Bulletin*, 204:301–328, 1979.

[MOP] P. Sinha E. Osuna M. Oren, C. Papageorgiou and T. Poggio. Pedestrian detection using wavelet templates. In *IEEE Conference on Computer Vision and Pattern Recognition*.

[MP89] J. Malik and P. Perona. A computational model of texture perception. Technical Report UCB/CSD 89/491, Computer Science Division, University of California, Berkeley, CA, February 1989.

[MPP01] A. Mohan, C. Papageorgiou, and T. Poggio. Example-based object detection in images by components. *IEEE Trans. Patt. Anal. Mach. Intell.*, 23:349–361, 2001.

[MW86] L. T. Maloney and B. Wandell. Color constancy: A method for recovering surface spectral reflectance. *J. Opt. Soc. Amer.*, A 3(1):29 – 33, 1986.

[MY] B. Moghaddam and M.-H. Yang. Gender classification with support vector machines. In *IEEE International Conference on Automatic Face and Gesture Recognition*.

[NS90] C. L. Novak and S. A. Shafer. Supervised color constancy using a color chart. Technical Report CUM-CS-90-140, School of Computer science, Carnagie Mellon University, 1990.

[NV] Gudivada V N and Raghavan V V. Content-based image retrieval systems.

[NV95] Gudivada V N and Raghavan V V. Design and evaluation of algorithms for image retrieval by spatial similarity. In *ACM Transactions on Information Systems*, volume 13(2), pages 115–144, 1995.

[PCGV02] M. Partio, B. Cramariuc, M. Gabbouj, and A. Visa. Rock texture retrieval using gray level co-occurrence matrix. In *5th Nordic Signal Processing Symposium*, 2002.

[PV] Massimiliano Pontil and Alessandro Verri. Support vector machines for 3d object recognition.

[PV97] Massimiliano Pontil and Alessandro Verri. Properties of support vector machines. Technical Report CBCL-152, MIT Artificial Intelligence Laboratory, August 27 1997.

[RH99] D. Roobaert and M. V. Hulle. View-based 3d object recognition with support vector machines. 1999.

[RS] Tarassenko L. Pardey J. Roberts, S. and D. Siegwart. A validation index for artificial neural networks.

[RW90] Todd R. Reed and Harry Wechsler. Segmentation of textured images and gestalt organization using spatial/spatial-frequency representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-12(1):1–12, January 1990.

[RYH] R. Jin R. Yan, Y. Lin and A. Hauptmann. On predicting rare classes with svm ensembles in scene classification. In *In Proc. of ICASSP*.

[SB91] Swain and Ballard. Color indexing. *IJCV: International Journal of Computer Vision*, 7, 1991.

[SC97] J. R. Smith and S.-F. Chang. Visually searching the web for content. In *IEEE Multimedia*, 1997.

[Sch] B. Sch˙Technical report.

[SK] H. Schneiderman and T. Kanade. A statistical method for 3d object detection applied to faces and cars. In *CVPR*.

[SP98] M. Szummer and R. W. Picard. Indoor-outdoor image classification. In *IEEE Intl Workshop on Content-based Access of Image and Video Databases*, 1998.

[SPST+01] B. Schölkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, and R.C.Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 2001.

[SPY02] J.C. Handley Z. Fan S. Prabhakar, H. Cheng and Y.Lin. Picture-graphics color image classification. In *IEEE International Conference on Image Processing*, 2002.

[SS02] B. Schölkopf and A.J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[SSSV98] B. Schölkopf, P. Simard, A. Smola, and V. Vapnik. Prior knowledge in support vector kernels. In *Advances in Neural Inf. Proc. Systems*, volume 10, pages 640–646. MIT Press, 1998.

[STC04] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge Univ. Press, 2004.

[Tax01] David M. J. Tax. One-class classification; concept-learning in the absence of counter-examples. Technical Report ASCI Dissertation Series-65, Ph.D. thesis Delft University of Technology, june 2001.

[TD99] D. Tax and R. Duin. Data domain description by support vectors. In M. Verleysen, editor, *Proceedings of ESANN99*, pages 251–256. D. Facto Press, 1999.

[TD01] David M. J. Tax and Robert P. W. Duin. Combining one-class classifiers, july 2001.

[TJ90] Mihran Tuceryan and Anil K. Jain. Texture segmentation using voronoi polygons. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-12(2):211–216, February 1990.

[TYD] D. Tax, A. Ypma, and R. Duin. Support vector data description applied to machine vibration analysis.

[VAF97] M.J. Swain V. Athitsos and C. Frankel. Distinguishing photographs and graphics on the world wide web. In *IEEE Workshop on content-based access on image and video libraries*, 1997.

[Vap95] V. Vapnik. *The nature of statistical learning Theory*. John Wiley and sons, New York, 1995.

[Vap98] V. Vapnik. *Statistical learning theory*. John Wiley and sons, New York, 1998.

[VP87] H. Voorhees and T. Poggio. Detecting textons and texture boundaries in natural images. In *First International Conference on Computer Vision, (London, England, June 8–11, 1987)*, pages 250–258, Washington, DC., 1987. IEEE Computer Society Press.

[VZ] A. Jain A. Vailaya and H.J. Zhang. City vs. landscape. In *IEEE Workshop on Content-Based Access of Image and Video Libraries*.

[WDR76] J S Weszka, C R Dyer, and A Rosenfeld. A comparitive study of texture measures for terrain classification. *IEEE Trans SMC*, 6(4):269–285, 1976. WESZKA76.

[WJL95] R. F. Walker, P. Jackway, and I. D. Longstaff. Improving co-occurrence matrix feature discrimination. In *Proceedings of Digital Image Computing: Techniques and Applications*, pages 643–648, 1995.