UvA-DARE (Digital Academic Repository)

Learning spatial relations for object recognition

Pham, T.V.

Publication date

2005

Document Version

Final published version

[Link to publication](#)

Citation for published version (APA):

Pham, T. V. (2005). *Learning spatial relations for object recognition*. [Thesis, fully internal, Universiteit van Amsterdam].

General rights

It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations

If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: <https://uba.uva.nl/en/contact>, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.

Learning Spatial Relations for Object Recognition

Pham Viet Thang

This book is typeset by the author using L^AT_EX2_ε.

Printing: Febodruk BV, Enschede, the Netherlands.

Publications:

Chapter 2: T. V. Pham, M. Worring and A. W. M. Smeulders. Face detection by aggregated Bayesian network classifiers. *Pattern Recognition Letters*, 23(4):451-461, February 2002.

Chapter 3: T. V. Pham and A. W. M. Smeulders. Statistical strategy for object class recognition using part detectors. In A. Leonardis, H. Bischof, editors, *Statistical Learning in Computer Vision*, pages 33-48, May 2004.

Copyright © 2005 by Thang V. Pham.

All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without written permission from the author.

ISBN 90-5776131-9

Learning Spatial Relations for Object Recognition

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam,
op gezag van de Rector Magnificus prof. mr. P.F. van der Heijden
ten overstaan van een door het College voor Promoties ingestelde commissie,
in het openbaar te verdedigen in de Aula der Universiteit
op dinsdag 1 februari 2005 te 10.00 uur

door

Viet Thang Pham

geboren te Hanoi, Vietnam

Promotiecommissie:

Promotor: Prof. dr. ir. A. W. M. Smeulders

Overige leden: Dr. ir. R. P. W. Duin
Dr. J. E. den Hartog
Prof. dr. ir. E. Backer
Prof. dr. D. M. Gavrilă
Prof. dr. P. W. Adriaans

Faculteit: Natuurwetenschappen, Wiskunde & Informatica

The work described in this thesis was supported by the ICES-KIS MIA-project and TNO-TPD.

The work described in this thesis has been carried out at the Intelligent Sensory Information Systems group of the University of Amsterdam, within graduate school ASCI. ASCI dissertation series number 111.

Intelligent Sensory Information Systems
University of Amsterdam
The Netherlands

Contents

1	Introduction	1
1.1	Object recognition using pictorial examples	1
1.2	Concepts of statistical learning theory	3
1.3	Approaches to object recognition	5
1.4	Problem statement	7
2	Face detection by aggregated Bayesian network classifiers	9
2.1	Introduction	9
2.2	Literature on appearance-based face detection	10
2.3	Data space exploitation and aggregated classifiers	12
2.4	Forest-structured Bayesian network classifier	13
2.5	Face detection system	15
2.5.1	Data for training	16
2.6	Experimental results	18
2.6.1	Experiment with the number of component classifiers K	18
2.6.2	Experiment with the Bayesian network classifier	18
2.6.3	Experiment on the full image test set	20
2.7	Discussion and conclusion	20
3	Statistical strategy for object class recognition using part detectors	25
3.1	Introduction	25
3.2	Bayes formulation of the problem	27
3.3	Learning an object model	28
3.3.1	The likelihood	28
3.3.2	The prior	29
3.3.3	Parameter estimation	32
3.4	Searching for the optimal solution	32
3.5	Experiments	34
3.5.1	On synthetic data	34
3.5.2	On real images	34
3.6	Discussion and conclusion	38

4	Learning spatial relations in object recognition	41
4.1	Introduction	41
4.2	Methods	44
4.2.1	Part-based learning	44
4.2.2	Bayesian network learning	46
4.2.3	Bayesian network learning with object masking	47
4.3	Datasets and annotation	47
4.3.1	The face dataset	48
4.3.2	The horse dataset	48
4.4	Experimental results	51
4.4.1	Object localization	51
4.4.2	Image classification	56
4.5	Discussion and conclusion	58
5	Sparse representation for coarse and fine object recognition	61
5.1	Introduction	61
5.2	Formulation of the problem	62
5.2.1	Related work	63
5.2.2	Problem statement	66
5.3	Model learning and recognition	67
5.3.1	Gaussian derivative bases	67
5.3.2	Piecewise polynomial approximation	69
5.3.3	Model learning with matching pursuit	70
5.3.4	Recognition	71
5.4	Experiments	75
5.4.1	Compactness of the representation	75
5.4.2	Recognition performance	76
5.5	Discussion and conclusion	78
6	Quadratic boosting	81
6.1	Introduction	81
6.2	Problem statement	83
6.3	Background	84
6.3.1	The AdaBoost algorithm	84
6.3.2	Optimization criterion for a base learner	84
6.3.3	VC dimension and bounds on generalization error	86
6.4	Quadratic boosting	88
6.4.1	Derivation of the intermediate base learner	88
6.4.2	The algorithm	90
6.4.3	Generalization error	90
6.5	Experiments	92
6.5.1	On synthetic data	92
6.5.2	On standard machine learning datasets	94
6.5.3	On datasets for learning in vision	98
6.6	Discussion and conclusion	100

7 Summary and discussion	103
7.1 Summary	103
7.2 Discussion	105
Bibliography	106
Samenvatting	115
Acknowledgements	119

Chapter 1

Introduction

1.1 Object recognition using pictorial examples

Visual object recognition plays an important role in our interaction with the world. From the simple task as picking up a book on a table to the complex task of riding a motorbike, we use our vision system to guide actions. Vision is used in a thousand different places and for a thousand different purposes. Assessing the quality of object after production is - in many situations - a tedious job. For example, food processing workers are constantly surveilling a fast-moving belt for a bad apple, or the police is looking for specific suspect in surveillance videos. In some cases, the working environment is dangerous such as in mining or space exploration. This selection, inspection and quality control are motivations for computerized solutions of object recognition.

Unlike humans, computers are procedurally very precise. They need to know, and rightly so, *what* we want them to recognize. Describing such turns out to be a non-trivial task. After all, an object is an abstract concept for which few aspects are valid in general. For certain objects or object classes, we may have the luxury of knowing how they have been created or composed, we know their surface properties, or their limbs or their geometry, or the response to illumination. We may be able to transfer the knowledge in the form of object models to some formal description computers can understand. For other objects such as rivers, clouds, mountains, trees and birds, a model is a different issue. It will never be precise neither in the surface properties, nor in the geometry nor in the response to illumination. Even the description of a stochastic model is hard as the appearance depends on the life past. Nevertheless, everybody will agree they are from an (approximate) class in the sense that they form a bounded region in concept space. In this thesis, we explore approaches to object description starting from pictorial examples rather than an explicit geometrical, surface and surface reflectance model.

Computers are not very smart in the sense of self-abstraction. We need to show them *how* to recognize objects. For instance, Grimson [37] provides an extensive list of strategies for recognition when various types of object models are available.

The case where knowledge is represented by pictorial examples is different. Examples are considered as experience from which one derives rules to recognize new instances. This is a prediction problem where one has to define a concept space and subsequently bounding the region of interest.

How hard is this prediction problem? One may have a rough idea by looking at the sources of variation that occur. Consider a problem which has received considerable attention in computer vision research because of a wide range of applications, namely recognizing human faces. The faces appear in different location, scales, orientation and head poses. The imaging conditions such as illumination and subsequent shadow also affect the appearance of the face, as well as weather conditions like rain and sweat. Moreover, human faces are non-rigid, as there are variations due to facial expression. The presence of attributes such as glasses, make-up, hair and beard contribute substantially to the variation in the (designed) facial appearance. Furthermore, there are differences among races, and between males and females. When the purpose is to capture all, the implicit assumption is that the model is able to handle at least these sources of variation upto a level compatible with human performance in detection.

There are several computer vision problems that go under the general name of *object recognition*. Here we mention two notions only. The classification problem aims to find the identity of the object in the input image. Typically, the appearance of one object dominates the image. An example is the individual face recognition task as addressed by Turk and Pentland [100]. Another problem is object detection, aiming to find all instances of the target object in an input image with clutter. Two examples are the face detection method developed by Sung and Poggio [94] and the pedestrian detection method of Gavrilu and Philomin [30].

Machine learning and pattern recognition are the fields engaged in the construction of statistical models from examples and derived prediction rules. The raw image material is transferred in features carrying a higher abstraction of the information in the example images. Examples are not the only source of *a priori* information. The physical, cultural, perceptual, and other external sources of knowledge help in the design of rules for recognizing specific classes of objects and hence in creating complete recognition systems. Amidst those aspects, this thesis concentrates on one such a source specifically, namely the spatial relationships as the features and the mechanism to express the *a priori* knowledge about the objects.

A majority of methods construct a feature space for object images and subsequently employing existing statistical models for recognition. The image histogram is a frequently used feature when it is desirable to ignore the spatial relations within image, while an appearance template encodes the rigid spatial relations of an object. Our aim is to understand and to exploit spatial relationships between these two ends of the spectrum, rigid and non-rigid, in recognition.

1.2 Concepts of statistical learning theory

This section briefly reviews key terms and concepts in statistical learning theory. Basic knowledge about statistics and probability theory is assumed. For a solid foundation, the reader is referred to [21, 105, 39].

The aim of a learning algorithm is to predict the output value $y \in \mathcal{Y}$ from an input observation $x \in \mathcal{X}$, given a set $\mathcal{D}$ of N known examples, $\mathcal{D} = \{(x_n, y_n)\}$. Let $f : \mathcal{X} \rightarrow \mathcal{Y}$ denote the prediction function. For the completeness of the presentation, we restrict ourselves to the binary case $\mathcal{Y} = \{-1, +1\}$ where -1 and $+1$ are class labels. For example in object recognition, they represent the class of background and foreground patches, respectively, or the labels face and non-face.

The learning problem aims for a classifier f that has a low generalization error $\epsilon(f)$

$$\epsilon(f) = E_{P(x,y)} \left\{ \frac{1 - yf(x)}{2} \right\} \quad (1.1)$$

where $E_{P(x,y)}$ denotes expectation over the joint distribution $P(x, y)$. A related notion is the empirical error measuring the performance of f over the set of training examples $\mathcal{D}$

$$\hat{\epsilon}(f, \mathcal{D}) = \frac{\# \{ y_n f(x_n) \mid y_n f(x_n) < 0; \ n = 1, \dots, N \}}{N} \quad (1.2)$$

where $\#$ denotes the number of elements of a set.

Typically, one can compute the empirical error $\hat{\epsilon}(f, \mathcal{D})$ given a classifier f but not the generalization error $\epsilon(f)$, since the probability $P(x, y)$ is unknown. Moreover, a low empirical error does not guarantee a low generalization error. This is known as the problem of overfitting where a classifier performs well on empirical data $\mathcal{D}$, but has poor generalization.

The practical approach to estimate the true error of a classifier is to divide the available data into a training set and a test set. The classifier is learned using the training data and evaluated on the test set. There are several ways the available data can be split. The cross validation method splits the data into equal partitions, then uses them one by one as a test set with the remaining data as a training set, and finally averages the error estimate over the multiple runs. The leave-one-out method is an extreme case of cross validation where the number of partitions equals the number of examples available. When the dataset is large, one may use the holdout method which keeps a separate test set for evaluation.

A general approach to the learning problem is to define a parameterized space $\mathcal{F}$ of classifiers that we are interested in, and subsequently find the best classifier $f^* \in \mathcal{F}$ using empirical data and some criterion c

$$f^* = \arg \min_{f \in \mathcal{F}} c(f, \mathcal{D}) \quad (1.3)$$

There are three intertwined issues: the space $\mathcal{F}$, the criterion c , and the optimization problem. These choices are made by the algorithm designer using knowledge about

the problem. Not only do they influence the error rate, they also determine the learning time and recognition time, of which typically the latter is critical for practical application. The art of designing a learning algorithm is to balance the various factors for the task at hand including accuracy, learning time and recognition time.

One class of classifiers that minimize the generalization error is the Bayes decision rule

$$f(x) = \begin{cases} +1 & \text{if } P(y = +1|x) > P(y = -1|x), \\ -1 & \text{otherwise.} \end{cases} \quad (1.4)$$

provided a perfect estimation of the posterior probability $P(y|x)$. One may first estimate the class conditional probability $P(x|y)$. For this purpose, various parametric models, e.g. Gaussian distribution, or nonparametric approaches, e.g. Parzen window method, can be used. The class conditional probability $P(x|y)$ together with the prior frequency $P(y)$ form the decision rule, which is the basis of the so-called generative approach where one tries to build models to explain the data. An alternative approach is discriminative modeling where one focuses only on the decision boundary by estimating the posterior probability $P(y|x)$ directly, for example logistic regression or neural networks [61].

Independence is an important notion in data modeling. The reason is that if two variables v_1 and v_2 are independent of one another, their joint probability distribution can be factorized, $P(v_1, v_2) = P(v_1)P(v_2)$. This reduces the computation and the amount of data needed to acquire a reasonable estimate for the density function enormously. As a result, in practice one wishes to explore the independence among variables as much as possible. In many cases, certain independence are assumed to make the computation feasible. The Bayesian network offers a tool for representing the independence among variables explicitly. The simplest case yields the popular naive Bayes classifier where all input variables are independent. More complex Bayesian networks include the various tree-based classifiers. The main challenge in Bayesian network modeling is to learn the independence from examples in a tractable manner.

State-of-the-art learning algorithms include support vector learning, the related kernel approaches, and one-class learning where the training data of one class is unavailable. These methods offer a variety of nonlinear decision boundaries efficiently via implicit feature mappings. Nevertheless, solving the optimization problem associated with these approaches is prohibited for very large datasets.

Bagging [10] and boosting [28] represent a special class of meta-learning algorithms. These methods convert a set of weakly performing base classifiers into one with better performance. Their main characteristic is that they treat the base classifiers as black boxes where their parameters are not accessible. Thus, the methods are flexible and easy to use. Among the various methods, the AdaBoost algorithm [28] has received considerable attention. It forms a linear combination of base classifiers in an iterative manner, which results in a high performance combined classifiers. The research in this area focuses on combination rules, algorithms for the selection of base classifiers and combination parameters, and the generalization performance of the algorithm.

Error bounds and classifier complexity are important concepts in recent research. The Bayes decision rule is optimal only when the probability distributions are estimated correctly, which is often not the case because of limited knowledge about the underlying mechanism generating the data, the high dimensionality of the data, and the limited number of training examples. Recent techniques such as support vector learning and AdaBoost follow another direction by establishing a computable upper bound on the true error $\epsilon(f)$ and subsequently aiming to minimize this bound. Bounding the true error requires the concept of classifier complexity as in [105].

In object recognition, and in particular object detection, one often makes a distinction between two types of error, false positive $\epsilon_+(f)$ and false negative $\epsilon_-(f)$

$$\epsilon_+(f) = E_{P(x,y=-1)} \left\{ \frac{1 - yf(x)}{2} \right\} \quad (1.5)$$

$$\epsilon_-(f) = E_{P(x,y=+1)} \left\{ \frac{1 - yf(x)}{2} \right\} \quad (1.6)$$

The former is the situation where a background patch is declared a target object by the model, while the latter is where a target object is not recognized. Often there is a parameter controlling the tradeoff between these two values. This is because in many cases the training set $\{(x_n, y_n)\}$ is not drawn randomly from $P(x, y)$, but rather from each conditional distribution $P(x|y = +1)$ and $P(x|y = -1)$. This is also the condition of many datasets studied in this thesis.

The goal of this thesis is to develop techniques with high accuracy. We are also interested in statistical models that require a small amount of training data, because collecting object examples is a tedious and time-consuming work. Moreover, we consider the ease of training an important aspect, which implies few parameter tuning. Finally, the training time and the recognition time are critical in practice, especially the latter.

1.3 Approaches to object recognition

The theory of machine learning and pattern recognition lays the foundation for learning object models from examples for the purpose of recognition. To exploit results in this field, one has to represent the images of object efficiently and effectively. The importance of a proper representation cannot be underestimated.

Let us assume the object images are normalized to a window of fixed size of $M \times N$. In the direct approach, each object instance is represented by a vector in an $M \times N$ -dimensional feature space of pixel intensity values. Subsequently, several techniques have been applied in this feature space to learn object models, including mixture-of-Gaussians [94], neural network [77], and support vector machine [62]. The drawback of this representation is the high dimensionality of the feature space, which requires lots of data before finding a stable solution. It also results in computational difficulty in both learning time and recognition time. In these approaches, the relationships among pixels are implicit in the statistical model. In contrast, we will develop a

method whereby the relationships are explicitly modeled. Furthermore, we restrict the complexity to a class of relationships that can be learned efficiently.

Linear transformations have been widely applied for feature extraction in object recognition, for example principle component analysis (PCA) [59, 100] and linear discriminant analysis (LDA) [6]. These methods find a new, data-dependent coordinate system for the representation. They render a condensation of each data point to a few coefficients. One may also employ a nonlinear transformation such as kernel-PCA [87] where the nonlinearity is handled by a technique from support vector learning. The shortcoming of these representations is that they do not consider the spatial coherence principle, namely nearby pixels are strongly correlated. We will develop in this thesis a new representation consisting of fragments of local image evidence that exploit this correlation between nearby pixels.

Another approach is to extract high-level features instead of using pixel values directly. In [86], the authors use wavelet features from subregions in the $M \times N$ window in a naive Bayes classifier. The features are manually selected, and hence the training is not straightforward, especially when one tries to apply the method for a new object class. With respect to spatial relationships, the naive Bayes classifier ignores relations between its input variables, and hence inducing a loss of discriminative power. The model we introduce in this thesis address the lack of relationships in this simple statistical model.

Selecting a small subset with high performance from a large feature set automatically for learning is a hard problem. While the general problem of automatic feature selection is still open, the AdaBoost algorithm offers an approach that selects features and learns a linear classifier at the same time. In [107], Viola and Jones exploit this to train a cascade classifier from very simple features that can be computed efficiently. The cascade classifier is specifically designed for the problem where the class prior frequencies are extremely imbalanced. The resulting system is the state-of-the-art in terms of recognition speed. The issue in this approach is that the construction of the cascade classifier has many free parameters, and hence is not easy to train. In this thesis, we develop an improvement to the AdaBoost algorithm that provides a wider class of configuration than the current linear combination while maintaining the classification time.

A common shortcoming of the above approaches is that the spatial relationships among object parts are implicit in a statistical model. This hinders the incorporation of *a priori* information about object structure and parts that might be available. Moreover, implicit spatial relationships do not help in understanding the result given by the recognition model.

In the study of spatial relationships for object recognition, in [95] the authors derive a metric between two Voronoi diagrams to describe spatial relationships of similar patches. However, such approach requires a very robust part-labeling scheme. It is too brittle for the effects of partial occlusion and undetected parts. In this thesis, we search for methods robust to such conditions.

Graph matching [12] provides another, more frequently explored path to describe the spatial relations of object parts. In this approach, the construction of object graphs is critical. The method in [75], for example, resorts to elaborate feature

extractors together with manual interactive design. In [26], the layout description of an object are hand-coded in a graph of relations between blobs. An important class of methods is the shock graph approach capturing the object topological structure [88, 89] where the extraction of the object graph and subsequent matching can be achieved efficiently. The critical issue in current graph-based representations is that they lack a statistical model that captures the variation of a collection of objects (or graphs). This thesis aims at techniques where the variation in structures can be modeled. Furthermore, we wish to learn the hidden important relationships rather than manual construction.

The part-based approach to object recognition refers to the class of methods where the object parts are first detected, then grouped to form objects according to an explicit spatial relationship among parts. In [57], the authors impose geometrical constraints on each object part in the form of a center location with tolerance in the scale and spatial domain. This type of geometrical constraints, however, does not model the statistical variation of the spatial relationship among parts. The work of Burl and Perona [13] is representative for the class of methods modeling the statistically deformable configuration to describe object structure. The method makes use of a shape space, obtained from original space by normalizing for translation, rotation and scale [20]. There are two main issues which have not been fully addressed. First, one has to solve the problem of object part design. Second, it is important to handle missing and false detection of object parts in the combination algorithm. The first issue is a parallel track of research where for example [108] shows some promising results. In this thesis, we will consider the second problem.

It can be concluded that the spatial relationships are in most cases implicit in the statistical models, supporting rigid object structures only. The part-based approach deals with spatial relations explicitly, providing an intuitive recognition model. However, the difficulties in object part design in handling of missing parts and false detection remain. The aim of this thesis is to exploit the spatial relationships in both rigid and flexible structures for recognition.

1.4 Problem statement

This thesis studies spatial relationships within object appearance. This is about combining pieces of local image evidence for recognition. This framework is intuitive as in the human parlance objects are often described by their parts. Hence, spatial relationships are an essential components of things being objects. Description with an emphasis on spatial relationships implies that not all pixels are equally important. Therefore, the study of spatial relations can also offer computational advantages by exploiting the sparsity of object description. For the same reason, it may enhance the discriminative power of existing statistical models by adding relations as features. To this end, we need to examine the different types of entities involved in the object appearance. We also need to consider how their relations can contribute to the recognition. Finally, a mechanism to identify the entities and relations best suited for recognition from training examples is required.

We explore new approaches between two ends of the spectrum; the first one is rigid and implicit relationships as in traditional statistical pattern recognition while the other one is non-rigid and explicit relationships with relations as feature. The principle component analysis technique used by Murase and Nayar [59] for object recognition is a representative example of rigid and implicit spatial relations. We observe that it does not enhance the strong spatial correlation of nearby pixels. It will result in inefficiency in object description. And, it does not support incremental learning as it requires re-training of all existing object models. Finally, it is not efficient for object localization due to the repeated computation of the representation at each image location.

We aim to specify the spatial relationships encoded in a statistical model explicitly to understand the role they play in recognition. The search is for models where the relations can be learned from examples efficiently rather than fixed *a priori* and with highly discriminative power.

The naive Bayes classifier of part-based features employed by Schneiderman and Kanade [86] is an example where relations between variables are ignored for computational simplicity. This results in loss of discriminative power. In contrast, neural networks conserve spatial relations better but they typically require manual building of the relations.

Within the class of part-based methods that offer non-rigid and explicit spatial relationships, the current solution of Burl and Perona [13] to the combinatoric problem of handling missing parts and false alarms is ad hoc and its optimality is not guaranteed.

We aim for techniques with high recognition accuracy. We concentrate on the ease of training, implying few parameter tuning. In addition, computational aspects - the training time and the recognition time - are important in our consideration throughout the thesis for practical purposes.

In chapter 2, we discuss spatial relationships in the form of statistical dependencies among pixel intensities. Here, we need to add spatial relationships to increase the power of the statistical model while maintaining the computational feasibility of learning. Chapter 3 addresses the problem of combining detected object parts into a characteristic spatial arrangement. The main difficulty is in dealing with a combinatoric problem resulting from missing and false detection of object parts. Chapter 4 contrasts two types of spatial relationships that can be learned from training examples, one presented in chapter 2 describing fine-grain pairwise relationships between pixels while the other presented in chapter 3 employing coarse-grain deformable relationships among object parts. In chapter 5, we investigate the problem of finding a characteristic arrangement of various fragments of local image evidence for object description. The issue here lays in the computational difficulty involved in selecting the suited local image fragments as well as in recognition. Finally, chapter 6 studies a wider class of configuration than existing linear boosting. This can also be seen as learning the spatial relationships among local image signs when individual base classifiers take their inputs from local image regions. We address a problem inherited from boosting algorithms, namely the lack of direct access to the parameters of the base classifiers.

Chapter 2

Face detection by aggregated Bayesian network classifiers

2.1 Introduction

Face detection is an important step in any automatic face recognition system. Given an image of arbitrary size, the task is to detect the presence of any human face appearing in the image. Detection is a challenging task since human faces may appear in different scales, orientations (in-plane rotations), and with different head poses (out-of-plane rotations). The imaging conditions, including illumination direction and shadow, also affect the appearance of human faces. Moreover, human faces are non-rigid objects, as there are variations due to varying facial expressions. Presence of other devices such as glasses is another source of variation. Facial attributes such as make-up, wet skin, hairs and beards also contribute substantially to the variation of facial appearance. In addition, the appearance differences among races, and between male and female are considerable. A successful face detection system should be able to handle the multiple sources of variation.

A large number of face detection methods have been proposed in literature. Face detection methods can be broadly divided into: model-based detection, feature-based detection and appearance-based detection.

In the model-based approach, various types of facial attributes such as the eyes, the nose and the corner of the mouth are detected by a deformable geometrical model. By grouping the facial attributes based on their known geometrical relationships, faces are detected [48, 113]. A drawback of this approach is the detection of facial attributes is not reliable [48], which leads to systems that are not robust against varying facial expressions and presence of other devices. This approach is better suited for facial expression recognition as opposed to face detection.

Among the feature-based approach, the most obvious feature is color. It is a rather surprising finding that the human skin color falls into a small range in different color spaces regardless of race. Many researchers have taken advantage of this fact in their

approach to the problem [112, 109]. Typically, regions with skin color are segmented to form face candidates. Candidates are further verified on the bases of the geometric face model. We choose not to use color information in this paper. It is partly because of the lack of a common color test set to evaluate different methods.

In the appearance-based approach, human faces are treated as a pattern directly in terms of pixel intensities [94, 77]. A window of fixed size $N \times M$ is scanned over the image to find faces. The system may search for faces at multiple image scales by iteratively scaling down the image with some factor. At the core of the system is a classifier discriminating faces from non-face patterns. Each intensity in the window is one dimension in the $N \times M$ feature space. The appearance-based methods are often more robust than model-based or featured-based methods because various sources of variations can be handled by their presence in the training set.

This paper presents a face detection system in the appearance-based approach. The class/non-class classification problem needs to be addressed because it is not possible to obtain a representative set of non-face patterns for training. Furthermore, because of the manifold of sources of variation, a complex decision boundary is anticipated. In addition, the classification methods should have a very low false positive rate since the number of non-face patterns tested is normally much higher than that of face patterns. Also due to a large number of patterns which need to be tested, a fast classification step is desirable.

The paper is organized as follows. The next section gives an overview of appearance-based classification methods. The construction of an aggregated classifier is described in section 2.3. Section 2.4 presents a new classification method using forest-structured Bayesian networks. The face detection system is described in section 2.5. Experimental results are given in section 2.6.

2.2 Literature on appearance-based face detection

It is the classification method and the type of features that characterize different appearance-based face detection systems. Many techniques from statistical pattern recognition have been applied to distinguish between faces and non-face patterns. The one-class classification problem can be solved by designing a mapping which concentrates the one class into a point while mapping the other, the non-class as widely spread over the feature space as possible as proposed by E. S. Gelsema et al. in [46].

Let $X = (X_1, X_2, \dots, X_n)$ be a random variable denoting patterns spanning the $n = N \times M$ -dimensional vector space $\mathcal{R}$. Let $x = (x_1, x_2, \dots, x_n)$ be an instantiation of X . In addition, let $Y = \{0, 1\}$ be the set of class labels, face and non-face respectively. Furthermore, let the two class conditional probability distribution be $P_0(X)$ and $P_1(X)$. Once both $P_0(X)$ and $P_1(X)$ are estimated, the Bayes decision rule [21] may be used to classify a new pattern:

$$\varphi(x) = \begin{cases} 0 & \text{if } \log \frac{P_0(x)}{P_1(x)} \geq \lambda \\ 1 & \text{otherwise} \end{cases} \quad (2.1)$$

where λ is an estimation of the log-ratio of the prior probability of the two classes. When it is not possible to obtain such approximation, one may assume equal class prior probabilities, that is $\lambda = 0$. This leads to the maximum likelihood decision rule. This leaves the question how to learn $P_y(X)$ effectively. Generally, it is not possible to estimate $P_1(X)$. This problem is often ignored. For the current face detection problem and the current dataset, we still form both $P_0(X)$ and $P_1(X)$. As a consequence, the solution will be specific for this dataset only. When the dataset grows large (we have 160,000 random patches), it will in the end be representative for the class of non-faces.

Moghaddam and Pentland [56] use principle component analysis to estimate the class conditional density. The vector space $\mathcal{R}$ is transformed into principle subspace E spanned by the V eigenvectors corresponding to the V largest eigenvalues and its complement $\bar{E}$ composed of the remaining eigenvectors. The authors show that in case of a Gaussian distribution, $P_y(X)$ can be approximated using the V components in the subspace E only. In case $P_y(X)$ cannot be adequately modeled using a single Gaussian, a mixture-of-Gaussians model can be used. A drawback of this method is that no guidelines are given to determine the number of dimension V . In addition, as each pattern is projected on to a subspace before classification, a matrix multiplication is involved. This is not desirable when the classification time is an important factor.

Sung and Poggio [94] present a face detection system which models $P_0(X)$ and $P_1(X)$, each by six Gaussian clusters. To classify a new pattern, a vector of distances between the pattern and the model's 12 clusters is computed, then fed into a standard multilayer perceptron network classifier. A preprocessing step is applied before classification to compensate for sources of image variation. It includes illumination gradient correction and histogram equalization. A shortcoming of this method is that there is no rule for selecting the number of Gaussian clusters.

The paper by Rowley et al. [77] is representative for a larger class of papers considering neural networks for face detection. A retinally connected neural network is used. There are three types of hidden units aiming at detecting different facial attributes that might be important for face detection. The network has a single, real-valued output. The preprocessing step in [94] is adopted. The system performs well on the CMU test set [77].

The naive Bayes classifier is used in [86]. Each pattern window is decomposed into overlapping subregions. The subregions are assumed statistically independent. Hence, $P_y(X)$ can be computed as:

$$\begin{aligned} P_y(X) &= P_y(\{R_i, P_i\}_{i=1}^{N_r}) \\ &= \prod_{i=1}^{N_r} P_y(R_i, P_i) \end{aligned} \quad (2.2)$$

for $y \in \{0, 1\}$. R_i is the subregion of X at location P_i and N_r is the number of subregions. The method has the power of emphasizing distinctive parts and encoding geometrical relations of a face, and hence contains elements of a model-based approach as well. A drawback of this method is the strong independence assumption. This

might not lead to high classification accuracy because of the inherent dependency among overlapping subregions.

Colmenarez and Huang [15] use first order Markov processes to model the face and non-face distributions:

$$P_y(X|S) = P_y(X_{S_1}) \prod_{i=2}^n P_y(X_{S_i}|X_{S_{i-1}}) \quad (2.3)$$

for $y \in \{0, 1\}$. S is some permutation of $(1, \dots, n)$ and used as a list of indices. The learning procedure searches for an S_m maximizing the Kullback-Leiber divergence between the two distributions $D(P_0(X)||P_1(X))$:

$$S_m = \arg \max_S D(P_0(X|S)||P_1(X|S)) \quad (2.4)$$

where $D(P_0(X)||P_1(X))$ is defined as:

$$D(P_0(X)||P_1(X)) = \sum_{x \in R} P_0(x) \log \frac{P_0(x)}{P_1(x)} \quad (2.5)$$

The Kullback-Leiber divergence is a non-negative value and equals 0 only when the two distributions are identical. The Kullback-Leiber divergence is a measure of the discriminative power between the probability distributions of the two classes [45]. By maximizing this measure, it is expected that a high classification accuracy can be achieved. The maximization problem, in this case, is equivalent to the traveling salesman problem [35]. An heuristic algorithm is applied to find an approximate solution. An advantage of this approach is that both training and classification steps are very fast.

Osuna et al. [63] apply support vector machines [105] to the face detection problem, which aims at maximizing the margin between classes. In order to train a large data set with vector support, a decomposition algorithm is proposed, in which a subset of the original data set is used. It is then updated iteratively to train the classifier.

One common characteristic of all methods is that they try to capture the decision boundary by the model supported by their classifiers. However, for classes with multiple sources of variation such as human faces, the decision boundary can be very complex. This might lead to poor accuracy performance for methods that can model simple decision boundaries. It might also lead to complex classifiers with a slow classification step. Hence, there is a need for a method which can model a complex decision boundary while allowing fast classification.

2.3 Data space exploitation and aggregated classifiers

In this section, we present a method which handles a complex decision boundary by using multiple classifiers in aggregation. Aggregated classifiers allow a natural way for solving the class/non-class classification problem.

A general learning set $\mathcal{L}$ consists of data $\{(y^t, x^t), t = 1, \dots, T\}$ where the x 's are the patterns and the y 's are their corresponding classes. The learning set is used to form a classifier $\varphi(x|\mathcal{L})$, that is the class of a new pattern x is determined by $\varphi(x|\mathcal{L})$. Let $\{\mathcal{L}_k; k = 1, \dots, K\}$ denote the K data sets to be created in order. Let φ_i denote the aggregated classifier formed by using $\{\mathcal{L}_j; j = 1, \dots, i\}$ for $i = 1, \dots, K$. The procedure for creating the data set is as follows:

1. Consider a set of face patterns $\mathcal{L}^a$. In addition, initially a set of non-face patterns $\mathcal{L}_1^{\bar{a}}$ is created by selecting randomly from a set of images containing no human faces. $\mathcal{L}^a$ and $\mathcal{L}_1^{\bar{a}}$ together form $\mathcal{L}_1$:

$$\mathcal{L}_1 = \mathcal{L}^a \cup \mathcal{L}_1^{\bar{a}} \quad (2.6)$$

2. For $i = 2, \dots, K$, apply the face detection system using the aggregated classifier φ_{i-1} on a set of images containing no human faces. False positives returned form a set of non-face patterns $\mathcal{L}_i^{\bar{a}}$. Apparently, these cases are hard cases for classifier φ_{i-1} . This set $\mathcal{L}_i^{\bar{a}}$ and the training set of face patterns $\mathcal{L}^a$ form $\mathcal{L}_i$:

$$\mathcal{L}_i = \mathcal{L}^a \cup \mathcal{L}_i^{\bar{a}} \quad (2.7)$$

The number of classifiers K may be selected according to the desired classification accuracy. Because of our selection of learning sets, if any component classifier returns a non-face decision, the pattern is classified as non-face.

We argue that this technique is suited for the face detection problem. A complex decision boundary caused by the manifold of variation is modeled by using multiple classifiers. Each has different level of difficulty of separating the two classes. Each component classifier need not be very complex, which could allow a fast classification step. In addition, the fact that a non-face pattern can be rejected at any level improves the classification time because of the normally large number of non-face patterns. Significantly, since the same face patterns, $\mathcal{L}^a$, are used for training, the true positive rate does not degrade multiplicatively as the number of component classifiers increases. Also, because the non-face patterns are generated in a bootstrap fashion, it is expected that the false positive rate decreases multiplicatively. This allows a very low false positive rate.

2.4 Forest-structured Bayesian network classifier

In this section a new classification method for the two-class problem is described. The method is in the same spirit as the Markov process-based method in [15]. However, forest-structured Bayesian networks are used to model the joint probability distribution of each class instead of Markov processes. We use this method in an aggregated classifier because it has a fast classification step.

Bayesian network is an efficient tool to model the joint distribution of variables [66]. The joint distribution $P_y(X_1, \dots, X_n)$ can be expressed using a forest structured

Bayesian network as follows:

$$P_y(x) = \prod_{i=1}^n P_y(X_i = x_i | \Pi_i = \pi_i) \quad (2.8)$$

for $y \in \{0, 1\}$. Π_i denote the parent of X_i in the network structure. $P_y(X_i = x_i | \Pi_i = \pi_i)$ are estimated from the training data $\mathcal{L}_i$ (eq. (2.6) or (2.7)). Figure 2.1 illustrates a forest structured Bayesian network modeling the joint distribution of six random variables $\{X_1, \dots, X_6\}$.

Figure 2.1: $P(X_1, \dots, X_6) = P(X_1)P(X_2|X_1)P(X_3)P(X_4|X_3)P(X_5|X_4)P(X_6|X_4)$. A dependency model of six random variables with a forest structured Bayesian network.

We search for a network structure that maximizes the Kullback-Leiber divergence eq. (2.5) between the two joint distributions.

The Kullback-Leiber divergence between two distributions in eq. (2.8) can be obtained as:

$$\begin{aligned}
 D(P_0(X) || P_1(X)) &= \sum_x P_0(x) \log \prod_{i=1}^n \frac{P_0(x_i | \pi_i)}{P_1(x_i | \pi_i)} \\
 &= \sum_{i=1}^n \sum_x P_0(x) \log \frac{P_0(x_i | \pi_i)}{P_1(x_i | \pi_i)} \\
 &= \sum_{i=1}^n \sum_{x_i} \sum_{\pi_i} P_0(x_i, \pi_i) \log \frac{P_0(x_i | \pi_i)}{P_1(x_i | \pi_i)} \quad (2.9)
 \end{aligned}$$

We show that the problem of maximizing eq. (2.9) is equivalent to the maximum branching problem [96]. In the maximum branching problem, a branching B of a directed graph G is a set of arcs such that:

1. if (x_1, y_1) and (x_2, y_2) are distinct arcs of B then $y_1 \neq y_2$.
2. B does not contain a cycle.

Given a real value $c(v, w)$ defined for each arc of G , a maximum branching of G is a branching such that $\sum_{(v, w) \in B} c(v, w)$ is maximum. It can be seen that maximizing

$D(P_0(X)||P_1(X))$ is equivalent to finding a maximum branching of a weighted directed graph constructed from the complete graph with node x_i 's plus a node x_0 with an arc from x_0 to all other nodes. $W(i, j) = \sum_{x_i} \sum_{x_j} P_0(x_i, x_j) \log \frac{P_0(x_i|x_j)}{P_1(x_i|x_j)}$ is the weight associated with each arc in the graph. There are algorithms for solving the maximum branching problem in low order polynomial time [96].

To classify a pattern x , the Bayes decision rule eq. (2.1) is used. Similar to the method in [15], fast classification of a pattern can be achieved by constructing a table for all possible values of a variable and its parent. By using eq. (2.8), the log likelihood value in eq. (2.1) becomes:

$$\begin{aligned} \log \frac{P_0(x)}{P_1(x)} &= \log \frac{\prod_{i=1}^n P_0(x_i|\pi_i)}{\prod_{i=1}^n P_1(x_i|\pi_i)} \\ &= \sum_{i=1}^n \log \frac{P_0(x_i|\pi_i)}{P_1(x_i|\pi_i)} \end{aligned} \quad (2.10)$$

Once all possible values of $\log \frac{P_0(X_i|\Pi_i)}{P_1(X_i|\Pi_i)}$ for all i are computed, the classification of a new pattern can be carried out with only n additions. This allows a very fast classification step.

2.5 Face detection system

The architecture of the system is adopted from [77]. A window of size 20×20 is scanned over each image location to find face patterns. The size 20×20 is selected because it is large enough to capture details of human faces, while allowing a reasonable classification time. The system searches the input image at multiple scales by iteratively scaling down the image with a scale step of 20% until the image size is less than the window size.

Sources of variation are captured in the training set: illumination and shadows, facial expressions, glasses, make-up, hairs, beards, races and sexes. Limited orientation and head pose, namely frontal faces and near-frontal faces, are present.

We adopt two preprocessing operations from [94]: illumination gradient correction and histogram equalization. The former reduces the effect of heavy shadows and the latter normalizes the illumination contrast of the image. Finally, each pattern is quantized to six levels of gray values to enable the estimation of the discrete probabilities. Figure 2.4 shows the quantized patterns from figure 2.3.

An aggregated classifier consisting of three Bayesian network classifiers, i.e. φ_3 , is used to classify faces and non-face patterns. The number 3 was selected based on the tradeoff between the false positive rate and true positive rate (see figure 2.6). For $K > 3$, the true positive rate is low for the detection task.

A postprocessing step is carried out to eliminate overlapping detections. When overlapping occurs, a straightforward approach would be to select the window having the largest log likelihood value. This generates sparse maxima, of which most are false positives as is observed in [77], that is most faces are detected at multiple

positions nearby in place or in scale. We have repeated the experiment and arrived at the same conclusion. For each detected location, if the number of detections within a predefined neighborhood is less than a threshold, the location is rejected.

2.5.1 Data for training

For the purpose of this paper, a set of 1112 face examples was gathered from the Internet without selection. Color images were converted to gray-scale images. Figure 2.2 gives 30 randomly selected face examples. The dataset is split into two subsets at random: 1000 faces examples are used to create the training set and 112 used to create the test set. Thirty face patterns of size 20×20 are extracted from each original face examples by rotating the images about their center points by one random less than 10 degree, scaling by one random value selected from the interval 0.9 and 1.1, translating by one random value less than 0.5 pixel, and mirroring as in [77]. Figure 2.3 illustrates 30 face patterns generated from one face example. In total, 33360 face patterns were created.

A set of 929 images containing no faces was also collected from the Internet. 360000 non-face patterns are extracted from the images by randomly selecting a square from an image and subsampling it to patterns of size 20×20 . Figure 2.5 contains 30 non-face patterns. From the next level downwards, non-face patterns were generated as described in section 2.3.

The dataset of 33360 face and 360000 non-face patterns is split into two subsets at random: the training set consists of 30000 face and 160000 non-face patterns, and the

Figure 2.2: 30 of 1112 randomly selected face examples.

Figure 2.3: An example of all 30 face patterns generated from each face example, yielding 30000 patterns to train the system.

Figure 2.4: Quantization to six levels of gray values of the patterns shown in figure 2.3. Note the preservation of geometrical layout after gray value normalization.

Figure 2.5: 30 non-face patterns randomly selected from the set of 160000 non-face patterns.

test set consists of 3360 face and 200000 non-face patterns. This test set is referred to as the pattern test set, $\mathcal{L}^T$. The face patterns of the two subsets were generated from two separate sets of face examples.

2.6 Experimental results

2.6.1 Experiment with the number of component classifiers K

Figure 2.6 shows the receiver operating characteristic curves for the four aggregated classifiers φ_1 , φ_2 , φ_3 and φ_4 on the pattern test set $\mathcal{L}^T$. At a low false positive rate an aggregated classifier with higher value of K achieves higher true positive rate. However, saturation occurs with $K > 1$, i.e. it is not possible to achieve higher true positive rate even at high false positive rate. This is due to the fact that the true positive rate of the previous levels puts an upper bound on the achievable rate of the next level.

2.6.2 Experiment with the Bayesian network classifier

Figure 2.7 shows the Receiver Operating Characteristic (ROC) curves of the three different classifiers on the pattern test set $\mathcal{L}^T$: the Markov process classifier [15], the naive Bayes classifier [21] and our method, the Bayesian network classifier φ_1 . Our method outperforms both the Markov process classifier and the naive Bayes classifier.

Figure 2.6: The Receiver Operating Characteristic (ROC) curves for φ_1 , φ_2 , φ_3 and φ_4 on the pattern test set $\mathcal{L}^T$.

Figure 2.7: The Receiver Operating Characteristic (ROC) curves of three classifiers: the Markov process classifier [15], the naive Bayes classifier [21] and the Bayesian network classifier φ_1 .

As an aside, it is interesting to see that the Markov process classifier performs better than the naive Bayes classifier only when the false positive rate is smaller than 6%.

2.6.3 Experiment on the full image test set

The system is evaluated using the CMU test set [77]. This test set consists of 130 images with a total of 507 frontal faces, including images of the MIT test set [94]. The images were collected from the Internet, scanned from photographs and newspaper pictures, and digitized from broadcast television. There is a wide range of variation in image quality. It should be noted that some authors report their results on a test set excluding 5 images of line draw faces [86], which leaves this test set with 125 images with 483 labeled faces only. We use the groundtruth with 507 faces as in [77].

Our system	MFs	Rate	FDs	System in [77]	MFs	Rate	FDs
φ_3	47	90.7%	264	NN, System 5	48	90.5%	570
				NN, System 6	42	91.7%	506
				NN, System 7	49	90.3%	440
				NN, System 8	42	91.7%	484

Table 2.1: Evaluation of the performance of the aggregated Bayesian networks, φ_3 , as compared to the neural network, NN [77] on the CMU test set [77]. The criteria are: the number of missed faces (MFs), the true detection rate (Rate) and the number of false detects (FDs).

Table 2.1 shows the performance of our face detection system in comparison with systems in [77] on the CMU test sets. It can be seen that with an equivalent detection rate, Bayesian network based method gives about half the number of false detections in comparison with the neural network method [77]. Figure 2.8 illustrates the detection result on some images of the CMU test set.

2.7 Discussion and conclusion

In this paper we have considered the face detection task as a representative of the class/non-class classification problem where the class is subjected to many sources of variation. The sources of variation include position of the face relative to the camera, illumination condition, non-rigid characteristic of the face, and presence of other devices. The appearance variation is also caused by differences among races, and between male and female. In addition, the classification method must have a very low false positive rate and a fast classification step.

Our face detection system performs well. On the CMU test set it achieves detection rate of about 90% with an acceptable number of false alarms. In comparison with other methods, our classification method using Bayesian networks outperforms related methods (namely the Markov process method [15] and the naive Bayes classifier [21],

Figure 2.8: Output of the system on some images of the CMU test set [77]. **MFs** is the number of missed faces and **FDs** is the number of false detections.

as shown in figure 2.7). On the CMU test set, our system performs better than the neural network method [77]. Our system gives about half the number of false alarms at an equivalent detection rate (see table 2.1).

Approximately half of the missed detections are caused by rotated angles (see figure 2.8, image D). Large in-plane rotation or out-of-plane rotation are not handled with this method. When the subject has the intention of looking into the camera, false negatives are rare. In fact, the missed detection in image D is one of the very few cases. Poor image conditions, such as low brightness and strong shadows, account for about one third of the missed detections (see the three examples in image E). In order to resolve this a special image enhancement preprocessing step might help. The remaining missed detections are caused by various reasons including the sizes of the faces being too small. Among the false positives, in 30 cases out of 264, the patches do appear as human faces (see the false alarm in image E and the top two false alarms in image F). Other cases might be eliminated by further postprocessing. Given the large number of tested windows [77], our method makes only one incorrect classification out of each 300000 tests.

Phase		Parameter	Value
learning	1	number of patterns per one example	30
	2	random rotating range (degree)	uniform over $[-10, +10]$
	3	random scaling range	uniform over $[0.9, 1.1]$
	4	random translating range (pixel)	uniform over $[-0.5, 0.5]$
learning & runtime	5	window size (pixel)	20×20
	6	scale step	20%
	7	number of discrete levels	6
runtime	8	K , number of classifiers	3
	9	threshold per classifier	0

Table 2.2: System parameters and their values.

Table 2.2 summarizes the parameters used by our system. The number of patterns created from one example (parameter 1) becomes saturated, that is the performance of the system does not improve by increasing this value. All parameters from 2 to 6 are identical to [77] to permit a fair comparison.

As concerns parameter 7, because our method uses a memory-based histogram for probability density estimation, there is a limitation on the number of discrete levels to be used. During the training process, at 6 discrete levels, each histogram takes up 44 Megabytes of memory. At 8 discrete levels, each histogram would take up about 78 Megabytes. Discretization causes loss of information, but does not necessarily reduce the classification accuracy. With higher number of discrete levels, more training data are needed to characterize the distributions. Furthermore, we still can distinguish face patterns from non-face patterns at 6 discrete gray values. An experiment with 4 discrete levels (data not shown) indicates a slightly degraded performance. For the purpose of this paper, 6-level discretization is appropriate. A higher number of levels might improve the performance of the system.

As concerns parameter 8, setting the value of K to 4 does not help because the true positive rate is low, see figure 2.6. Regarding the last parameter, the thresholds

of all classifiers are set to 0. Nevertheless, the threshold of the last classifier may vary, reflecting the tradeoff between true positive and false positive rate of the system.

Our system makes use of the symmetry property of the human face only implicitly by the mirroring operation on the training face examples. It is interesting to investigate how symmetry can be encoded in the Bayesian network prior to the learning phase. It is important to note, however, that structural biases and lighting may affect the symmetry property.

In conclusion, this paper presents a face detection system using an aggregation of Bayesian network classifiers. The use of an aggregated classifier is well suited for the class/non-class classification problem in the visual domain, where a complex decision boundary is anticipated due to many sources of variation. In addition, aggregated classifiers allow a very low false positive rate and fast detection.

Chapter 3

Statistical strategy for object class recognition using part detectors

3.1 Introduction

A major problem in computer vision is object recognition as the accidental circumstances of the scene and the recording dominate the appearance of the object. Apart from the variability of the object, the variability of the light source including its position and viewpoint, occlusion, noise and clutter require careful consideration. Despite advances of systems recognizing specific objects such as [100], their use in real world applications is still limited, as these methods effectively require standardized recording circumstances. Recently, the problem of recognition of object class has received considerable attention because it is useful in a wide range of applications such as image retrieval, surveillance and robot navigation.

This chapter investigates the recognition of object class using detectors of object parts. In this approach the object parts are first detected, then grouped to form objects according to an explicit global spatial relationship among parts [13]. This approach aims to overcome two problems of global description as of appearance-based methods [94, 77]. First, a single template is not able to represent many object classes without the inclusion of background where one wishes to detect objects and object parts without the accidental background. Experimental study in [1] has showed that the background affects adversely the performance of appearance-based recognition in a general context. Second, a global template allows only rather fixed structures.

The idea of object recognition by parts is not new. Grimson [37] provides a complete account for methods that use geometrical constraints for describing the configuration. Nevertheless, the use of geometrical constraints among parts is not suited for describing the global variability of object configuration. Recently, developments in statistical shape theory have proved to be useful in modeling the global config-

uration [9, 41, 16, 34, 42]. The work of Burl and Perona [13] is representative for the class of methods using these types of global deformable configuration to describe object structure. The method makes use of a shape space, obtained from original space by normalizing for translation, rotation and scale, in which the distribution can be expressed in a closed form, when the original space is Gaussian [20]. However, there is a difficulty in estimating the parameters of the distribution in their shape space. Another drawback of this method is that there is no guarantee that the search procedure will lead to an optimal solution.

We identify three issues in learning and recognition by parts. The first task requires determination of the object parts and their detectors. In the second task one has to provide a model to integrate the detected part results into a general configuration. Finally, the recognition problem requires an efficient strategy to locate the object in an image once object parts have been detected.

We do not address the first issue in this chapter. Automatic part selection is a parallel track of research where for example [108] shows some promising results. For learning part detectors, one could make use of various techniques in the appearance-based approach.

We focus on the modeling and recognition problem given the part detectors. The difficulty lies in the fact that the scene is cluttered, possibly occluded, and part detectors are not reliable. Therefore, we will consider the issue of false and missed detection of object parts. In addition, unlike appearance-based approach where it is possible to conduct an exhaustive search in the space of all possible candidates, the hypothesis space in recognition by parts explodes very quickly.

We propose a Bayesian approach to integrate the distinctiveness of the object global configuration and the presence of its parts in a complex scene. The Bayesian formulation has been used for the object recognition problem in [92, 51, 93, 3, 110]. The advantage of Bayesian analysis is that prior knowledge can be incorporated into the evaluation procedure. Our work differs from previous work in the model we propose for the problem of recognition by parts. After normalized for translation, the configuration of an object class is modeled as multivariate Gaussian distribution. The performance of the part detectors is used to determine the relative weight of the parts. Both the configuration and the weights are combined into a single Bayesian objective function to rank object hypotheses. Based on this criterion, we derive the necessary heuristics to use in an A^* search algorithm [78] to find the optimal solution.

For the optimization of the posterior, stochastic optimization algorithms such as simulated annealing and sampling have been successfully used by [32, 54, 93]. This class of algorithms is useful, but generally does not yet satisfy the computation time constraint [114]. A gradient descent method can be used to obtain a local solution [92]. For the specific probabilistic model in [31], the authors present a tree search procedure to optimize the posterior. This model is further studied in [114] in relation with the A^* search algorithm. Nevertheless, to use the A^* algorithm with a probabilistic model, one has to derive appropriate heuristics for that specific model. In this respect, our solution to the problem of optimizing the posterior is novel.

The organization of the chapter is as follows. In section 3.2 we present a Bayes formulation of the problem. In section 3.3 we derive the objective function and dis-

cuss the estimation of its parameters. The subsequent section is devoted to the search method for our objective function. Section 3.5 presents our experimental results on both synthetic and real data. Finally, we discuss various issues for further investigation and conclude the chapter in section 3.6.

3.2 Bayes formulation of the problem

Consider an object model consisting of a set of p parts, with a spatial configuration x in some configuration space $\mathcal{X}$, here by locations on the plane of the parts. Thus, x is a point in a $2p$ -dimensional space. Note that the components of x are ordered because the parts are labeled.

Let $\{d_\alpha | \alpha = 1, \dots, p\}$ be a set of detectors corresponding to the set of parts. When d_α is applied to an image I in some image space $\mathcal{I}$, the response $d_\alpha(I)$ is a binary image, indicating any detected presence of part α . Let $d_\alpha^v(I)$ denote the response at location $v \in I$, a value of 1 for the presence of part and 0 otherwise. We assume that by local maximization of the response the detector provides one location of α in a small neighborhood. In other words, each hit in $d_\alpha(I)$ corresponds to one detected part α .

We choose to use the maximum a posteriori (MAP) estimate in deriving the objective function. Given an image $I \in \mathcal{I}$, one would like to find the “most probable” object. This can be formulated as

$$\begin{aligned} x^* &= \arg \max_x P(x|I) \\ &= \arg \max_x P(I|x)P(x) \end{aligned} \quad (3.1)$$

It is often convenient to consider the log form of eq. (3.1)

$$x^* = \arg \max_x \{\log(P(I|x)) + \log(P(x))\} \quad (3.2)$$

In other words, $\log(P(I|x)) + \log(P(x))$ is used as an objective function to evaluate a hypothesis x .

In this model, $P(I|x)$ is the likelihood measure that an image I is observed given the presence of an object at some specified location x . The estimation of this measure is related to statistics of natural images $P(I)$, where a set of filters is often used. In that case, $P(I)$ is characterized by the distribution (or histogram) of the filter responses [115, 36]. The method described in [93] also estimates the image (observation) likelihood given an object by means of filter banks. These filter banks aim at capturing the distribution of responses of both background and foreground. In section 3.3.1, we present our estimation method when only the performance statistics of the part detectors is available.

The prior density distribution $P(x)$ reflects our knowledge about the object. As an example, suppose x consists of two components $x = (x_S, x_T)$, denoting a translation-normalized shape x_S and a translation vector to position the object in the image x_T . Suppose that the two components are independent. Thus, one can write $P(x) =$

$P(x_S)P(x_T)$. We model the prior knowledge about the object location via $P(x_T)$, which is useful for datasets where objects of interest have a high or low probability of presence in some region. In this chapter, we assume a uniform distribution for $P(x_T)$. In section 3.3.2, we present a translation invariant density distribution for the global configuration, which does not assume a Gaussian distribution for the original space as in [13]. This enables simple estimation of model parameters. The scale and rotation invariant properties are not dealt with. When the scale and rotation of the object are involved, one needs to address those aspects not only for shape but also for part detectors. This presents a significant additional complexity.

Finally, we address the optimization problem of eq. (3.1). This problem is difficult because of a large search space. In section 3.4, we formulate the problem as a graph search problem. We then present an *admissible* heuristic required to use the A^* search algorithm to find the optimal solution.

In summary, we are concerned with the two terms in eq. (3.2) and the optimization problem. In the context of this chapter we need to approximate the first term with the information extracted from the part detectors. The second term should be modeled in such away that captures the distinctiveness of the object global configuration. Finally, the optimization method should be efficient because a very large search space is expected.

3.3 Learning an object model

We present the likelihood and the prior terms of the model in eq. (3.2). We then discuss the estimation of model parameters from training examples.

3.3.1 The likelihood

In this section, we estimate the likelihood $P(I|x)$ that the image I is observed, given an observation of object x . We use the performance statistics of part detectors for our estimation. For the purpose of this chapter, assuming the part detectors are independent, we have

$$P(I|x) = \prod_{1 \leq \alpha \leq p} P_\alpha(d_\alpha(I)|\mathbf{x}_\alpha) \quad (3.3)$$

where $d_\alpha(I)$ is the binary response image and $\mathbf{x}_\alpha$ is the vector of the true location of part α in x .

This assumption is not realistic because object parts might bear great similarity, e.g. the left eye and the right eye of human faces. Despite of this fact, the assumption is needed to reduce computational complexity.

For each part α and its detector d_α , let γ_α be the detection rate, that is $\gamma_\alpha = P(d_\alpha^v(I) = 1|\mathbf{v} = \mathbf{x}_\alpha)$ where $\mathbf{v}$ is a two-dimensional vector denoting an image location. Thus, $P(d_\alpha^v(I) = 0|\mathbf{v} = \mathbf{x}_\alpha) = 1 - \gamma_\alpha$. Furthermore, let β_α be the probability that a background location is detected as part (or false alarm rate), that is $\beta_\alpha = P(d_\alpha^v(I) = 1|\mathbf{v} \neq \mathbf{x}_\alpha)$. Thus, $P(d_\alpha^v(I) = 0|\mathbf{v} \neq \mathbf{x}_\alpha) = 1 - \beta_\alpha$. In addition, let d_0 and d_1 denote

the number of non-responses and responses, respectively. Hence, $d_0 + d_1$ is the size of image I .

We make a further assumption that the responses are independent given the true location of parts. Again, this assumption is not strictly true, because object parts are correlated with neighboring locations. Nevertheless, this assumption reduces the model complexity greatly. The conditional probability distribution $P_\alpha(\cdot|\cdot)$ in eq. (3.3) can be expressed as

$$\begin{aligned}
P_\alpha(d_\alpha(I)|\mathbf{x}_\alpha) &= P(d_\alpha^v(I)|\mathbf{v} = \mathbf{x}_\alpha) \prod_{\mathbf{v} \neq \mathbf{x}_\alpha} P(d_\alpha^v(I)|\mathbf{v} \neq \mathbf{x}_\alpha) \\
&= \gamma_\alpha^{d_\alpha^{\mathbf{x}_\alpha}(I)} (1 - \gamma_\alpha)^{1 - d_\alpha^{\mathbf{x}_\alpha}(I)} (1 - \beta_\alpha)^{d_0 - (1 - d_\alpha^{\mathbf{x}_\alpha}(I))} \beta_\alpha^{d_1 - d_\alpha^{\mathbf{x}_\alpha}(I)} \\
&= \left(\frac{\gamma_\alpha(1 - \beta_\alpha)}{(1 - \gamma_\alpha)\beta_\alpha} \right)^{d_\alpha^{\mathbf{x}_\alpha}(I)} (1 - \gamma_\alpha)(1 - \beta_\alpha)^{d_0 - 1} \beta_\alpha^{d_1} \\
&= \left(\frac{\gamma_\alpha(1 - \beta_\alpha)}{(1 - \gamma_\alpha)\beta_\alpha} \right)^{d_\alpha^{\mathbf{x}_\alpha}(I)} c_\alpha
\end{aligned} \tag{3.4}$$

where c_α is a value that does not depends on x . Substituting (3.4) into (3.3), we have

$$P(I|x) = \prod_{1 \leq \alpha \leq p} \left(\frac{\gamma_\alpha(1 - \beta_\alpha)}{(1 - \gamma_\alpha)\beta_\alpha} \right)^{d_\alpha^{\mathbf{x}_\alpha}(I)} c_\alpha \tag{3.5}$$

and in the log form

$$\log(P(I|x)) = c + \sum_{1 \leq \alpha \leq p} d_\alpha^{\mathbf{x}_\alpha}(I) \log \left(\frac{\gamma_\alpha(1 - \beta_\alpha)}{(1 - \gamma_\alpha)\beta_\alpha} \right) \tag{3.6}$$

where c is a value that does not depend on x .

An intuitive interpretation of Eq. (3.6) is that a bonus value of $\log \left(\frac{\gamma_\alpha(1 - \beta_\alpha)}{(1 - \gamma_\alpha)\beta_\alpha} \right)$ is awarded if part α is detected. For the method to bias towards detected parts, this value should be greater than zero, which is equivalent to $\gamma_\alpha > \beta_\alpha$. Informally, the detection rate should be greater than the probability that a background location is detected as part. It is often the case that γ_α and β_α are functions of some threshold t_α . Choosing an appropriate (or optimal) value for the threshold t_α is an interesting problem. In this chapter we do not consider this question, as well as the issue regarding the quality of parts and detectors, which is reflected in γ_α and β_α .

In short, eq. (3.6) is our approximation of the image likelihood given an object. When used in the objective function to evaluate different values of x , the constant c can be ignored.

3.3.2 The prior

This section presents a model for the object configuration. First we present a translation invariant model by moving the object location to a new coordinate centered at an object part. The chosen object part is called the mapping point. We then address the problem when this mapping point is not detected. Finally, we present our solution to the problem of missed detection of other non-mapping points in the configuration.

Translation invariance

Consider p labeled points $(X_i, Y_i), X_i, Y_i \in \mathcal{R}, i = 1, \dots, p$. A random $2p$ -vector $\mathbf{Z}$ can be constructed

$$\mathbf{Z} = (X_1, \dots, X_p, Y_1, \dots, Y_p)'$$

A new vector $\mathbf{Z}^*$ can be obtained from $\mathbf{Z}$ by mapping (X_1, Y_1) to the origin

$$\mathbf{Z}^* = (X_2 - X_1, \dots, X_p - X_1, Y_2 - Y_1, \dots, Y_p - Y_1)' \quad (3.7)$$

We assume that the configuration after translation normalization $\mathbf{Z}^*$ is jointly Gaussian, denoted by $\mathbf{Z}^* \sim \mathcal{N}_{2p-2}(\boldsymbol{\mu}^*, \Sigma)$. We can approximate the density distribution of $\mathbf{Z}$ by $\mathbf{Z}^*$. Effectively, we have a translation invariant configuration space. The density of $\mathbf{Z}^*$ is given by

$$P(\mathbf{Z}^*) = (2\pi)^{-\frac{1}{2}(2p-2)} |\Sigma|^{-\frac{1}{2}} \exp \left[-\frac{1}{2} (\mathbf{Z}^* - \boldsymbol{\mu}^*)' \Sigma^{-1} (\mathbf{Z}^* - \boldsymbol{\mu}^*) \right] \quad (3.8)$$

and in the log form

$$\log(P(\mathbf{Z}^*)) = k + \left[-\frac{1}{2} (\mathbf{Z}^* - \boldsymbol{\mu}^*)' \Sigma^{-1} (\mathbf{Z}^* - \boldsymbol{\mu}^*) \right] \quad (3.9)$$

where k is a value that does not depend on x .

First, we review two properties of a multivariate Gaussian distribution that will be needed.

Lemma 1 *Let $\mathbf{t}$ be an ℓ -vector and $\mathbf{t} \sim \mathcal{N}_\ell(\boldsymbol{\theta}, \Sigma)$. Then for any $r \times \ell$ matrix C , $C\mathbf{t} \sim \mathcal{N}_r(C\boldsymbol{\theta}, C\Sigma C')$.*

Consult [91] for the proof.

Lemma 2 *Let $\mathbf{t} = (\mathbf{t}_1', \mathbf{t}_2')' \sim \mathcal{N}_\ell(\boldsymbol{\theta}, \Sigma)$, that is vector $\mathbf{t}$ is decomposed into $\mathbf{t}_1$ and $\mathbf{t}_2$. Let $\mathbf{t}_1$ and $\mathbf{t}_2$ be r - and $(\ell - r)$ -vectors respectively. Suppose the corresponding partition of $\boldsymbol{\theta}$ and Σ are respectively given by*

$$\boldsymbol{\theta} = \begin{pmatrix} \boldsymbol{\theta}_1 \\ \boldsymbol{\theta}_2 \end{pmatrix} \quad \text{and} \quad \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{12}' & \Sigma_{22} \end{pmatrix}$$

where $\boldsymbol{\theta}_1$ is an r -vector and Σ_{11} is an $r \times r$ matrix. Then the conditional distribution $P(\mathbf{t}_1 | \mathbf{t}_2)$ is

$$\mathcal{N}_r(\boldsymbol{\theta}_1 + \Sigma_{12} \Sigma_{22}^{-1} (\mathbf{t}_2 - \boldsymbol{\theta}_2), \Sigma_{1 \cdot 2}),$$

where

$$\Sigma_{1 \cdot 2} = \Sigma_{11} - \Sigma_{12} \Sigma_{22}^{-1} \Sigma_{12}'$$

Again, consult [91] for the proof.

So far we have considered the case where all parts are detected. When some parts are missed, we still need to approximate the density. A special case is when the part used as mapping point is missed. First, we show that $P(\mathbf{Z}^*)$ can be computed when a different point is used as mapping point. Then we describe a procedure for estimating $P(\mathbf{Z}^*)$ when missing values occur.

Mapping point invariance

Consider a situation when point (X_i, Y_i) , $i \neq 1$, is mapped to the origin instead of (X_1, Y_1) . This is needed in case where the first part is not detected. In that case, a new $(2p - 2)$ -vector can be constructed

$$\mathbf{Z}_i^* = (X_1 - X_i, \dots, Y_1 - Y_i, \dots)' \quad (3.10)$$

Vector $\mathbf{Z}_i^*$ can be obtained from $\mathbf{Z}^*$ via a linear transformation, that is $\mathbf{Z}_i^* = L_i \mathbf{Z}^*$, where the transformation matrix L_i is defined as follows. The entries in the column $(i - 1)$ of L_i are (-1) 's. In every other column j , there is only one value of 1 at row $(j + 1)$ for $j < (i - 1)$, and at row j for $j > (i - 1)$. All other entries are 0. An example of a transformation matrix of size 8 is given below

$$L_4^{(8)} = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Therefore, by lemma 1, $\mathbf{Z}_i^* \sim \mathcal{N}_{2p-2}(L_i \boldsymbol{\mu}^*, L_i \Sigma L_i')$. Moreover, $P(\mathbf{Z}^*)$ is proportional to $P(\mathbf{Z}_i^*)$ with a factor equal the absolute value of $\det(L_i)$ which is 1 (the sub-matrix obtained by the elimination of the first row and column $(i - 1)$ is an identity matrix). Thus, $P(\mathbf{Z}^*) = P(\mathbf{Z}_i^*)$.

Approximation under missing parts

When a set of parts is left undetected, we assume that the missing parts are located at their best locations for the sake of the global configuration. Note that given a fixed state of detected and non-detected parts, the likelihood term of eq. (3.2) does not change. Therefore, we can maximize $P(\mathbf{Z}^*)$.

When the first part is detected, an obvious choice for the mapping point is (X_1, Y_1) . Otherwise, because our computation does not depend on the choice of the mapping point, we can map any detected point to the origin. For generality, let (X_o, Y_o) be the mapping point. This results in a vector $\mathbf{Z}_o^*$ of $(2p - 2)$ dimensions. This is a random vector distributed according to a multivariate Gaussian distribution $\mathbf{Z}_o^* \sim \mathcal{N}_{2p-2}(L_o \boldsymbol{\mu}^*, L_o \Sigma L_o')$, where L_o is given in 3.3.2.

We can decompose $\mathbf{Z}_o^*$ into two parts $\mathbf{t}_1$ and $\mathbf{t}_2$ as in lemma 2, where $\mathbf{t}_2$ denotes the observed components and $\mathbf{t}_1$ denotes the missing parts. Further, we have $P(\mathbf{Z}_o^*) = P(\mathbf{t}_2)P(\mathbf{t}_1|\mathbf{t}_2)$. Maximizing $P(\mathbf{Z}_o^*)$ is therefore equivalent to maximizing $P(\mathbf{t}_1|\mathbf{t}_2)$. According to lemma 2, $P(\mathbf{t}_1|\mathbf{t}_2)$ is also a multivariate Gaussian distribution; therefore $P(\mathbf{t}_1|\mathbf{t}_2)$ is maximal when $\mathbf{t}_1$ equals the mean of the distribution, which is specified in lemma 2. Given the value of $\mathbf{t}_1$, $P(\mathbf{Z}_o^*)$ can be computed easily, which is also the value of $P(\mathbf{Z}^*)$.

3.3.3 Parameter estimation

The final form of the objective function $f(x)$ is obtained by substituting eq. (3.6) and eq. (3.9) into eq. (3.2), ignoring the constant values

$$f(x) = \sum_{1 \leq \alpha \leq p} d_{\alpha}^{x_{\alpha}}(I) \log \left(\frac{\gamma_{\alpha}(1 - \beta_{\alpha})}{(1 - \gamma_{\alpha})\beta_{\alpha}} \right) + \left[-\frac{1}{2}(\mathbf{Z}^* - \boldsymbol{\mu}^*)' \Sigma^{-1}(\mathbf{Z}^* - \boldsymbol{\mu}^*) \right] \quad (3.11)$$

where $\mathbf{Z}^*$ is obtained from x as in eq. (3.7). Learning the object model involves the estimation of the parameters $\{\gamma_{\alpha}, \mu_{\alpha} | 1 \leq \alpha \leq p\}$, $\boldsymbol{\mu}^*$ and Σ .

In our framework, the training data is manually labeled for object parts. From the labeled points, we extract image patches to serve as examples for object parts. In addition, we extract randomly patches from the background to serve as non-part examples.

The value of γ_{α} and β_{α} can be estimated by applying the detector d_{α} on the training set, then computing the detection rate and false alarm rate for part α . Alternatively, we can define a certain class of detectors, and subsequently train detector d_{α} using the training examples for part α . In this case, the values of γ_{α} and β_{α} can be estimated in various ways depending on the number of training examples available. One can employ the hold-out method by splitting the dataset further into training set and test set. Another method is n -fold cross validation where the dataset is divided into n approximately equal partitions. The training is carried out n times; each time $n - 1$ different partitions are used for training the one partition is used for testing. The final estimation is the average of the estimation of the n runs.

The parameters for the Gaussian distribution $\boldsymbol{\mu}^*$ and Σ are estimated from the locations of the labeled points. The maximum likelihood estimator can be used. The properties of this estimator can be found in [91]. In case there are missing points in the training examples, one can make use of the expectation maximization algorithm [18, 80].

3.4 Searching for the optimal solution

Given eq. (3.11) for the objective function, the next topic is how to find the optimal solution efficiently. We first review briefly the A^* search algorithm [78]. We then formulate the problem as a graph search problem, and subsequently establish the necessary heuristic to use the A^* search algorithm to find the optimal solution.

The A^* search is an algorithm to find an optimal path from a start vertex A to a destination vertex B in a graph. We consider the maximum score problem. At each vertex C , the algorithm estimates the score $f(C)$ of the path from A to B going through C . The score $f(C)$ is the sum of the score of the real path from A to C and an estimated score going from C to B (see figure 3.1). The algorithm stores all explored sub-paths in a priority queue, and iteratively expands the sub-path with the highest estimated score $f(C)$. The first path reaching B is the solution of the algorithm. It is proved that if the estimated score (heuristic) is always greater than the true score,

Figure 3.1: At each vertex C , an estimate $f(C)$ of the score going from A to B through C guides the exploration of the A^* search algorithm.

Figure 3.2: Search graph. M_α is the number of part α detected, for $1 \leq \alpha \leq p$.

the algorithm will lead to an optimal solution. The heuristic in that case is called an *admissible* heuristic. The problem is to devise a heuristic for a particular problem.

First we formulate the optimization problem as a graph search problem. The graph is a directed weighted graph of $(2 + p + \sum_{1 \leq \alpha \leq p} M_\alpha)$ nodes as in figure 3.2, where M_α is the number of part α detected. There is one starting vertex and destination vertex denoted by A and B , respectively. For each object part α , there is one vertex $w_{\alpha i}$ for each detected location i , $1 \leq i \leq M_\alpha$. In addition, there is one vertex $w_{\alpha 0}$ denoting the missing value. Every path from A to B specifies an object hypothesis and is associated with a score measure. This measure is computed by the objective function (eq. 3.11). The problem is to find a path from A to B with the highest score.

There are a total of $\prod_{1 \leq \alpha \leq p} (M_\alpha + 1)$ paths from A to B . A brutal force search approach is not feasible for a large number of parts in a cluttered scene. Fortunately, we can derive an estimate of the total cost at each vertex along a path. This estimation can be used as a heuristic in an A^* search for the optimal solution.

The derivation of the estimation is as follows. Suppose we are at vertex $w_{\alpha k}$, $0 \leq k \leq M_\alpha$, after traveling path $q_{A \rightarrow w_{\alpha k}}$. The vertices in $q_{A \rightarrow w_{\alpha k}}$, except A , determine the “best” possible configuration for the remaining vertices as in 3.3.2. Further, imagine that at every “best” location in the unexplored path, there is a corresponding part. It is clear that the score f^* of this ideal path is an upper bound

for any real path from A to B with the sub-path $q_{A \rightarrow w_{\alpha k}}$. Thus, this heuristic is admissible.

The A^* search will explore the vertex with highest f^* value. This strategy guarantees that the first path exploring B is an optimal path.

3.5 Experiments

3.5.1 On synthetic data

The purpose of this experiment is to show the working of the objective function as well as the efficiency of the search procedure.

The performance of the A^* search is often judged by the number of sub-paths that are in the search queue when the solution is reached. A small value for this number indicates that the solution is reached quickly [78]. This value and the total number of possible paths (of a brutal force search) will be used to evaluate the algorithm.

First, we create 10000 samples of human figures. Each object consists of 11 parts. The creation of objects is as follows. Basically, each object forms a tree, as shown in figure 3.3. We fix the root of the tree, and the subsequent nodes are generated according to pre-defined Gaussian distributions conditional on the previous node. Figure 3.3 shows fifteen examples of these figures.

Each object is then placed on an image of size 300×300 at a random position. For each pair of image and object, we generate two types of noise as discussed in section 3.3.1. For each part, the number of false alarms is drawn from a Poisson distribution with a pre-defined mean.

The search method in section 3.4 is applied to find an object in each image. An example of the part detected image is given in figure 3.4(a). The corresponding search result is shown in figure 3.4(b). The method is able to locate objects in case of missing parts and with the presence of clutter. When missing parts occur, the method also approximates their locations.

The number of sub-paths that are in the search queue when the solution is reached is very small in comparison with the total number of paths. This shows an excellent performance of the search algorithm.

3.5.2 On real images

We use the Caltech face dataset [108]. The dataset consists of 450 images, each containing a human face in a clutter background with various lighting conditions. Upon examination of the dataset, we remove nine images (from image 328 up to 336) due to the small scale and three hand-draw images (image 400, 402 and 403). These images are significantly different from the rest of the dataset. In total, there are 438 images of 28 people. The faces appear at different locations in each image with a variety of facial expressions. The original images are converted into grayscale images and downsampled by half to a resolution of 448×296 pixels.

We split the dataset into two partitions. The training set contains 216 images of 14 people. The test set contains 222 images of the remaining 14 people. For each

Figure 3.3: Fifteen examples of a human figure; each is an object consisting of 11 parts.

Figure 3.4: An example where the object is correctly located in case of two missing parts denoted by asterisks (*). (a) Part detection results. Parts are denoted by letters from a to k. The total number of possible paths is close to 10^{14} . (b) Search result. The total number of sub-paths in the queue when the optimal solution is reached is 2437 only.

training image, we manually labeled five object parts: the two eyes, the nose, and the two corners of the mouth, each by one mouse click. The size of each part is 32×32 pixels. There is no evidence why this decomposition is optimal, but for the purpose of this chapter, the choice is sufficient.

We use a simple form of object part detectors, namely template matching with

Figure 3.5: Templates of the five selected parts of a human face. The templates are arranged according to the average configuration.

Figure 3.6: ROC curves for the five part detectors on the training sets.

normalized cross correlation [73]. Given an input template u , the normalized cross correlation ρ between object template t_α and u is defined as

$$\rho(t_\alpha, u) = \frac{\sum_{\mathbf{i}} (t_\alpha(\mathbf{i}) - \bar{t}_\alpha)(u(\mathbf{i}) - \bar{u})}{\left(\sum_{\mathbf{i}} (t_\alpha(\mathbf{i}) - \bar{t}_\alpha)^2 \sum_{\mathbf{i}} (u(\mathbf{i}) - \bar{u})^2 \right)^{1/2}} \quad (3.12)$$

where $\mathbf{i}$ denotes the two-dimensional index vector, $t_\alpha(\mathbf{i})$ and $u(\mathbf{i})$ denote the pixel values of the templates at image location $\mathbf{i}$, and $\bar{t}_\alpha$ and $\bar{u}$ denote the average pixel values of the two templates. The template t_α for each part α is the average of the training examples aligned by the manually selected centers. Figure 3.5 shows the five templates obtained, as arranged according to the mean of the configuration model. An object part is detected by applying its detector at all image locations, followed by a non-max suppression operation on the responses.

A threshold value is needed to determine whether an image patch u is classified as object part α . In this experiment, we set the false alarm rate for each detector at approximately 10^{-2} . To estimate the false alarm rate, we sample uniformly 20000 patches of size 32×32 from the training images. The false alarm rate is rather

Figure 3.7: Example of the detection results. The search algorithm is able to locate the object in a cluttered scene. It also works in case missed detection occurs. The asterisks (*) indicate the likely location of the missed parts. The letters from a to e denote object parts (see the legend in figure 3.6).

unimportant for the purpose of this chapter since we want to show that the method works for various detectors and various uncertainties of the part detection. (We note that we expect a correlation in the performance of the two eye detectors, distorting the independence assumption.)

Figure 3.6 shows performance curves of the five detectors with varying values of the thresholds. It can be seen that the simple eye detectors are more reliable than the other equally simple detectors. The difference between the performance of the left and right mouth corner detectors is an illustration of the fact that the detectors are unstable.

Figure 3.7 shows the detection result of four images in the test set. The faces are correctly localized in a cluttered scene and there is some missed detection. Overall, the method obtains a correct localization rate of 92% on the training set and 91% on the test set. Figure 3.8 shows two examples where the faces are not localized correctly. In general, mistake happens in cases where the detection of many parts fails due to non-standard lighting conditions. Again, this is not the purpose of this chapter. This chapter aims to lay down a method for spatially combining classifiers.

It is difficult to compare this result with the result of Burl and Perona [13] because they use a different dataset. They obtain 94% correct localization on a dataset of 130 images of one person. However, on the training set with a variety of individuals, the result is only 63%. Note that our test set consists of 14 individuals and is different from the training set.

On average, the search space consists of 556848 hypotheses, but the search reaches

Figure 3.8: Example of incorrect object localization. In both cases the part detectors fail to find a reasonable number of correct parts. In the image on the left only the left eye (denoted by letter b) is located correctly. In the image on the right, only the nose (denoted by letter c) is located correctly.

the optimal solution when there are about 360 sub-paths in the queue only. This translates into approximately half of a second on a 1 GHz CPU with a simple Matlab implementation. The main computational cost is the detection of the five object parts, which take a few seconds.

3.6 Discussion and conclusion

In this chapter we consider the recognition of an object once its parts have been detected. The problem is how to encode the global configuration of an object and the presence of its parts in a cluttered scene into an objective function in order to evaluate object hypotheses. In addition, an efficient search procedure is required for finding the best hypothesis according to the objective function. This is because a potentially large search space is anticipated.

We propose an objective function in the Bayesian framework. This criterion is capable of encoding the global configuration, as well as information of the cluttered scene using the performance statistics of the part detectors. An important characteristic of the proposed objective function is that it allows a fast search algorithm, namely the A^* search. By deriving an upper bound on the final value of the objective function, we effectively achieve an *admissible* heuristic for the A^* search, which guarantees an optimal solution.

We performed experiments on both synthetic and real data. The experimental results show an excellent performance of the proposed heuristic for the search problem. The high recognition rate achieved also indicates that the objective function is suitable model for the problem of recognition by parts.

Our recognition system is placed in a unified probabilistic framework. The probabilistic model shows an interesting link between the recognition of objects and performance of part detectors, detection algorithms and the cluttered scene. The future research will focus on the recognition performance with respect to the performance of the part detectors. This includes investigation into the independence assumption

among part detectors, the use of likelihood values returned by detectors when available, and the combination of part detection and the statistics of natural images.

For a robust recognition system, the problems of self-occlusion and natural occlusion need to be addressed, together with the study of various scene accidental conditions. We also leave that for future investigation.

In conclusion, we propose a probabilistic framework for the object class recognition problem. The two important components of this framework are the objective function and the search method. The objective function is derived in a Bayesian approach, handling of the uncertainty of the part detection. It is able to encode the object global configuration as well as the presence of clutter. In addition, a heuristic is devised for the A^* search, which achieves an optimal solution for recognition.

Chapter 4

Learning spatial relations in object recognition

4.1 Introduction

Object is an abstract concept for which just few aspects are valid in general, for all types of objects. In the recognition of object classes, objects are usually characterized by distinct appearances (signs), textures (tiger), features (coins) or parts (eyes plus mouth for a face). In this chapter, we concentrate on the other common aspect of object: that they share some spatial order. We focus on learning spatial relationships among in the object appearance or among object parts for the purpose of object recognition.

In the study of spatial relationships one may take a syntactical approach. Tagare et al. [95] derives a metric between two Voronoi diagrams to describe spatial relationships of similar patches. This is similar to the approach in [22] where they admit a relaxation of the spatial constraint. At any rate, such approaches require a very robust part-labeling scheme. They are too brittle for the effects of partial occlusion and undetected parts that the method is not effective for object recognition.

Graph matching [12] provides another, more frequently explored path to describe the spatial relations of object parts. The construction of object graphs is not obvious. The method in [75], for example, resorts to elaborate feature extractors together with manual interactive design. A separate class are the shock graphs capturing the object topological structure [88, 89]. This is an important class because the extraction and subsequent matching can be achieved efficiently. Nevertheless, the shock graph as well as graph representation in general lack a statistical model that can capture the variation of a collection of objects (or graphs). As a consequence, these approaches are not yet suitable for learning and recognition of object classes.

In this chapter we oppose two types of spatial relationships that can be learned from a set of examples for object recognition. The first uses deformable relationships between named entities, for examples the approaches in [13, 108, 24, 25, 23]. The

named entities are derived by specific part detectors, as opposed to Harris detectors and spot detectors generating nameless points of interest. Using parts is advised when one wishes to extend the class of objects with a fixed spatial layout to include objects with articulated structures. Examples are windmills and animals with their limbs. One critical issue not fully addressed in these approaches is that of missing parts and image clutter. In this study, we choose a method using a translation invariant shape configuration [67] where the problem of missing parts and image clutter is handled efficiently.

The second type of spatial relations we study are between pairs of intensity values on the face of the object. For that purpose, the spatial relation of a pair of pixels is combined with pixel intensities at either end. Such is the topic of “graphical models”, which utilize results in graph theory and probability theory into a single framework [66, 40]. Unlike the aforementioned graph matching approach where each object is associated with a separate graph, in this approach a graph describes a class of objects. Since the class of an object can be derived easily by holding it against the graphical model, this approach is suited for the object class recognition problem. Graphical models also differ from other statistical approaches that have been applied to the object recognition problem, such as [94, 47, 17], in that the spatial relationships among input pixels, are described explicitly. One critical issue in this approach is how to learn the structure from examples. Here we consider the graphical model in [68] that looks for the co-occurrence of pixel values over distance. This model is learned efficiently and shows superior performance [68] in comparison with methods employing simpler structures [86, 15].

We contrast the recognition performance of a model for coarse-grain spatial relationship between parts [67] with two models for fine-grain spatial relation between pixels [68]. We do so on two data sets, one being human faces in frontal view as the best example of the class of objects with a fixed spatial layout, and the other horses faced to the right with large variations in action and hence in the relative positioning of the limbs. The last data set is taken as the best example of the articulated object class. We do not aim to reach for optimal performance but rather to understand the behavior of spatial relation classifiers.

A proper comparison requires careful consideration of the amount of a priori knowledge inserted in each method. Ideally, the a priori information should be equal for all different methods. We consider annotation as sending information to the learning algorithm. One mouse click in the annotation is one message to the learning algorithm. From that point of view, we take care that the amount clicks is (almost) identical for all methods. For the learning phases, object examples are extracted from images by one mouse click per object roughly in the center of its frame. In addition, there is a few mouse clicks on one image to initialize the part learning.

The chapter is organized as follows. After the review of the methods in the next section, we describe the datasets used for evaluation in section 4.3. The experimental results are presented in section 4.4. Finally, we draw conclusions in section 4.5.

Box 4.1: Part-based, Bayesian network and Bayesian network with object masking.

4.2 Methods

In this section, we discuss part-based learning, Bayesian network learning and Bayesian network learning with object masking. In the first method, the object of interest is decomposed into parts. The learning phase consists of learning the part detectors and the spatial relationships among them. The recognition phase consists of the detection of object parts and searching for the spatial configuration optimal for a target object. In the second method, object appearance is learned by finding optimal pairwise relations among the pixels as well as modeling the dependences of the pixel values. The runtime process includes scanning the input image to find the best object location according to the learned statistical model. The last method is a variant of the second one where an object mask is employed to reduce the effect of background pixels. Box 4.1 summarizes the methods, maximizing their similarity.

4.2.1 Part-based learning

Let the object consist of p parts. The training object examples are labeled for the part positions. That is, for each training object example j , there are p labeled points in a two-dimensional plane $(x_{\alpha,j}, y_{\alpha,j}) \in R^2$, $\alpha = 1, \dots, p$. For each object part α , a detector d_α is learned using the labeled examples extracted from the corresponding labeled points. We use the very simple template matching with normalized cross correlation [73] as a part learner. Recall that the normalized cross correlation measure ρ between part α template t_α and a template u is

$$\rho(t_\alpha, u) = \frac{\sum_{\ell} (t_\alpha(\ell) - \bar{t}_\alpha)(u(\ell) - \bar{u})}{\left(\sum_{\ell} (t_\alpha(\ell) - \bar{t}_\alpha)^2 \sum_{\ell} (u(\ell) - \bar{u})^2 \right)^{1/2}} \quad (4.1)$$

where ℓ is a two dimensional index vector, $t_\alpha(\ell)$ and $u(\ell)$ denote pixel values of the templates at image location ℓ , and $\bar{t}_\alpha$ and $\bar{u}$ denote the average pixel values of the two patches. For each detector d_α we estimate its performance on the training set in terms of the detection rate γ_α and the false alarm rate β_α . The fact that the part detector is simple and could be substituted by better ones is not of prime relevance here as we study the virtue of spatial relations among parts as detected.

The parts of one object are spatially related as modeled by a Gaussian distribution after translation normalization. From the labeled points in each object example j , we can constructed a vector $\mathbf{z}_j$ in a $(2p - 2)$ dimensional space by mapping $(x_{1,j}, y_{1,j})$ to the origin

$$\mathbf{z}_j = (x_{2,j} - x_{1,j}, \dots, x_{p,j} - x_{1,j}, y_{2,j} - y_{1,j}, \dots, y_{p,j} - y_{1,j})' \quad (4.2)$$

The normalized configuration vector $\mathbf{z}$ is modeled by a Gaussian $\mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \Sigma)$. The estimation of the parameters of the Gaussian distribution is straightforward using the maximum likelihood estimator [91].

The object localization problem is treated as a graph search problem with an objective function integrating the performance of part detectors and the object configuration. Let $\mathcal{O} = \{(h_\alpha, x_\alpha, y_\alpha) | \alpha = 1, \dots, p\}$ be an object hypothesis obtained

from an input image, where h_α is an indicator equal 0 if part α is not detected and 1 otherwise, and (x_α, y_α) is the spatial location of part α . The score for $f(\mathcal{O})$ is

$$f(\mathcal{O}) = \sum_{1 \leq \alpha \leq p} h_\alpha \log \left(\frac{\gamma_\alpha(1 - \beta_\alpha)}{(1 - \gamma_\alpha)\beta_\alpha} \right) + \left[-\frac{1}{2}(\mathbf{z} - \boldsymbol{\mu})' \Sigma^{-1}(\mathbf{z} - \boldsymbol{\mu}) \right] \quad (4.3)$$

For the object localization task, we search for the solution of

$$\mathcal{O}^* = \arg \max_{\mathcal{O}} f(\mathcal{O}) \quad (4.4)$$

An A^* search algorithm is used to obtain the optimal solution of (4.4), exploiting two special properties of a Gaussian distribution to handle the missing part problem (see [67] for detail).

For the image classification task, we find the best object hypothesis $\mathcal{O}^*$ in the input image. Consequently, the value $f(\mathcal{O}^*)$ is used as a confidence value. If $f(\mathcal{O}^*)$ is greater than a threshold value, the input image is classified as containing an object instance. Otherwise, it is declared as background.

Part alignment in learning phase

One issue with the part-based approach is that it requires annotated training data of object parts. When the training data are collected using one mouse click as in this study, one has to perform automatic part annotation.

Alignment algorithm

1. Initialize $p_{\alpha,j}$ by extracting the image patch at ℓ_α in each training example j , $j = 1, \dots, N$.
2. Repeat T times
 - (a) update part α template t_α as the mean of $p_{\alpha,j}$.

$$t_\alpha = \frac{1}{N} \sum_{j=1}^N p_{\alpha,j} \quad (4.5)$$

- (b) update part α in image j , $p_{\alpha,j}$

$$p_{\alpha,j} = \arg \min_{u \in \mathcal{N}(j, \ell_\alpha, m)} \rho(t_\alpha, u) \quad (4.6)$$

where $\mathcal{N}(j, \ell_\alpha, m)$ denotes the set of patches in the $m \times m$ neighborhood of ℓ_α of image j .

Box 4.2: The part alignment algorithm.

In order to label object parts in all training examples, we use the first example to create a map of relative part positions. Let ℓ_α denote the center position of part α . The image patch for part α in each training example j , denoted by $p_{\alpha,j}$, can fluctuate within a predefined neighborhood of size $m \times m$ of ℓ_α . Let $\mathcal{N}(j, \ell_\alpha, m)$ denote the set of all image patches in this neighborhood.

Box 4.2 presents an iterative algorithm for part alignment. In each loop, the algorithm updates the template t_α for each part α and subsequently finds in the neighborhood $\mathcal{N}(j, \ell_\alpha, m)$ in each training example j the closest image patch to t_α in terms of normalized cross correlation. The result serves as the updated annotation in each example j .

4.2.2 Bayesian network learning

The Bayesian network method [68] treats an image example as a feature vector $\mathbf{v} = [v_1, v_2, \dots, v_{n_1 \times n_2}]'$ for a resolution of $n_1 \times n_2$. Furthermore, each vector $\mathbf{v}$ is considered as an instantiation of a random variable $\mathbf{V} = [V_1, V_2, \dots, V_{n_1 \times n_2}]'$. The joint distribution for each class $P_c(\mathbf{v})$, where $c \in \{-1, +1\}$ with -1 denoting the background class and $+1$ the object class, is estimated using a forest structured Bayesian network

$$P_c(\mathbf{v}) = \prod_{i=1}^{n_1 \times n_2} P_c(V_i = v_i | \Pi_i = \pi_i) \quad (4.7)$$

where Π_i denote the parent of V_i in the network structure. The lower case notations v_i and π_i denote specific values of the corresponding random variables in upper case. Each pair (V_i, Π_i) forms a directed edge of the network.

The special characteristic of the method is that the network structure is learned from training data rather than fixed a priori. This is achieved by maximizing the Kullback-Leibler divergence between the two distributions $P_c(\mathbf{v})$. This optimization problem is equivalent to the problem of finding a maximum branching in a weighted graph where the weight $W_{(V_i, \Pi_i)}$ for each directed edge (V_i, Π_i) is

$$W_{(V_i, \Pi_i)} = \sum_{\pi_i} \sum_{v_i} P_{+1}(v_i, \pi_i) \log \frac{P_{+1}(v_i | \pi_i)}{P_{-1}(v_i | \pi_i)} \quad (4.8)$$

The maximum branching problem is a graph problem that can be solved in polynomial time [96]. Thus, learning the structure is efficient. The weight $W_{(\Pi_i, V_i)}$ indicates the discriminatory power of the relation. In other words, a larger weight contributes more to the classification of object versus background.

To classify an input pattern $\mathbf{v}$, the log likelihood is computed

$$\log \frac{P_{-1}(\mathbf{v})}{P_{+1}(\mathbf{v})} = \sum_{i=1}^{n_1 \times n_2} \log \frac{P_{-1}(v_i | \pi_i)}{P_{+1}(v_i | \pi_i)} \quad (4.9)$$

This step is involved with $n_1 \times n_2$ additions, which is equal to the number of input pixels. The values $\log\{P_{-1}(v_i | \pi_i)/P_{+1}(v_i | \pi_i)\}$ are stored in memory for efficient computation.

For the object localization problem, we search for the best candidate

$$\mathbf{v}^* = \arg \max_{\mathbf{v}} \log \frac{P_{-1}(\mathbf{v})}{P_{+1}(\mathbf{v})} \quad (4.10)$$

In this case, the Bayesian network classifier is treated as an object filter operating over the image.

Again, for the image classification task we find the optimal object hypothesis $\mathbf{v}^*$ for the input image. Subsequently, a threshold value is employed on the likelihood value of the optimal solution $\log\{P_{-1}(\mathbf{v}^*)/P_{+1}(\mathbf{v}^*)\}$ to determine whether or not the input image contains an instance of the target object.

4.2.3 Bayesian network learning with object masking

In the Bayesian network method we make no distinction between object and background. In effect, pixels and their intensities in the background area will be uncorrelated with the foreground area and hence deteriorate the result by the random noise they produce.

Instead of using a full image patch as a feature vector, an object mask is employed to better specify the object example. This extension is rather simple since we only need to rearrange the pixels under the mask into a feature vector. Specifically, the mask can be seen as a projection matrix M as follows. Let $\mathbf{v}$ be an $n_1 \times n_2$ vector as in the previous section. The matrix M is binary having r rows and $n_1 \times n_2$ columns where r is the number of elements to be selected. Each row in M has one non-zero element indicating the corresponding selected component of $\mathbf{v}$. The new feature vector $\mathbf{v}_M$ is

$$\mathbf{v}_M = M\mathbf{v} \quad (4.11)$$

The process of learning and classification is left unchanged from the standard Bayesian network method, except $\mathbf{v}_M$ is used in place of $\mathbf{v}$.

The mask is generated once for the whole dataset such that the interior of the object silhouette is sufficiently covered, masking out the remaining pixels most of which will be background pixels.

As the pixels are features input to the classifiers, masking can be seen as feature selection. In this respect, Bayesian network with object masking is closer to part-based learning in terms of a priori knowledge than standard Bayesian network learning. This is because part selection in part-based learning is also a step to include interesting features for recognition.

4.3 Datasets and annotation

We use two datasets in the experiments: human faces and horses. The former is from [108], while the latter is collected from the Internet and in part from the Corel dataset. Note that the evaluation of object localization performance requires images containing an object instance. Whereas for the evaluation of image classification performance,

apart from a set of object images, another set of background images containing no object instance is required.

4.3.1 The face dataset

This dataset contains 438 face images of 28 people (after removing nine images that are clearly different in scale from the rest of the dataset). The original images are converted to grayscale and resized by half.

The face training set consists of 216 images of 14 people in the set of face images. We manually annotate the nose of each human face in the training set. This step requires one mouse click in each image. The face examples are extracted from a frame of size 128×128 centered at the labeled points. See figure 4.1(a) for examples.

We also extract 10000 patches of the same size by sampling uniformly over the training images. See figure 4.1(b) for examples.

The face test set consists of the remaining 222 images of 14 people. Figure 4.1(c) shows six examples of this set.

For the image classification task, we use an additional set of background images also provided by the authors of [108]. The images are assorted scenes around the Caltech campus and in their Vision laboratory. There are 451 images in total. Figure 4.1(d) show five background images of this test set.

4.3.2 The horse dataset

The second dataset consists of 472 horse images. All the horses are faced to the right by flipping the images left right when needed. The training set consists of 269 images. We annotate the center of the horse in each image of the training set. Again, this step requires only one click in each image. Subsequently, training horse examples of size 160×128 are extracted. See figure 4.2(a) for examples.

In addition, we extract 10000 background patches of the same size by sampling uniformly over the training images. See figure 4.2(b) for examples.

The horse test set consists of the remaining 203 images. Figure 4.2(c) shows examples of this set.

For the image classification task we use 865 background scenery images. Figure 4.2(d) shows three background images in this test set.

It can be seen that the recognition of horses on this dataset is hard. The images are taken outdoor with different cameras, purposes and under various lighting conditions. Although the horses all face to the right, there is still large variation in pose, background, skin, rider and pace. As a consequence, the horse recognition seems hopeless for the Bayesian network methods because the structure of the horse is non-fixed. It also seems hopeless for the part-based method because we have a very simple part learner while the dataset contains considerable variation in the object parts. However, our main interest is not in high recognition results but in understanding the role of spatial relations for object classification.

(a) Training faces

(b) Training background patches

(c) Test face images

(d) Test background images

Figure 4.1: Examples of the face dataset.

Figure 4.2: Examples of the horse dataset.

4.4 Experimental results

4.4.1 Object localization

Performance of part-based learning

For the face dataset, we learn eight object parts including the two eyes, the two cheeks, the two mouth corners, the nose, and the forehead. For the horse dataset, we learn seven parts including the hip, neck (lower part and upper part), front leg (lower part and upper part) and back leg (lower part and upper part). The size of the part templates is 32×32 in both cases. Each part α is allowed to move within 15 pixels in each direction from its center position ℓ_α for the face dataset and 20 pixels for the horse dataset. Finally, we let the annotation algorithm run for 20 iterations ($T = 20$).

Figure 4.3 illustrates the state of the parts initially and as learned after 1, 5 and 20 iterations. It can be seen that the learned parts after 20 iterations are visually much clearer than their initial states for both datasets. Figure 4.4 shows the average shift of each part alignment step. The iterative algorithm achieves stable result after about five loops. The complete process takes less than 30 minutes on a 1 GHz CPU with a Matlab implementation. Note that one may use the average shift as a criterion to terminate the alignment procedure instead of fixing the number of iterations. However, one still needs to check the convergence condition of the average shift.

Figure 4.3: The state of the part alignment algorithm (a) the initial state, (b) after one iterations, (c) after five iterations and (d) after twenty iterations.

Finally, the threshold value for each detector is set such that the error rates are equal for detection and false alarm. That is, $1 - \gamma_\alpha = \beta_\alpha$ (see eq. 4.3). This is known as the equal error rate criterion.

We also performed experiments where a global object template is used for object localization. We used the same measure for matching as in our part detectors (eq. 4.1). Our purpose is to examine the performance gained by the additional complexity of the part-based method.

Figure 4.4: Average shift at each iteration, (a) on the face dataset and (b) on the horse dataset.

Template matching		Part-based
88.5%		96.5%
45.0%		63.0%

Table 4.1: Accuracy of object localization.

Table 4.1 presents the results of the part-based method together with the results of global template matching on the two datasets. Significant improvement in accuracy is achieved, 8% on the face dataset and 18% on the horse dataset, due to the tolerance to the variation in the object structures.

We can observe that the result on the horse dataset is low. This can be explained by the poor quality of the part detectors on this dataset, namely template matching with normalized cross correlation.

We carried out an experiment to compare the results between manual annotation versus maximal correlation annotation for the two eyes, nose and two mouth corners composition. The experimental result shows that the alignment algorithm performs well with 92% correct localization in comparison to the result of 91% for the manually annotated data [67]. In this particular case, the algorithm reduces 80% of the amount of tedious annotation work and achieves an equivalent result.

(a) Faces

(b) Horses

Figure 4.5: Example of part-based object localization on the two test sets, (a) results on the face test set, see figure 4.1(c), and (b) results on the horse test set, see figure 4.2(c). The letters denote the detected locations of object parts. The diamonds represent detected parts of the solution while the *'s represent parts not found.

Figure 4.5 shows examples of the detection result of the part-based method on the two datasets. Note that the part-based method also gives the estimated position of the missing parts. The incorrect localization examples (the second picture in the second row in figure 4.5(a) and 4.5(b)) are due to severe missing parts.

Performance of Bayesian network learning

The Bayesian networks are learned with a resolution of 32×32 for the face dataset and 40×32 for the horse dataset. Figure 4.6 shows the relations and the weights of the incoming edge in the two networks learned over the two datasets. Light pixels indicate high values for the weights, hence incoming relationships important for recognition.

For the face dataset, it can be seen that Bayesian network learning indicates that edges to the forehead and the cheeks have the most weights. That these edges depart from the eye and cheek regions (data not shown) indicates that the most important discriminating relations are found by long distance co-occurrences of forehead and eyes and cheeks. Similarly, for the horse dataset the Bayesian network method is able to learn long distance dependences among pixel values. In particular, the relations between the head region and the body region and between the body region and the front leg region are clearly shown. From figure 4.6(b) and 4.6(d) it can be learned that there is little or no contribution from the context of the object.

Table 4.2 gives the localization result of the Bayesian network method on the two

Figure 4.6: Bayesian network relations and their strengths (see eq. 4.8) as learned over the training sets, (a,b) faces and (c,d) horses.

Dataset	Localization accuracy
Faces	92.5 %
Horses	83.0 %

Table 4.2: Performance of Bayesian network learning for the object localization task on the test sets. See figures 4.1(c) and 4.2(c) for examples of these test sets.

datasets. The result on the face dataset is worse than that of the part-based method with the eight-part composition, which almost completely covers the face. The result on the horse dataset is superior to that of the part-based method. This is because Bayesian network learning can focus on the stable regions and models the dependences among them, including long distance ones.

Performance of Bayesian network learning with object masking

The Bayesian networks for the two object classes are depicted in figure 4.7. For the face dataset, about half of the pixels are masked out. For the horse dataset, approximately one third of the pixels are masked out.

Table 4.3 shows the result of Bayesian network with object masking on the two

Figure 4.7: Bayesian network relations and their strengths (see eq. 4.8) as learned over the training sets, (a,b) faces and (c,d) horses. Object masking is employed for both datasets.

Dataset	Localization accuracy
Faces	97.0 %
Horses	83.0 %

Table 4.3: Performance of Bayesian network with object masking for the object localization task on the test sets. See figures 4.1(c) and 4.2(c) for examples of these test sets.

datasets. The removal of pixels by masking is effectively a feature selection procedure as the pixel relations and pixel values corresponding to the background are no longer in the classifier equation. For those images where the background acts as noise to the object recognition, it is done to improve the recognition rate. The improvement is clearly shown for the face dataset where the correct localization rate increases from 92.5% to 97%. This effect, however, is absent in the horse dataset, where the floor of grass and sand will be highly correlated with horse images.

Note that the computational gain is significant in both cases. As can be seen from eq. (4.9), the number of operations required for each classification is equal to the number of features. Thus, the classification cost is reduced by half and one third for the face dataset and the horse dataset, respectively.

4.4.2 Image classification

In the next set of experiments, we evaluate the methods on an image classification task. The aim is to determine whether or not a test image contains an instance of

Figure 4.8: ROC curves for image classification. BN indicates the Bayesian network method and masked BN indicates Bayesian network with object masking. (a) On the face test set, see figure 4.1(c). Note that the figure is magnified around the region of equal error rates. (b) On the horse test set, see figure 4.2(c).

the target object.

Figure 4.8(a) and 4.8(b) show the ROC curves for the three methods on the face dataset and horse dataset, respectively. One can observe that for the face dataset, the Bayesian network method with object masking performs best. The Bayesian network methods perform significantly better on the horse dataset. The poor performance of the part-based method on this dataset is because the variation in appearance of object parts is not sufficiently captured by the simple part learner.

In the next experiment, we simulate object occlusion by covering a quarter of the object in back. Figures 4.9(a) and 4.9(b) give examples of synthetic occlusion for each dataset.

The ROC curves for image classification under occlusion are shown in figure 4.10, together with those curves obtained on the original datasets. The performance of all three methods drops substantially.

The part-based method appears least sensitive to the effect of occlusion. This is because the method is designed to search for the best subset of part in the desired configuration. One can also observe that the Bayesian network method with object masking is more sensitive to occlusion than without object masking. This is another sign of the feature reduction removing the redundancy by masking the background.

(a)

(b)

Figure 4.9: Examples of the test set with randomly assigned occlusions, (a) occluded faces and (b) occluded horses.

Figure 4.10: ROC curves for image classification. [occ] indicates the result on the datasets with occlusion.

In short, this suggests that feature selection in general does not help with the problem of occlusion.

4.5 Discussion and conclusion

In this chapter, we evaluate two statistical methods to evaluate the use of spatial relationships in object recognition.

We emphasize that all methods are presented with the same set training examples with (almost) identical annotation, which is one click on the center of an object in the learning set. In this way, for the purpose of fair comparison, we equate the amount of annotation information available to the methods. Due to this constraint, part alignment is indispensable. As a by-product, we conclude that the alignment works well. On the face dataset automatic alignment and manual part labeling generate the same localization results.

For the Bayesian network methods, one can observe that the long-distance dependencies within the face of the object are most informative. This is due to the spatial coherence in images implying that nearby pixels are strongly correlated almost everywhere; typically only broken at the edges between foreground objects and the background. Hence short distance relations are commonly correlated and less informative. Strongly correlated long-distance relations are rare and typical for pixel pairs on one face of an object. This observation holds true for the faces as expected for an example of the fixed layout class as well as for the horses.

Object masking is designed in analogy to a feature selection procedure for the Bayesian network. It improves the recognition result significantly on the face dataset while maintaining the result on the horse dataset. Bayesian network with object masking outperforms the part-based method on both datasets. That is, even in the

case of the flexible articulated object class, here represented by the horse dataset, the Bayesian network performed superior. The Bayesian networks are capable of capturing the stable relations in the object appearance.

It should be noted however that for the part-based method, the issue of part learners is critical for effective recognition. We do not use the most advanced part learners. One method to be investigated further is the use of a local Bayesian networks for that purpose. Alternatively, one may consider local descriptors invariant to scale and orientation as part detectors.

None of the methods copes well with severe random patch occlusion. The part-based method is less sensitive to occlusion than the Bayesian network methods. This is ascribed to the tolerance to missing parts of the part-based method. Between the two Bayesian network methods, object masking is more sensitive to occlusion than without object masking as expected since the masking brings the information back to the minimum set of pixels. This result means that feature selection by Bayesian masking is unlikely to solve the problem of occlusion. In applications where occlusion is expected, it should be dealt with explicitly.

Further research is conducted into the issue of automatic feature selection. This is important for generic object recognition where a priori knowledge is limited and the range of object classes is huge. One approach might be to reject regions with low weight in a learned Bayesian network in an iterative fashion. The experiments show that the background regions carry little weight in classification.

In conclusion, we have presented a study for three methods that exploit spatial relations for object recognition. The relations are learned from training examples. By imposing (almost) the same amount of annotation of training data, which can be seen as sending knowledge to the learning algorithm, we can identify the strengths and weaknesses of each method. The part-based method and Bayesian networks are capable of exploiting the spatial relations for recognition. To be sure, the part-based method outperforms global template matching on both datasets. (This is expected as the part-based method is designed to be tolerant to some variation in the object structure, making the method less brittle than global matching.) And, for the Bayesian network method, a previous study [68] has already shown its superior performance in comparison with the naive Bayes classifier, which encodes no spatial relations at all. This leads us to conclude that both part-based and especially pair-wise relations in Bayesian networks are superior to strictly spatial matching by templates and strictly non-spatial classifiers.

Chapter 5

Sparse representation for coarse and fine object recognition

5.1 Introduction

Automatic learning of object models from pictorial examples plays an important role in recent computer vision systems. For object manipulation and robot navigation, as well as content-based image retrieval, the ability to recognize the object view is essential.

We make a distinction between coarse and fine recognition. Coarse recognition is the ability to distinguish between different objects. For this purpose, one can construct various classifiers including boundary-based methods, for example [72]. On the other hand, fine recognition is the case where the classes are defined on a gradual or continuous scale such as aging or pose estimation. In fine recognition, discriminative approaches such as support vector classification [105], boosting [28] or one-class learning [97] cannot be applied. In this chapter, we focus on the problem of both coarse and fine recognition of object views.

This chapter presents a novel representation for object models, which is the foundation for accuracy and efficiency. The representation is built from sparse, significant details rather than complete image arrays. The representation is made up from literal view fragments. It is incrementally accurate, where one may employ degrees of accuracy depending on the task at hand. The representation is extensible in that new objects can be learned incrementally without involvement of the other objects. The representation is readily available when one has to scan over an image at multiple scales and locations in search for an object.

Learning models in the new representation is based on sparse function approximation from a large dictionary of Gaussian derivative bases. It is related to the classic Njet representation [44] at multiple locations by the truncated Taylor expansion. The

advantage of the approach is that from a rich repertoire of local signs those that best characterize the object are selected. We will show that object models can be learned efficiently from examples with the matching pursuit algorithm [53].

The new representation is related to the work of Schmid and Mohr [85], which uses combinations of Gaussian derivative filter responses invariant to certain image transformations and imaging conditions. In this method as well as others employing invariant descriptors [81] the input image is matched against known views only. On the other hand, our method provides a mechanism for matching against in-between views that are not present in the training set, and therefore is capable of fine classification. In addition, the selection of salient points to achieve sparsity in the invariant approaches is left open. In our approach, the sparsity property has a direct link to the approximation accuracy.

Another related method is presented in [69] where the author adapts the matching pursuit algorithm to learn a linear combination of second order Gaussian derivative bases for face recognition. A major drawback of this method is that all objects share the same set of bases, which is not suited where the object appearances differ sharply from one to another. In our approach, each object is projected on a different set of bases, which allows efficient representation and expansion to a variety of objects.

The chapter is organized as follows. In section 5.2 we review previous approaches and formulate the problem. Section 5.3 presents our solution to the problem. We describe the experiments with the new approach in section 5.4. Finally, section 5.5 concludes the chapter.

5.2 Formulation of the problem

The intensity images taken from different view points of an object d , $d \in \{1, \dots, D\}$, are samples of a three-dimensional function $f^{(d)}$ of intensity

$$f^{(d)}(x, y, \phi) : \mathbb{R}^3 \rightarrow \mathbb{R} \quad (5.1)$$

where x and y denote the two spatial axes while ϕ denotes the orientation axis. By sampling the orientation ϕ by P samples and the x and y direction by N samples, the set of images for object d can be represented as a matrix $F^{(d)}$

$$F^{(d)} = \begin{bmatrix} \mathbf{f}_1^{(d)} & \mathbf{f}_2^{(d)} & \dots & \mathbf{f}_P^{(d)} \end{bmatrix} \quad (5.2)$$

where the p th column of the matrix $\mathbf{f}_p^{(d)}$ is a row-wise vectorization of the pixel values of the object d at view ϕ_p , $f^{(d)}(x_i, y_j, \phi_p)$ for $i = 1, \dots, N$ and $j = 1, \dots, N$.

Let $\mathbf{f} \in \mathbb{R}^{N \times N}$ be a vector representing an input image. For fine recognition, from the training data $F^{(d)}$ one tries to learn a mapping $\varphi^{(d)}(\mathbf{f}) : \mathbb{R}^{N \times N} \rightarrow \mathbb{R}$ to estimate the pose of the input object. For coarse recognition, given a training set of D objects $\{F^{(d)}\}$, one tries to learn a mapping $\omega(\mathbf{f}) : \mathbb{R}^{N \times N} \rightarrow \{1, \dots, D\}$ to identify the object.

Figure 5.1: Examples of an object seen from eight views, courtesy of [60].

5.2.1 Related work

Poggio et al. [70] pioneer research in view-based representation using function approximation with radial basis function networks [71]. Specifically, the approximation of the function $\varphi^{(d)} : \mathbb{R}^{N \times N} \rightarrow \mathbb{R}$ is

$$\varphi^{(d)}(\mathbf{f}) = \sum_{k=1}^K \alpha_k G(\|\mathbf{f} - \bar{\mathbf{f}}_k^{(d)}\|) \quad (5.3)$$

where the K coefficients α_k and the centers $\bar{\mathbf{f}}_k^{(d)}$ are found in the learning phase and $G(\cdot)$ is an appropriate basis function, typically Gaussian [71]. In this case, the image $\mathbf{f}$ of the object is represented by the similarities to the centers $\bar{\mathbf{f}}_k^{(d)}$ which can be seen as object image prototypes. The problem with this approach is that one has to store the prototypes $\bar{\mathbf{f}}_k^{(d)}$, each of which is of the size of the input image. The coarse recognition is carried out by generating a standard view for the input image by learning one function for each pixel, similar to the pose function in eq. (5.3), and subsequently comparing the generated view with the standard views of all objects in database. This is computationally expensive as the image generation and comparison is done in the high dimensional space of $\mathbf{f}$.

There is a link between the type of basis function G and a priori information about the function to be approximated. In [71] the authors show that function $\varphi^{(d)}$ in eq. (5.3) minimizes the functional

$$H[\varphi] = \sum_{p=1}^P \ell(\phi_p - \varphi(\mathbf{f}_p^{(d)})) + \lambda \|\mathcal{P}\varphi\|^2 \quad (5.4)$$

where $\varphi(\cdot)$ is the function we search for, and ϕ_p is the true value. The function $\ell(\cdot)$ is a loss function, $\mathcal{P}\varphi$ is a constraint operator, and λ is a regularization parameter. The first term in eq. (5.4) addresses the proper fitting the data points while the second term penalizes overfitting. The regularization term reflects the a priori information about the function. For instance, when G in eq. (5.3) is a Gaussian function, it is shown in [71] that the regularization term penalizes all derivatives of φ to obtain smooth solutions. The derivation of G for other types of a priori knowledge is not straightforward.

The support vector estimation of functions also approximates functions by a linear combination of bases as in eq. (5.3) with the prototypes being a subset of the training data

$$\varphi^{(d)}(\mathbf{f}) = \sum_{p=1}^P \alpha_p G(\|\mathbf{f} - \mathbf{f}_p^{(d)}\|) \quad (5.5)$$

where $G(\cdot)$ is a kernel function. The link between various kernel functions and a priori information is established in [90] via the regularization approach as in eq. (5.4). This approximation employs the so-called ϵ -sensitive loss function ℓ

$$\ell(z) = \begin{cases} |z| & \text{if } |z| \geq \epsilon \\ 0 & \text{otherwise} \end{cases} \quad (5.6)$$

where z is the difference between the true value and the approximated value in eq. (5.4). The interesting aspect of this loss function is that the solution $\varphi^{(d)}$ of eq. (5.4) has a small number of non-zero coefficients α_p . Thus, only the corresponding support vectors $\mathbf{f}_p^{(d)}$, $\alpha_p \neq 0$, need to be kept. Similar to the radial bases function approach, one problem with the support vector method is the number of data points that have to be stored, especially when the function to be approximated is nonlinear. In addition, the families of functions provided by the support vector method are not broad enough. An example is that the kernel cannot reflect the periodic nature of orientation.

The work of Murase and Nayar [59] is representative for the class of methods that employ dimension reduction by linear projection prior to recognition. The approach uses principle component analysis (PCA), similar to the eigenface method in [100]. Specifically, let $\bar{\mathbf{f}}$ denote the average of all training vectors

$$\bar{\mathbf{f}} = \frac{1}{DP} \sum_{d,p} \mathbf{f}_p^{(d)} \quad (5.7)$$

and M denotes the projection matrix learned by PCA. The training set $F^{(d)}$ is projected to the low dimensional space spanned by the row vectors of M

$$F_{\text{PCA}}^{(d)} = M \left\{ F^{(d)} - \bar{\mathbf{f}} \mathbf{1}_P^T \right\} \quad (5.8)$$

where $\mathbf{1}_P^T$ is the transpose of a vector of P components of ones. Murase and Nayar use cubic-spline interpolation among the training points $F_{\text{PCA}}^{(d)}$ to learn smooth curves

$\Gamma^{(d)}$ parameterized the pose parameter ϕ

$$\Gamma^{(d)}(\phi) = \mathbf{a}_{p,0}^{(d)} + \mathbf{a}_{p,1}^{(d)}\phi + \mathbf{a}_{p,2}^{(d)}\phi^2 + \mathbf{a}_{p,3}^{(d)}\phi^3 \quad \text{for } \phi_p \leq \phi < \phi_{p+1} \quad (5.9)$$

The curve $\Gamma^{(d)}(\phi)$ is a vector-valued function with the number of components equal to the number of dimensions of the eigenspace. The coefficients $\mathbf{a}_{p,r}^{(d)}$ are computed by solving systems of linear equations derived from the smoothness condition at the known points, including the periodic condition.

At runtime, the input image $\mathbf{f}$ is also projected to the low dimensional space

$$\mathbf{f}_{\text{PCA}} = M \{ \mathbf{f} - \bar{\mathbf{f}} \} \quad (5.10)$$

The pose estimation function $\varphi^{(d)}(\mathbf{f})$ is defined as the pose of the closest point to the curve $\Gamma^{(d)}(\phi)$ in terms of sum of squared differences

$$\varphi^{(d)}(\mathbf{f}) = \arg \min_{\phi} \|\mathbf{f}_{\text{PCA}} - \Gamma^{(d)}(\phi)\|^2 \quad (5.11)$$

and for coarse recognition

$$\omega(\mathbf{f}) = \arg \min_d \left\{ \min_{\phi} \|\mathbf{f}_{\text{PCA}} - \Gamma^{(d)}(\phi)\|^2 \right\} \quad (5.12)$$

To solve the minimization problem in eq. (5.11), the authors sample the curve $\Gamma^{(d)}(\phi)$ densely, and subsequently search for the closest one among the sampled points. Clearly, this is not efficient in both memory storage and recognition time. The problem in eq. (5.12) is solved by minimizing eq. (5.11) for all d .

There is an alternative approach for recognition in the eigenspace using radial basis function networks [58]. However, a more fundamental drawback resides in the representation of the object data. The pooled eigenspace can only work as long as all objects share common features. For a large dataset, it may be too general to be successful. This also holds true for other techniques for linear dimension reduction, for example the so-called optimal linear projection [49]. One solution is to use object eigenspaces [59] that are specific for individual objects. However, one needs to overcome the problem of matching objects across different low-dimensional representations.

For the class of nonlinear reduction techniques such as kernel PCA [87], automatic selection of a proper nonlinear model is a hard problem in itself. For other nonlinear techniques that have been applied successful for data visualization such as local linear embedding [76] and isomap [98], both projection and reconstruction of a new data point (not in the training set) are nontrivial [7]. Hence, they are not suited for the object recognition purpose.

A major drawback of all methods described in this section is that they use the vector representation for images, which does not exploit an essential image property, namely spatial coherence allowing for condensation in the image representation to a few readily computable points. This is demonstrated by the fact that permuting the pixels in the images does not change the results. In the rest of the chapter, we present

a new approach for recognition that approximates the function $f^d(x, y, \phi)$ in eq. (5.1) directly, exploiting a rich amount of a prior information about $f^d(x, y, \phi)$, including the spatial coherence property.

5.2.2 Problem statement

We wish to approximate the function $f^{(d)}(x, y, \phi)$ from the training examples $F^{(d)}$. To this end, let $f^{(d)}(x, y, \phi)$ be parameterized by $m_d \in M$ where M denotes a family of functions to be specified. The parameter m_d can be seen as an object model for the object d .

The problem of learning a model for object d includes the specification family M , and subsequently the estimation of $m_d \in M$ from training data $F^{(d)}$. Choosing M is hard because there is no criterion for the goodness of a family of functions. It is often defined vaguely as one that has good approximation properties for the problem at hand and supports computational efficiency. In section 5.3, we will propose the use of sparse function approximation.

Once the object models $\{m_d\}$ have been learned, we can estimate the optimal pose $\phi^{(d)}$ of each object d for a two-dimensional input image $f(x_i, y_j)$ with the sum of squared differences measure by solving the following minimization problem

$$\phi^{(d)} = \arg \min_{\phi} \sum_{i,j} \left(f^{(d)}(x_i, x_j, \phi) - f(x_i, y_j) \right)^2 \quad (5.13)$$

For the coarse recognition task, the object identity d^* can be recognized by finding the closest view across the whole dataset

$$d^* = \arg \min_d \left\{ \min_{\phi} \sum_{i,j} \left(f^{(d)}(x_i, x_j, \phi) - f(x_i, x_j) \right)^2 \right\} \quad (5.14)$$

The minimization problem in eq. (5.13) is nontrivial since the objective function is generally nonconvex. Thus, local optimization methods do not suffice. In addition, the optimization problem is carried out in a high dimensional space that is equal to the number of pixels of the input image. In section 5.3.4, we present a solution to this optimization problem by turning it into the problem of piecewise polynomial evaluation.

To minimize eq. (5.14), we solve eq. (5.13) repeatedly for all d , $1 \leq d \leq D$. The challenge of minimizing eq. (5.14) without having to solve eq. (5.13) for all d is beyond the scope of the current chapter. Nevertheless, it does not stand in the way of real-time performance.

In short, we have to specify a family of functions M and learn the functions in M from training data. In addition, we have to solve the minimization problem in eq. (5.13) for recognition.

5.3 Model learning and recognition

We treat the problem of model learning for each object d as that of sparse multivariate function approximation [53, 14]. The function $f^{(d)}(x, y, \phi)$ is approximated by a linear combination of bases

$$f^{(d)}(x, y, \phi) = \sum_{k=1}^K \alpha_k^{(d)} \psi_k(x, y, \phi) \quad (5.15)$$

where $\psi_k(x, y, \phi) \in \Psi$, a predefined dictionary of K bases, and $\alpha_k^{(d)}$ are the coefficients. Consequently, the model m_d consists of the non-zero coefficients $\alpha_k^{(d)}$ and the indices of corresponding bases in the dictionary. The sparsity comes from the fact that only a small number of $\alpha_k^{(d)}$ differ from zero.

We first present our preferred dictionary Ψ in 5.3.1. After that we discuss the model learning from the chosen dictionary.

5.3.1 Gaussian derivative bases

The design of the dictionary Ψ reflects the a priori information about the object appearance. First of all, one can observe that the functions representing images have certain general properties. There are abrupt changes due to geometric concavities of the object and of the projection. Other causes include object sharp convex folds, albedo transition and projected shadow. In spite of these abrupt changes in the image, the majority of pixels are smooth by the spatial coherence of object surfaces. The general challenge is to find a set of functions, which match comfortably to the smooth regions almost everywhere and at the same time adapt to abrupt changes here and there, see also figure 5.2.

Figure 5.2: The middle lines of 2D views of the object in figure 5.1 as the pose changes.

The dictionary Ψ consists of separable bases

$$\psi_k(x, y, \phi) = \psi_{\tau_k^x}(x) \psi_{\tau_k^y}(y) \psi_{\tau_k^\phi}(\phi) \quad (5.16)$$

where $\psi_{\tau_k^x}$, $\psi_{\tau_k^y}$ and $\psi_{\tau_k^\phi}$ are the bases in each dimension, indexed by τ_k^x , τ_k^y and τ_k^ϕ in the corresponding dictionary.

Figure 5.3: Gaussian derivative bases (a) zero order (b) first order and (c) second order. The bases in the ϕ -axis are periodic over $[0, 2\pi)$, see (d).

Our choice for the one-dimensional bases is the Gaussian derivatives $G_n(z; \sigma, \mu)$ of order n at scale σ and location μ

$$G_n(z; \sigma, \mu) = \frac{\partial^n}{\partial x^n} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}} \quad (5.17)$$

Here the analyzing scales are sampled exponentially. The analyzing locations are sampled equally at each scale. Figures 5.3(a), 5.3(b) and 5.3(c) show examples of zeroth, first and second order Gaussian derivative bases, respectively. The zeroth, first and second order bases form the bases for the second order approximation of the Taylor expansion of the image [44], where the finite recording and observation resolutions imply that the finest scales are capable of sufficiently localizing the abrupt changes.

5.3.2 Piecewise polynomial approximation

In practice, to acquire great computational advantages, we approximate the Gaussian derivatives, similar to recursive implementation [103, 33], by B-spline [101, 102], see figure 5.4.

Figure 5.4: The cubic B-splines (dash lines) approximate well the Gaussian derivatives (solid lines) up to the second order.

A B-spline of order n denoted by $\beta^n(z)$ is generated by convolving $n + 1$ times $\beta^0(z)$ with itself

$$\beta^n(z) = \beta^{n-1}(z) * \beta^0(z) = \underbrace{\beta^0(z) * \beta^0(z) * \cdots * \beta^0(z)}_{n+1 \text{ times}} \quad (5.18)$$

where $*$ denotes convolution and $\beta^0(z)$ is a centered normalized rectangle pulse

$$\beta^0(z) = \begin{cases} 1 & \text{if } |z| < \frac{1}{2} \\ \frac{1}{2} & \text{if } |z| = \frac{1}{2} \\ 0 & \text{otherwise} \end{cases} \quad (5.19)$$

The n th order derivative of the B-spline can be obtained from the $(n - 1)$ th order B-spline as follows

$$\frac{\partial \beta^n(z)}{\partial z} = \beta^{n-1}\left(z + \frac{1}{2}\right) - \beta^{n-1}\left(z - \frac{1}{2}\right) \quad (5.20)$$

For our purpose, the cubic B-splines are represented by piecewise polynomials. Hence, the 1D basis functions $\psi_{\tau_k^x}(x)$, $\psi_{\tau_k^y}(y)$ and $\psi_{\tau_k^\phi}(\phi)$ are piecewise polynomial of degree three. We will show in 5.3.4 that efficient recognition is achieved by polynomial calculation.

5.3.3 Model learning with matching pursuit

Once a family of functions Ψ has been chosen, the next step is to estimate a specific model from a set of training examples. Among the various methods for function approximation from a dictionary of bases (see appendix A), we use the matching pursuit algorithm of Mallat and Zhang [53] yielding a local optimum.

The matching pursuit algorithm aims to learn $f^{(d)}(x, y, \phi)$ in eq. (5.15) as follows. At each step it finds the basis most correlated with the residual. Let $\langle \cdot, \cdot \rangle$ denote inner product. Initialize the residual function $\rho^1(x, y, \phi)$

$$\rho^1(x, y, \phi) = f^{(d)}(x, y, \phi) \quad (5.21)$$

Then loop over all bases and select the index k_r^* of the best basis function in Ψ

$$k_r^* = \arg \max_k \langle \rho^r, \psi_k \rangle \quad (5.22)$$

with a coefficient α along that dimension

$$\alpha_{k_r^*}^{(d)} = \langle \rho^r, \psi_{k_r^*} \rangle \quad (5.23)$$

After that, update the residual function

$$\rho^{r+1} = \rho^r - \alpha_{k_r^*}^{(d)} \psi_{k_r^*} \quad (5.24)$$

and continue over the next basis. The number of bases R can be chosen a priori. Alternatively, we can stop the iteration when a good approximation has been reached by checking the residual

$$\|\rho^r\|^2 = \left\| f^{(d)}(x, y, \phi) - \sum_{k=1}^K \alpha_k^{(d)} \psi_k(x, y, \phi) \right\|^2 \quad (5.25)$$

The computation of our matching pursuit algorithm is intensive. In each iteration of the matching pursuit algorithm one has to compute the project of the current residual ρ^r on all bases $\psi_k \in \Psi$ (eq. 5.22). Fortunately, one can use the so-called network calculations for matching pursuit as in [52]. By taking an inner product with a basis ψ_k on each side of eq. (5.24), we have

$$\langle \rho^{r+1}, \psi_k \rangle = \langle \rho^r, \psi_k \rangle - \alpha_{k_r^*}^{(d)} \langle \psi_{k_r^*}, \psi_k \rangle \quad (5.26)$$

Hence, the correlation between the new residual ρ^{r+1} with a basis ψ_k can be computed from the existing correlation value and the correlation between the selected basis and ψ_k . By storing the correlation between all pairs of bases, eq. (5.26) can be computed in a constant time.

The number of bases K can be quite large, equal to the product of the number of bases of the 1D dictionaries, far more than the capability of standard computer

systems. Thanks to the separable bases, we can reduce the memory storage greatly. The correlation between two bases ψ_k and $\psi_{k'}$ can be computed as (using eq. 5.16)

$$\begin{aligned}\langle \psi_k, \psi_{k'} \rangle &= \langle \psi_{\tau_k^x}(x) \psi_{\tau_k^y}(y) \psi_{\tau_k^\phi}(\phi), \psi_{\tau_{k'}^x}(x) \psi_{\tau_{k'}^y}(y) \psi_{\tau_{k'}^\phi}(\phi) \rangle \\ &= \langle \psi_{\tau_k^x}(x), \psi_{\tau_{k'}^x}(x) \rangle \langle \psi_{\tau_k^y}(y), \psi_{\tau_{k'}^y}(y) \rangle \langle \psi_{\tau_k^\phi}(\phi), \psi_{\tau_{k'}^\phi}(\phi) \rangle\end{aligned}\quad (5.27)$$

Thus, instead of storing one big table for the correlation between all pairs of bases in Ψ , we store three tables, one for each dimension. And, each correlation value of bases in Ψ is obtained by just two multiplications.

In summary, the learning of an object model in eq. (5.15) is done efficiently by the matching pursuit algorithm. The network computation technique is used where the correlation between all pairs of bases are pre-computed. This large table is implemented by storing three correlation tables, one for each dimension, and is accessed by two multiplications.

5.3.4 Recognition

Expanding eq. (5.13), we have

$$\begin{aligned}\phi^{(d)} &= \arg \min_{\phi} \sum_{i,j} \left(f^{(d)}(x_i, y_j, \phi) - f(x_i, y_j) \right)^2 \\ &= \arg \min_{\phi} \left\{ \sum_{i,j} f^{(d)}(x_i, y_j, \phi)^2 - 2 \sum_{i,j} f^{(d)}(x_i, y_j, \phi) f(x_i, y_j) \right\}\end{aligned}\quad (5.28)$$

The first term in eq. (5.28) is the energy

$$\begin{aligned}\sum_{i,j} f^{(d)}(x_i, y_j, \phi)^2 &= \\ &= \sum_{i,j} \left(\sum_{k=1}^K \alpha_k^{(d)} \psi_{\tau_k^x}(x_i) \psi_{\tau_k^y}(y_j) \psi_{\tau_k^\phi}(\phi) \right)^2 \\ &= \sum_{k'=1}^K \sum_{k''=1}^K \left\{ \alpha_{k'}^{(d)} \alpha_{k''}^{(d)} \left(\sum_i \psi_{\tau_{k'}^x}(x_i) \psi_{\tau_{k''}^x}(x_i) \right) \left(\sum_j \psi_{\tau_{k'}^y}(y_j) \psi_{\tau_{k''}^y}(y_j) \right) \right. \\ &\quad \left. \times \psi_{\tau_{k'}^\phi}(\phi) \psi_{\tau_{k''}^\phi}(\phi) \right\} \\ &= \sum_{k'=1}^K \sum_{k''=1}^K \left\{ \alpha_{k'}^{(d)} \alpha_{k''}^{(d)} \langle \psi_{\tau_{k'}^x}, \psi_{\tau_{k''}^x} \rangle \langle \psi_{\tau_{k'}^y}, \psi_{\tau_{k''}^y} \rangle \psi_{\tau_{k'}^\phi}(\phi) \psi_{\tau_{k''}^\phi}(\phi) \right\}\end{aligned}\quad (5.29)$$

Note that each term in eq. (5.29) is a product of two piecewise polynomials, each of degree three. Hence, the energy is a piecewise polynomial of degree six at most.

In addition, this piecewise polynomial can be pre-computed since it does not involve the input image.

The second term is the cross correlation term (ignoring the constant factor)

$$\begin{aligned}
\sum_{i,j} f^{(d)}(x_i, y_j, \phi) f(x_i, y_j) &= \\
&= \sum_{i,j} \left(\sum_{k=1}^K \alpha_k^{(d)} \psi_{\tau_k^x}(x_i) \psi_{\tau_k^y}(y_j) \psi_{\tau_k^\phi}(\phi) \right) f(x_i, y_j) \\
&= \sum_{k=1}^K \alpha_k^{(d)} \psi_{\tau_k^\phi}(\phi) \left(\sum_{i,j} \psi_{\tau_k^x}(x_i) \psi_{\tau_k^y}(y_j) f(x_i, y_j) \right) \quad (5.30)
\end{aligned}$$

The sum within the brackets is an Njet coefficient corresponding to the basis $\psi_{\tau_k^x}$ in the x -axis and the basis $\psi_{\tau_k^y}$ in the y -axis. Let $\mathcal{N}_{\tau_k^x, \tau_k^y}^I$ denote this value. By grouping the Njets that belong the same basis in the ϕ -axis, we have

$$\sum_{i,j} f^{(d)}(x_i, y_j, \phi) f(x_i, y_j) = \sum_{\tau^\phi=1}^{K_\phi} \left(\sum_{k, \tau_k^\phi = \tau^\phi} \alpha_k^{(d)} \mathcal{N}_{\tau_k^x, \tau_k^y}^I \right) \psi_{\tau^\phi}(\phi) \quad (5.31)$$

where K^ϕ denotes the number of bases in the ϕ -axis. Eq. (5.31) is the sum of piecewise polynomials of degree three. Hence, the result is also a piecewise polynomial of degree three.

Because the first term (eq. 5.29) is a piecewise polynomial of degree six and the second term (eq. 5.31) is a piecewise polynomial of degree three, (eq. 5.28) is a minimization problem of a piecewise polynomial $g(\phi)$ consisting of Q polynomials $g_q(\phi)$, $1 \leq q \leq Q$, each of which is of degree six at most

$$\begin{aligned}
(\phi^{(d)}, q^*) &= \arg \min_{\phi, q} g_q(\phi) \\
&= \arg \min_{\phi, q} \sum_{\gamma=0}^6 b_{q,\gamma} \phi^\gamma \quad \text{for } \phi_q \leq \phi < \phi_{q+1} \quad (5.32)
\end{aligned}$$

where ϕ_q are the break points, and $b_{q,\gamma}$ are the polynomial coefficients of $g_q(\phi)$. As a consequence, the problem (5.28) can be turned into Q minimization of $g_q(\phi)$.

A simple approach to minimize (5.32) is to evaluate $g_p(\phi)$ at a set of densely sampled points, hence, is equivalent to the approach of Murase and Nayar to solving eq. (5.11). However, the evaluation of polynomial is very fast. The evaluation of eq. (5.32) at an arbitrary point $\phi = v$ requires only six additions and six multiplications using Horner's rule [43]

$$g_q(v) = (((((b_{q,6}v + b_{q,5})v + b_{q,4})v + b_{q,3})v + b_{q,2})v + b_{q,1})v + b_{q,0} \quad (5.33)$$

Figure 5.5 depicts the computation of eq. (5.28). The input image is transformed into the Njet representation. The Njet coefficients $\mathcal{N}_{\tau_k^x, \tau_k^y}^I$, weighted by the

Figure 5.5: The recognition scheme.

corresponding $\alpha_k^{(d)}$, are grouped according to the model in eq. (5.31). The resulting scalars are in turn the weights of the corresponding bases in the ϕ -axis. The final result $g^{(d)}(\phi)$ to be minimized is the sum of the weighted bases, which is a piecewise polynomial of degree six.

Our recognition strategy compares favorably to the approach of Murase and Nayar in both storage space and time performance. In terms of storage space, the PCA method has to store all sampled points in the eigenspace. For instance, 3600 points are required for each object for accuracy of 0.1 degree. On the contrary, the new method stores the coefficients of the polynomials, which is marginal. Furthermore, it is independent of the accuracy; that is, higher accuracy is obtained at no extra storage cost.

Figure 5.6(a) shows the time performance of the two methods with respect to the number of objects. For the case one object fills the image, it takes more time to compute the Njet representation than PCA projection in this case. However, the new method solves the minimization problem in eq. (5.32) (including the computation of the polynomial coefficients $b_{q,\gamma}$) faster than the minimization in the eigenspace (eq. 5.11) as shown in figure 5.6(b). As a result, for compatible accuracy for a dataset of 100 objects the new method with 300 bases performs similar to PCA with 6 dimensions. With 1000 bases it is two times faster the PCA method with 20 dimensions. The differences will become more pronounced for larger datasets. In the other important case one has to scan an image over multiple scales and locations in search for an object, our method is advantageous, because the Njet representation is computed only once for the whole image, as opposed to the PCA approach which

Figure 5.6: Time performance of the new method versus PCA with the number of principle components C on a computer with a Pentium 4 CPU 2.8 GHz for object images of size 128×128 . For both methods, 3600 points are sampled along the ϕ -axis for each object. (a) The total recognition time with respect to the number of objects. (b) Minimization time for eq. (5.32) versus that for eq. (5.11) in the PCA approach.

has to do projection for each location and scale separately. The complete Njet-based computation is now four to ten times faster than the PCA, depending on the number of its dimension.

5.4 Experiments

We contrast the performance of the new approach to the approach of Murase and Nayar using PCA on the COIL-100 dataset [60]. This dataset consists of images of 100 objects of size 128×128 , each of which is captured at 72 poses (five degrees apart). We convert all images into grayscale.

We divided the dataset into two partitions of the same size. The training set has 36 views for each object, at ten degrees apart. The remaining views are used for testing. Therefore, there are 720 object views in the training set and 720 object views in the test set.

The parameters of the method are set as follows. The dictionary Ψ consists of three 1D dictionaries. The dictionaries in the two spatial directions x and y have three scales $\sigma \in \{4, 8, 16\}$; in each scale the space μ are sampled at 4, 8, 16 (pixels) apart, respectively. In total, there are 120 bases in each spatial 1D dictionary. In the ϕ direction, we examine at scale $\sigma \in \{1, 2, 4, 8\}$. The space is also sampled correspondingly at 1, 2, 4 and 8 ($\times 10$ degrees).

The number of discrete points sampled for our minimization problem (eq. 5.32) and the PCA approach (eq. 5.11) is 3600, equivalent to 0.1 degree in the orientation directions.

5.4.1 Compactness of the representation

First of all, we examine the visual quality of the reconstructed images using the new representation in comparison with PCA.

Figure 5.7(a) shows examples of generated images by the new representation using 1000 bases for each object. For this representation each basis requires four storage locations, one for the coefficient and three for the indices of the three one-dimensional bases. Thus the total storage locations required for each objects is

$$4 \times R \quad (5.34)$$

where R is the number of bases. For PCA, one has to store the mean and the eigenvectors which has the same size as the input dimensionality ($N \times N$), and the representation in the eigenspace with dimensionality C , see figures 5.7(b) and 5.7(c). The number of storage locations required for each of the D objects is

$$\frac{(1+C)N^2}{D} + CS \quad (5.35)$$

where S is the number of points to be stored for each object in the eigenspace. The storage space for PCA is much larger than that for the new representation.

Figure 5.7: Generated images from different representation learned on the COIL-100 training set. (a) The new method with 1000 bases. (b) PCA with 20 dimensions ($T = 20$). (c) PCA with 100 dimensions ($T = 100$). The new method gives better results at a lower storage cost by exploiting the spatial correlation in images.

5.4.2 Recognition performance

Figure 5.8(a) shows the coarse recognition error rate on the test set. The method obtains an error rate of 0.17% with about 300 bases only. Given the variety, object discrimination typically does not require precise descriptions. For fine recognition, the accuracy increases as more bases are used as shown in figure 5.8(b). The pose error appears saturated after approximately 1000 bases.

We can observe similar behavior with the PCA approach. Figure 5.9(a) shows the recognition error rate on the test set as a function of dimensionality of the eigenspace. The recognition error goes down to 0.19% using only six dimensional eigenspace for coarse classification. For a 20 dimensional eigenspace, the average pose error shown in figure 5.9(b) decreases to an equal number for the new method with 1000 bases. Hence, in terms of accuracy the two methods are comparable when a 20 dimensional eigenspace are compared to 1000 sparse bases. As an aside, the new method (and also PCA) compares favorably to the support vector machine method with various kernel functions [106] where the best result is 0.4% error rate.

In the next experiment, we vary the size of the training set for $D = 10, 20, \dots, 100$,

Figure 5.8: Recognition and pose estimation results on the COIL-100 test set. The number of evaluation points (eq. 5.33) for each object is 3600 (0.1 degree accuracy).

Figure 5.9: Recognition and pose estimation results of PCA on the COIL-100 test set. The number of discrete points stored for each object is 3600 (0.1 degree accuracy).

by sampling from the COIL-100 dataset. We split each dataset into a training set and a test set. We repeat the experiment 20 times and average the results. Figure 5.10(a) shows the result for coarse recognition and figures 5.10(b), 5.10(c) and 5.10(d) show results for pose accuracy within 45, 10 and 2 degrees, respectively. We extrapolate an exponential model $\xi = \gamma e^{\delta D}$ where ξ denotes the correct recognition rate and γ and δ are the parameters of the model. If the average recognition rate of two datasets of size u_1 and u_2 are respectively ξ_1 and ξ_2 , then the average recognition rate of a dataset of size $u_1 + u_2$ is approximately $\xi_1 \times \xi_2$. Thus, an exponential model is suited. The accuracy on a dataset with 500 objects is predicted to be 99% for coarse recognition and 96% for accuracy within 10 degrees using 1000 bases.

In short, the accuracy of the new method is comparable to PCA. The new method has advantage over time and storage space in comparison with PCA.

Figure 5.10: The recognition rate on datasets of different sizes. Each result is obtained by averaging over 20 subsets randomly taken from the COIL-100 dataset [60]. The dashed lines are obtained by fitting an exponential model. The pose error is (a) ignored, (b) within 45 degrees, (c) within 10 degrees (d) within 2 degrees.

5.5 Discussion and conclusion

This chapter addresses the problem of coarse and fine object recognition. We have considered three main issues in this problem, namely object representation, model learning from examples and efficient object matching.

We proposed a new representation by sparse function approximation in both spatial dimensions together with the orientation dimension. Thus, the generalization to unseen views is naturally obtained. This is in contrast to previous approaches that use low dimensional representation by a linear projection, such as principle component analysis, or the view-based representation where the spatial coherence principle is ignored. The new representation is able to exploit a rich amount of a priori information

about the object views in general.

We have also presented a solution to the problem of object view matching in a high dimensional space by polynomial computation. In particular, we showed that the minimization problem involved in recognition phase can be solved by evaluating polynomials of degree six at multipoints. Hence, the computation of the new algorithm equals the PCA approach with six principle components.

We performed experiments on the COIL-100 dataset. The results show the high visual quality of the new representation in comparison with PCA. This is because the new representation takes into account of the spatial correlation. The experimental results also show that the new method performs as well as the approach using PCA in term of accuracy.

There are three significant advantages of the new method over PCA. Firstly, the storage of sampled points for recognition is not required for the new approach. Secondly, the addition of new objects into an existing dataset in the new approach is trivial, because unlike PCA re-training other object models is not needed. Thirdly, in case one has to scan the input image for recognition, the Njet representation is computed only once, as opposed to PCA projection at each location.

In summary, we propose a new representation for object recognition where the object model can be learned efficiently with a rich amount of a priori information. The new representation also allows fast recognition by polynomial computation. Overall, the method is comparable to the PCA approach in terms of accuracy, and more efficient in terms of storage space and recognition time. Real-time performance is achieved for a dataset of 100 objects.

Appendix A

An alternative approach for sparse function approximation is the basis pursuit denoising algorithm [14], which provides a global optimum for

$$\min_{\alpha^{(d)}} \frac{1}{2} \left\| f^{(d)}(x, y, \phi) - \sum_{k=1}^K \alpha_k^{(d)} \psi_k(x, y, \phi) \right\|^2 + \lambda \cdot \sum_{k=1}^K |\alpha_k^{(d)}| \quad (5.36)$$

where λ is a parameter reflecting the tradeoff between accuracy (the first term) and sparsity of the solution (the second term). This optimization procedure is infeasible for our problem due the large size of the dictionary Ψ , typically, several orders of magnitude larger than those tested in [14]. Also, in some cases the result of matching pursuit may be better than that of basis pursuit. Figure 5.11 shows an example where the basis pursuit has an optimal solution of 22 bases, while matching pursuit obtain good result with only 10 bases. This result is not surprising because while basis pursuit denoising has a global optimum, like matching pursuit it only provides a suboptimal solution to finding the best approximation given a fixed number of bases, or minimizing the number of bases given the allowed reconstruction error.

Figure 5.11: The matching pursuit (MP) algorithm versus the basis pursuit (BP) algorithm for sparse function approximation. The original 1D signal is the middle line of the first object in figure 5.1. Both algorithms use the same dictionary of 2048 Gaussian bases. The optimal solution obtained by the basis pursuit algorithm consists of 22 bases. The results obtained by the matching pursuit algorithm with 1, 3, 10 and 22 bases are shown respectively in (a),(b),(c) and (d).

Chapter 6

Quadratic boosting

6.1 Introduction

Combining classifiers by boosting algorithms, especially the AdaBoost [28] method, is currently in frequent use in machine learning and pattern recognition [82, 19]. These methods convert a weakly performing base learner into one with better performance. The main characteristic of boosting algorithms is that they treat the base learner as a black box. They do not require access to the parameters of the base classifiers. This model of learning is flexible since knowledge about the problem can be incorporated in the base learner while it performs with moderate accuracy. Viola and Jones [107] use a space of features designed to ensure fast calculation, while the use of AdaBoost results in a face classifier that is both fast and accurate. Other typical application fields of AdaBoost are image retrieval [99], part-based object classifiers [86], and dynamic Bayesian network classifiers for event detection in video [65].

The AdaBoost algorithm uses a linear combination f of the base classifiers $h(\cdot)$,

$$f(x) = \sum_t \alpha_t h_t(x) \quad (6.1)$$

where α_t is the weight of the base classifier h_t . The index t denotes the iteration of the algorithm. The algorithm learns classifiers by exploiting the fact that the given learner can be used with different versions of the training set. In each round the AdaBoost algorithm employs a complex scheme of reweighting of training examples.

The limitation of the linear combination in eq. (6.1) has not received much attention partly because the choice of the base learner is flexible. Virtually any learning algorithm can be used. This can lead to very complex decision boundaries. However, there are cases where the choice of the base learner is restricted. We mention two cases. First, apart from accuracy the base classifiers may be subject to other design constraints such as classification time or integration with other modules of a larger system. Second, in many situations the use of some classifiers (or features) is an established practice and the combined classifier is to be based on these given results [39].

Under these circumstances the limitation of the linear combination is well studied. A standard example is the exclusive OR problem [55]. Given a space consisting of two classifiers $H = \{h_1, h_2\}$ as in figure 6.1, there is no (threshold) linear combination of h_1 and h_2 that can separate the two classes denoted by the empty and filled circles. Several practical limitations of the linear combination are also given in [55]. For instance in visual recognition, when each base classifier draws its input from a local region in the image, the combined classifier belongs to the class of diameter-limited perceptrons which cannot recognize global concepts such as connectedness.

Figure 6.1: The exclusive OR problem. A classifier space $H = \{h_1, h_2\}$, where each classifier is represented by a straight line boundary in 2D. The input space X consists of 4 points in 2D. The filled circles are labeled +1 while the empty circles -1. The two classes cannot be separated by any (threshold) linear combination of the two classifiers [55].

In this chapter, we consider a quadratic combination of two-class classifiers

$$f_q(x) = \sum_i \alpha_i h_i(x) + \sum_{ij} \alpha_{ij} h_i(x) h_j(x) \quad (6.2)$$

While it seems that this is a simple generalization of the core of AdaBoost in eq. (6.1), learning this form of combination is nontrivial. Replacing eq. (6.1) of AdaBoost by eq. (6.2) in a functional minimization strategy as introduced by Breiman [11] (to be discussed later) would lead to the task of selecting a pair of classifiers together minimizing the empirical error, which cannot be solved by a simple one term inversion as is possible for linear boosting by invoking the input learner. An equivalent view of this derivation is to treat the quadratic combination as linear in an extended classifier space. The learner to operate on the combined linear and quadratic terms inherits a fundamental difficulty of a boosting method, namely the input classifiers are not available directly. The realization of standard optimization techniques becomes impossible since we do not have direct access to the parameters of the base classifiers.

We will show that learning in the extended classifier space is possible by at first randomizing the labels of the training examples and subsequently steering the performance of the new learner in an indirect way with a systematically updated set of

labels. Thus learning the quadratic combination is done effectively by both reweighting and relabeling of training data.

We provide an analysis of the generalization power of the new algorithm. We also discuss the computational aspects of the new combination in learning and classification. We perform extensive experiments on synthetic data, on standard machine learning datasets, as well as on datasets for the object recognition problem in computer vision. The experimental results show that the new combination outperforms the existing linear boosting in most cases, specifically on large datasets. Significantly, while the new algorithm requires more training time to learn the decision boundary better, it does not require more classification time.

The chapter is organized as follows. In the next section we formulate the problem. In section 6.3 we provide detail of the AdaBoost algorithm and related theoretical background. Section 6.4 presents the new algorithm and our analysis of its performance. The experiments are given in section 6.5. Finally, section 6.6 concludes the chapter.

6.2 Problem statement

Let $\{(x_n, y_n)\}$ be the training set of N labeled examples drawn randomly from an unknown distribution $P(x, y)$, where $x \in X$ denotes a pattern and $y \in Y$ is the class label. We only consider the two-class case, that is $Y = \{-1, +1\}$. A classifier $h \in H$ predicts the class label y of an input pattern x , $y = h(x)$. A learning algorithm L_H is a functional taking the training data as input and returns a classifier, $h = L_H(\{(x_n, y_n)\})$. The aim of the learning algorithm L_H is to find the optimal classifier $h^* \in H$, or an approximation to it, that makes the least number of mistakes in predicting class labels of future patterns. This number is the generalization error of a classifier h , denoted by $\epsilon(h)$.

Given the set of classifiers H and the learner L_H , one can train a combined classifier $\mathcal{F}(x; \{h_i\}, \Gamma)$, where $\{h_i\} \subset H$ is a finite subset of base classifiers, Γ is a set of additional parameters and $\mathcal{F}(x; \cdot)$ is some combination function that will perform better than each of the classifiers learned by L_H .

We consider a combination rule $\mathcal{F}(x; \cdot)$ which is the sign of a quadratic combination of classifiers f_q as in eq. (6.2)

$$\mathcal{F}(x; \cdot) = \begin{cases} +1 & \text{if } f_q(x) > 0, \\ -1 & \text{otherwise.} \end{cases} \quad (6.3)$$

For eq. (6.2), $f_q(x)$ is linear in the space $G = H \cup H^2$, where $H^2 = \{h_i h_j | h_i, h_j \in H\}$. As a result, it can be learned by existing methods for linear combination with a base learner in the space G . We choose the AdaBoost algorithm as the learner for this linear combination. Based on the argument of Breiman [11] presented in the next section, an optimization criterion for the base learner is to minimize the error on the training data. Thus, the first problem we address is to construct from the input learner L_H an intermediate learner L_G aiming at minimizing the empirical error.

The second issue we address is that of the generalization error. By going from the classifier space H to G we effectively increase the degree of freedom of the combined classifier. In case H is finite with C classifiers, the number of free parameters increases from C to $C + C^2$. Thus care must be taken with regard to overtraining the combined classifier. Our analysis is based on the concept of the capacity of learning algorithms and the theory of error bound [105, 28, 83] as reviewed in the next section.

6.3 Background

We present the AdaBoost algorithm and related theoretical studies that characterize its performance. The first study treats the AdaBoost algorithm as a minimization procedure, which motivates our choice of optimal criterion for L_G . The second analyzes the generalization capability of the algorithm, which provides a basis for the analysis of the new algorithm.

6.3.1 The AdaBoost algorithm

The AdaBoost algorithm is described in box 6.1. The central aspect of the algorithm is to maintain and to update a weight $w_n^{(t)}$ associated with each example x_n in each iteration t . In each round a classifier h_t is learned from the weighted training set and α_t is computed based on the performance of h_t . In this algorithm T is a pre-defined number of iterations. The algorithm stops when this number is reached, or the empirical error $\hat{\epsilon}_t$ is 0.5.

6.3.2 Optimization criterion for a base learner

To construct a base learner from the input learner, we need a design criterion. In [11], the author shows that under the condition that the base learner returns a classifier with minimal empirical error, the AdaBoost algorithm coincides with the so-called Gauss Southwell method [50] to solve the following minimization problem

$$\boldsymbol{\alpha}^* = \arg \min_{\boldsymbol{\alpha}} F(\boldsymbol{\alpha}) \quad (6.8)$$

where

$$F(\boldsymbol{\alpha}) = \sum_n e^{-y_n f(x_n)} \quad (6.9)$$

$\boldsymbol{\alpha}$ is the vector containing all α_t and f is the linear combination of classifiers in eq. (6.1). Thus the learned classifiers are h_t with $\alpha_t \neq 0$.

Optimizing this criterion is reasonable because on average we want $yf(x)$ as large as possible. It is shown in [84] that $F(\boldsymbol{\alpha})/N$ is an upper bound on the empirical error. This is because for an incorrect prediction we have $y_n f(x_n) \leq 0$; hence, $e^{-y_n f(x_n)} \geq 1$. This implies that $F(\boldsymbol{\alpha})$ is at least equal to the total number of incorrect predictions on

Input: Base learner L_H
Training data $\{(x_n, y_n)\}, n = 1, \dots, N$.

Initialize: $w_n^{(1)} = 1/N$

Do for $t = 1, \dots, T$

1. Learn a classifier h_t and its weight α_t with the aid of the learner L_H

$$h_t = L_H \left(\{(x_n, y_n, w_n^{(t)})\} \right) \quad (6.4)$$

$$\hat{\epsilon}_t = \sum_{n, y_n \neq h_t(x_n)} w_n^{(t)} \quad (6.5)$$

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \hat{\epsilon}_t}{\hat{\epsilon}_t} \right) \quad (6.6)$$
2. update weights
$$w_n^{(t+1)} = \frac{w_n^{(t)} e^{-\alpha_t y_n h_t(x_n)}}{Z_t} \quad (6.7)$$

where Z_t is a normalization factor.

Output:

$$f(x) = \sum_t \alpha_t h_t(x)$$

Box 6.1: The AdaBoost algorithm.

the training set. Thus minimizing $F(\alpha)$ drives down the training error. Friedman *et al.* [29] provide another argument for minimizing $F(\alpha)$. They show that minimizing this criterion is equivalent to a stagewise fitting of the so-called additive logistic regression model.

Specifically, Breiman [11] shows that the AdaBoost algorithm minimizes $F(\alpha)$ by first starting with all α_t equal zero. Let f_t be the combined classifier after t rounds. Let $w_n^{(t)}$ and Z_t be defined in each round t of the AdaBoost algorithm as in box 6.1 with $Z_0 = 1$. In the following rather than using t as we have so far for indexing classifiers and weights by iteration, now we use i for indexing all classifiers and weights. Let $\alpha_i^{(t)}$ denote the values of α_i in round t . In round t , consider the partial derivative with respect to the coordinate α_i

$$\left. \frac{\partial F}{\partial \alpha_i} \right|_{\alpha_i = \alpha_i^{(t-1)}} = \sum_n e^{-y_n f_{t-1}(x_n)} (-y_n) h_i(x_n) \quad (6.10)$$

By expanding the weight updating rule (eq. 6.7) of the AdaBoost algorithm up to and including round $(t-1)$, we have

$$\left. \frac{\partial F}{\partial \alpha_i} \right|_{\alpha_i = \alpha_i^{(t-1)}} = \sum_n \left\{ \prod_{j=0}^{t-1} Z_j \right\} w_n^{(t)} (-y_n) h_i(x_n) \quad (6.11)$$

$$= \left\{ \prod_{j=0}^{t-1} Z_j \right\} (2\hat{\epsilon}_i - 1) \quad (6.12)$$

where $\hat{\epsilon}_i$ is the empirical error of h_i . The absolute value of the gradient $|\partial F / \partial \alpha_i|$ at $\alpha_i = \alpha_i^{(t-1)}$ is proportional to $|\hat{\epsilon}_i - 0.5|$. Thus, in round t the learner L_H helps to select a classifier h_{i^*} as far from a random predictor as possible

$$i^* = \arg \max_i |\hat{\epsilon}_i - 0.5| \quad (6.13)$$

The AdaBoost algorithm selects the classifier h_t to be h_{i^*} , and α_t to be the optimal increase in $\alpha_{i^*}^{(t-1)}$ (again, the subscript t of h_t and α_t indicates boosting iteration)

$$\alpha_t = \alpha_{i^*}^{(t)} - \alpha_{i^*}^{(t-1)} \quad (6.14)$$

$$= \arg \min_{\Delta \alpha_{i^*}^{(t-1)}} \sum_n e^{-y_n (f_{t-1}(x_n) + \Delta \alpha_{i^*}^{(t-1)} h_{i^*}(x_n))} \quad (6.15)$$

This optimization problem has an analytic solution by setting the derivative to zero. Canceling out the nonzero constant we have

$$\sum_n e^{-y_n \Delta \alpha_{i^*}^{(t-1)} h_{i^*}(x_n)} w_n^{(t)} (-y_n) h_{i^*}(x_n) = 0 \quad (6.16)$$

Note that $y_n h_{i^*}(x_n) = 1$ indicates a correct prediction and $y_n h_{i^*}(x_n) = -1$ an incorrect prediction. This leads to

$$e^{\Delta \alpha_{i^*}^{(t-1)} \hat{\epsilon}_{i^*}} \hat{\epsilon}_{i^*} - e^{-\Delta \alpha_{i^*}^{(t-1)}} (1 - \hat{\epsilon}_{i^*}) = 0 \quad (6.17)$$

which, in turn, leads to eq. (6.6) of the AdaBoost algorithm.

In short, when the base learner returns the classifier with minimal empirical error the AdaBoost algorithm can be seen as selecting the coordinate with maximal gradient magnitude to descent in each round. In this view we design the new learner L_G to aim at minimizing the empirical error.

6.3.3 VC dimension and bounds on generalization error

Let $\hat{\epsilon}(h, S)$ denote the empirical error of a classifier $h \in H$ on a training set S . We have seen that the AdaBoost algorithm drives down the empirical error. The goal of a learning algorithm, however, is to minimize the error made on future data, $\epsilon(h)$. In most cases $\epsilon(h)$ cannot be computed.

To control the generalization error, Vapnik and Chervonenkis [104] introduced a measure of capacity of H . We will use the notations as in [5]. Let S be a set of m points in X . A dichotomy of S induced by $h \in H$ is a partition of S into two disjoint subsets S^+ and S^- such that $h(x) = +1$ for $x \in S^+$ and $h(x) = -1$ for $x \in S^-$. Let $\Delta_H(S)$ denote the number of distinct dichotomies of S induced by classifiers of H . Let $\Delta_H(m)$ denote the maximum of $\Delta_H(S)$ over all $S \in X$ of size m . The Vapnik and Chervonenkis (VC) dimension of H , denoted by $VCdim(H)$, is the largest m such that $\Delta_H(m) = 2^m$. Vapnik shows that for any training set S of size N and for any $h \in H$, with probability $1 - \eta$ one can assert

$$\epsilon(h) < \hat{\epsilon}(h, S) + 2\sqrt{\frac{VCdim(H) \left(\ln \frac{2N}{VCdim(H)} + 1 \right) - \ln \frac{\eta}{9}}{N}} \quad (6.18)$$

This bound indicates that as we minimize the empirical error the true error will also go down if the second term is sufficiently small. In particular, the smaller the VC-dimension, the smaller is the second term of the bound.

To analyze the generalization error of classifiers learned by the AdaBoost algorithm, Freund and Schapire [28] derive an upper bound on the VC dimension of this class of classifiers. Let $\Theta_T(H)$ be the class of all classifiers defined as a linear threshold of T classifiers in H . Clearly, the classifier learned by the AdaBoost algorithm after T iterations belongs to $\Theta_T(H)$. Freund and Schapire show that if $d = VCdim(H) \geq 2$, then

$$VCdim(\Theta_T(H)) \leq 2(d+1)(T+1)\log_2(e(T+1)) \quad (6.19)$$

where e is the base of the natural logarithm.

Another bound, which does not depend on the number of iterations T , is derived by Schapire *et al.* [83]

$$\epsilon(h) < P_S[yf(x) \leq \theta] + \mathcal{O}\left(\frac{1}{\sqrt{N}} \left(\frac{d \log^2(N/d)}{\theta^2} + \log(1/\eta) \right)^{1/2}\right) \quad (6.20)$$

for all $\theta > 0$, where P_S denotes the probability over the finite set S and the notation $\mathcal{O}(\cdot)$ means that constant factors are ignored. Again, this bound depends on the VC-dimension of the class of the base classifiers H . If all other terms are equal, the smaller the VC-dimension, the closer the true error is to the empirical *margin error* $P_S[yf(x) \leq \theta]$.

It is noted by the authors in [28, 83] that, in general, both bounds (6.19) and (6.20) are quite loose. However, it remains of theoretical interest to derive bounds on the generalization error. Furthermore, for a fixed value of the VC-dimension of the hypothesis space ($< \infty$), a large value of N leads to a small value of the second term in both bounds. Hence, the true error rate will follow the empirical error rate closely for large dataset.

In conclusion, the VC-dimension of the base classifiers plays a central role in analyzing the generalization error of the classifier learned by the AdaBoost algorithm.

We will derive an upper bound on the VC-dimension of the class of classifiers G . Since $f_q(x)$ is a linear combination of classifiers in G , we can control the generalization error using the above results as they are for a linear combination as well.

6.4 Quadratic boosting

We will first present a base learner that aims at minimizing the empirical error. We show that under the condition that the given learner L_H minimizes the empirical error, the new algorithm converges to a local minimum. The new algorithm is summarized in subsection 6.4.2. Finally we derive an upper bound for the VC-dimension of the classifier space G .

6.4.1 Derivation of the intermediate base learner

Our purpose is to design a base learner $g^* = L_G(\{(x_n, y_n, w_n)\})$ returning the best classifier on the training data

$$g^* = \arg \min_{g \in G} \hat{\epsilon}(g, \{(x_n, y_n, w_n)\}) \quad (6.21)$$

where $\{(x_n, y_n, w_n)\}$ is a labeled weighted training set and $\hat{\epsilon}(g, \{(x_n, y_n, w_n)\})$ denotes the empirical error

$$\hat{\epsilon}(g, \{(x_n, y_n, w_n)\}) = \sum_{n, y_n \neq g(x_n)} w_n$$

Recall that $G = H \cup H^2$. The case $g \in H$ is dealt with using the original learner L_H . For the case $g \in H^2$ we follow an iterative coordinate descent approach. The method consists of selecting an initial value and repeatedly descending along the two coordinates in turn. When the parameter set of the classifier space is known, one can easily initialize a random value and possibly implement a gradient descent algorithm, rather than coordinate descent. However, in boosting the parameter set is unknown. We will show that both selecting an initial value and performing coordinate descent are possible by calling L_H with different training sets obtained from the original training set by relabeling of examples. Consider the following procedure

Procedure 1

1. First, set $k = 0$ and let h_0 be a classifier resulting from randomly relabeling y_n .

$$h_0 = L_H(\{(x_n, y_n r_n, w_n)\})$$

where r_n is a random value in $\{-1, +1\}$.

2. Subsequently, learn classifiers by systematically relabeling y_n while incrementing k

$$\begin{aligned} h_k &= L_H(\{(x_n, y_n h_{k-1}(x_n), w_n)\}) \\ \hat{\epsilon}^{(k)} &= \hat{\epsilon}(h_k, \{(x_n, y_n h_{k-1}(x_n), w_n)\}) \end{aligned}$$

We will prove the following convergence result of $\hat{\epsilon}^{(k)}$.

Lemma 3 *Under the condition that L_H always achieves minimal empirical error, $\hat{\epsilon}^{(k)}$ converges.*

Proof: We have that h_k is learned by L_H . Therefore $h_k \in H$. Since L_H always finds the best $h \in H$ in terms of empirical error and $h_{k-1} \in H$ for $k > 0$, $h_{k+1} = L_H(\{(x_n, y_n h_k(x_n), w_n)\})$ will perform better than h_{k-1} on the dataset it is trained on

$$\hat{\epsilon}(h_{k+1}, \{(x_n, y_n h_k(x_n), w_n)\}) \leq \hat{\epsilon}(h_{k-1}, \{(x_n, y_n h_k(x_n), w_n)\}) \quad (6.22)$$

$$= \sum_{n, y_n h_k(x_n) \neq h_{k-1}(x_n)} w_n \quad (6.23)$$

$$= \sum_{n, y_n h_{k-1}(x_n) \neq h_k(x_n)} w_n \quad (6.24)$$

$$= \hat{\epsilon}(h_k, \{(x_n, y_n h_{k-1}(x_n), w_n)\}) \quad (6.25)$$

In short, $\hat{\epsilon}^{(k+1)} \leq \hat{\epsilon}^{(k)}$. In addition, $\hat{\epsilon}^{(k)} \geq 0$. Thus $\hat{\epsilon}^{(k)}$ converges. $\square$

Assign $g = h_k h_{k+1}$, we have

$$\hat{\epsilon}(g, \{(x_n, y_n, w_n)\}) = \sum_{n, y_n \neq h_k(x_n) h_{k+1}(x_n)} w_n \quad (6.26)$$

$$= \sum_{n, y_n h_k(x_n) \neq h_{k+1}(x_n)} w_n \quad (6.27)$$

$$= \hat{\epsilon}^{(k+1)} \quad (6.28)$$

Thus, procedure 1 effectively finds a local minimum for eq. (6.21) by iterative coordinate descent. To avoid local minima, the practical implementation of the intermediate base learner L_G invokes this procedure several times (with different set of random values $\{r_n\}$) and compares the relative improvement in performance.

Note that the nature of the combination rules as well as the optimization criteria employed by AdaBoost and L_G are different. As a result, while both algorithms use a special type of coordinate descent, the optimization procedure leads to relabeling in case of L_G and to reweighting in case of AdaBoost.

Procedure 1 can be extended to accommodate a polynomial combination of degree d with $d > 2$. Instead of relabeling y_n by multiplying it with the classification result of the previously learned classifier as in eq. (6.22), we now multiply it with the product of $d - 1$ previously learned classifiers. A similar convergence result as lemma 1 can be obtained.

6.4.2 The algorithm

Box 6.2 summarizes the quadratic boosting algorithm. In this algorithm R is the number of initial starting points to have a better chance to end up in the global minimum of (6.21). The maximum number of iterations of the inner most loop is K . It is needed because in practice certain base learners do not achieve minimal empirical error. In this situation the convergence result as derived in lemma 1 is not guaranteed. The algorithm invokes the base learner $R \times K + 1$ times at most. This provides a worse case estimate for the training time of the algorithm. By increasing the value of R and K one can look for a better solution at the cost of more training time.

The learner L_G fits well with the AdaBoost algorithm. Effectively the AdaBoost algorithm employs the base learner L_H by calling it repeatedly with a new set of weights, while L_G exploits the input base learner L_H by changing the labels of the training set. As a result, the new quadratic boosting algorithm makes more intensive use of the base learner than the original AdaBoost algorithm.

6.4.3 Generalization error

For the generalization error of the quadratic boosting we prove the following lemma.

Lemma 4 *Let d be the VC-dimension of H . The VC-dimension of G is less than $10d$.*

Proof: This proof follows the approach in [5] for bounding the VC-dimension of a linear threshold network. Let S be a set of m points in data space X . Recall from the definitions in subsection 6.3.3 that $\Delta_H(m)$ is the maximum number of distinct dichotomies that can be induced by the classifiers in H over all subsets of X of size m . Hence there are at most $\Delta_H(m)$ different dichotomies that can be induced on S by the classifiers in H . Consider the space of classifiers H^2 . Because $h(x)h(x)$ is constant for all $h \in H$ and $h_0(x)h_1(x) = h_1(x)h_0(x)$ for all $h_0, h_1 \in H$, the number of distinct dichotomies of S by classifiers in H^2 is at most $\Delta_H(m)(\Delta_H(m) - 1)/2 + 1$.

Input: Base learner L_H , training data $\{(x_n, y_n)\}$, $n = 1, \dots, N$.
 Number of restarts R , number of maximum inner iterations K .

Initialize: $w_n^{(1)} = 1/N$

Do for $t = 1, \dots, T$

1. Learn a classifier g_t and its weight α_t

(a) Learn a single classifier with the aid of L_H , $g^{(0)} = L_H \left(\{(x_n, y_n, w_n^{(t)})\} \right)$.

(b) Learn several compound classifiers $g^{(i)} = h_k^{(i)} h_{k-1}^{(i)}$, $i = 1, \dots, R$, to avoid local minima

i. First, set $k = 0$ and let $h_0^{(i)}$ be a classifier resulting from randomly relabeling y_n .

$$h_0^{(i)} = L_H \left(\{(x_n, y_n r_n^{(i)}, w_n^{(t)})\} \right)$$

where $r_n^{(i)}$ are random values in $\{-1, +1\}$.

ii. Subsequently, learn classifiers by systematically relabeling y_n while incrementing k until $\hat{\epsilon}(h_k^{(i)} h_{k-1}^{(i)}, \{(x_n, y_n, w_n^{(t)})\})$ converges [or $k=K$]

$$h_k^{(i)} = L_H \left(\{(x_n, y_n h_{k-1}^{(i)}(x_n), w_n^{(t)})\} \right)$$

(c) Assign the best classifier to g_t and compute α_t

$$g_t = \arg \min_{g^{(i)}, i \geq 0} \hat{\epsilon}(g^{(i)}, \{(x_n, y_n, w_n^{(t)})\})$$

$$\hat{\epsilon}_t = \hat{\epsilon}(g_t, \{(x_n, y_n, w_n^{(t)})\})$$

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \hat{\epsilon}_t}{\hat{\epsilon}_t} \right)$$

2. Update example weights

$$w_n^{(t+1)} = \frac{w_n^{(t)} e^{-\alpha_t y_n g_t(x_n)}}{Z_t}$$

where Z_t is a normalization factor.

Output: $f_q(x) = \sum_t \alpha_t g_t(x)$

Box 6.2: The quadratic boosting algorithm.

Thus

$$\begin{aligned}\Delta_G(m) &\leq \frac{\Delta_H(m)(\Delta_H(m) - 1)}{2} + 1 + \Delta_H(m) & (6.29) \\ &\leq \Delta_H(m)^2 & (6.30)\end{aligned}$$

Furthermore, by the result of Sauer [79, 2] we have, whenever $VCdim(F) = k < \infty$, $\Delta_F(m) \leq (em/k)^k$ for all $m \geq k$ where e is the base of the natural logarithm. Hence, $\Delta_G(m) \leq (em/d)^{2d}$ for all $m \geq d$.

It is simple to verify that $c = 10$ is the smallest integer satisfying the inequality $(ec)^2 < 2^c$, for $c \geq 1$. For $m = 10d$, we have $\Delta_G(m) \leq (e10)^{2d} < 2^{10d} = 2^m$. Thus for any set S of $m = 10d$ points, $\Delta_G(m) < 2^m$. Therefore, the VC dimension of G is less than $10d$. $\square$

Theoretically, this result can be used together with the bounds in 6.3.3 to control the generalization error.

6.5 Experiments

We perform three sets of experiments to compare quadratic boosting with the AdaBoost algorithm. The first experiment is done on synthetic data. Our purpose is to highlight the difference between quadratic boosting and AdaBoost. The next set of experiments is carried out on 18 standard machine learning datasets previously used to evaluate the AdaBoost algorithm. Finally, we perform the comparison on two very large datasets for the object classification problem.

6.5.1 On synthetic data

The following three datasets were generated to visualize the difference between the two methods. The patterns x are uniformly random points in the square $(0, 1) \times (0, 1)$. Each dataset has 300 examples. The labels of y are depicted in figure 6.2. The first

Figure 6.2: Three synthetic datasets. **A**: a straight boundary between classes. **B**: XOR pattern in the boundary. **C**: Band classes.

dataset has a straight boundary between the two classes. The second dataset has an XOR pattern in the class boundary. Finally, the third dataset has a band class separation.

The classifier space H consists of classifiers parallel with one of the two coordinates. Let $x = (u, v)$ be a data point. The classifiers are $\text{sign}(i/10 - u)$, $-\text{sign}(i/10 - u)$, $\text{sign}(i/10 - v)$, and $-\text{sign}(i/10 - v)$ for $i = 0, 1, \dots, 10$. Overall, H consists of 44 classifiers. The base learner L_H returns the best classifier in terms of empirical error for any set of weights.

Note that the AdaBoost algorithm using this learner has a staircase decision boundary on the first dataset [19]. This standard example shows the power of the AdaBoost algorithm to approximate more complex decision boundaries using simpler rules [19].

Figure 6.3 shows the result on the first dataset. The AdaBoost algorithm produces the staircase approximation as expected. The quadratic boosting algorithm performs only slightly worse than AdaBoost, but still produces a good approximation in spite of the fact that it has more degrees of freedom.

The improvement of the new algorithm is clearly seen in figure 6.4 and 6.5. On the last two datasets AdaBoost performs poorly. In contrast, quadratic boosting approximates well the true class boundary because of a richer classifier space.

Figure 6.3: Results on dataset **A** (300 data points) with single-coordinate base classifiers. A straight edge between two classes is the most favorable case for linear boosting and an extreme case for quadratic boosting. Nevertheless, the performance is more or less similar.

Figure 6.4: Results on dataset **B** (300 data points) with single-coordinate base classifiers. As expected, quadratic boosting performs better than AdaBoost in this case.

Figure 6.5: Results on dataset **C** (300 data points) with single-coordinate base classifiers. Again, quadratic boosting outperforms AdaBoost on this type of class separation.

6.5.2 On standard machine learning datasets

The next set of experiments is performed on standard machine learning datasets. All datasets are obtained from the UCI machine learning repository [8]. There are 18 datasets in total; all of which are two-class datasets. Table 6.1 lists the detail of these datasets, ordered by the number of examples. Most of these datasets have been used

#	dataset	# examples	#attributes		missing values
			disc.	cont.	
1	labor	57	8	8	x
2	promoters	106	57	-	-
3	hepatitis	155	13	6	x
4	sonar	208	-	60	-
5	cleve	303	7	6	x
6	ionosphere	351	-	34	-
7	house-votes-84	435	16	-	x
8	vote1	435	15	-	x
9	crx	690	9	6	x
10	breast-cancer-w	699	-	9	x
11	pima-indians-di	768	-	8	-
12	german	1000	13	7	-
13	hypothyroid	3163	18	7	x
14	sick-euthyroid	3163	18	7	x
15	kr-vs-kp	3196	36	-	-
16	clean2	6598	2	166	-
17	agaricus-lepiota	8124	22	-	-
18	adult	48842	12	2	x

Table 6.1: The 18 two-class datasets used in our experiment.

in previous studies of the AdaBoost algorithm [27, 74, 11, 4].

We use decision stump [38, 111] as the base learner. This learner produces a decision tree with only one level. In particular, we use the implementation available in the Weka machine learning package [111]. We leave the handling of discrete/continuous attributes and missing values to the base learning algorithm.

For evaluation we use ten fold cross validation [39, 111]. Each dataset is split into ten disjoint partitions of (approximately) equal sizes. Both training and evaluation are repeated ten times for each dataset. Each time a different partition is used for testing and the rest (nine partitions) are used for training. The error rates are averaged over ten runs to obtain the final estimate.

Table 6.2 gives the test error of AdaBoost and quadratic boosting on the 18 datasets after 250 base classifiers were learned. The quadratic boosting algorithm outperforms AdaBoost on large datasets. On dataset number 17 “agaricus-lepiota” although both algorithms reach an error rate of 0% quadratic boosting does so with many fewer base classifiers than AdaBoost, 22 in comparison with 74.

Quadratic boosting performs consistently better than the AdaBoost algorithm on training data (data not shown). However, on datasets with less than 1000 samples, overfitting occurs more often with the new algorithm than it does with the AdaBoost algorithm (see figure 6.6 for dataset “vote1”). This can be explained by the fact that the larger VC-dimension of the new base learner in this case does not guarantee

#	dataset	N	test error (%)			
			decision stump	AdaBoost	quadratic boosting	
1	labor	57	19.7	5.7	10.8	
2	promoters	106	29.9	6.6	6.6	
3	hepatitis	155	19.8	16.8	18.0	
4	sonar	208	26.8	17.2	14.9	✓
5	cleve	303	27.6	19.4	22.3	
6	ionosphere	351	17.7	8.3	7.7	✓
7	house-votes-84	435	4.4	3.4	4.6	
8	vote1	435	14.4	9.4	9.6	
9	crx	690	14.5	13.3	13.2	✓
10	breast-w	699	8.4	4.2	4.0	✓
11	pima-indians-di	768	28.1	23.9	26.3	
12	german	1000	30.0	24.7	24.4	✓
13	hypothyroid	3163	2.6	0.8	0.9	
14	sick-euthyroid	3163	5.6	2.6	2.3	✓
15	kr-vs-kp	3196	33.9	4.0	1.3	✓
16	clean2	6598	12.5	0.7	0.3	✓
17	agaricus-lepiota*	8124	11.3	0.0	0.0	✓
18	adult	48842	23.9	15.0	14.6	✓

Table 6.2: The generalization error (%) using ten fold cross validation after 250 classifiers have been learned. The check mark (✓) indicates datasets where quadratic boosting outperforms AdaBoost. *See the text for a discussion of the result on dataset “agaricus-lepiota”.

the decrease of the test error as the training error goes down. When the number of samples is large as of datasets from 15 to 18, the test error follows the training error closely (see figure 6.7 for dataset “kr-vs-kp”). This can be explained by the fact that the second terms of both error bound eq. (6.18) and eq. (6.20) are small because a large value of N overwhelms the VC-dimension of the base learner.

The next run of the experiment examines the relative performance of quadratic boosting and AdaBoost with respect to the amount of training data. Figure 6.8 shows the performance curve on the dataset “kr-vs-kp”. For this dataset, when the size is less than about 400, both AdaBoost and quadratic boosting do not perform well. The two methods are equivalent. For a larger size the result is stable, and quadratic boosting achieves better result than the AdaBoost algorithm.

In this experiment, the training time of quadratic boosting varies from approximately five to twenty times longer than that of the AdaBoost algorithm. Note that we compare the two methods with the same number of base classifiers. Assume the evaluation of each base classifier is equal, the classification time of quadratic boosting is not more than that of the AdaBoost algorithm. In the linear combination (eq. 6.1)

Figure 6.6: Performance curve with respect to the number of base classifiers for dataset **vote1** (435 examples).

Figure 6.7: Performance curve with respect to the number of base classifiers for dataset **kr-vs-kp** (3196 examples).

Figure 6.8: Performance curve with respect to size of the dataset for **kr-vs-kp**.

one addition and one multiplication are required for each base classifier. Whereas for each compound term (with two classifiers) in the quadratic combination (eq. 6.2) one addition and two multiplications are required.

6.5.3 On datasets for learning in vision

The next set of experiments is performed on the pedestrian and car datasets for the object detection problem [64]. Figure 6.9 and 6.10 show four examples of each dataset, respectively. The task is to distinguish target objects from the background. For each dataset in addition to the object examples, we generate 400000 background patterns by sampling uniformly over a set of images containing no target object. Each dataset is split into two equal partitions, one for training and one for testing.

In [107], the authors use the AdaBoost algorithm with set of features designed to be computed very fast. A two-rectangle feature is the difference between values of two vertically or horizontally adjacent regions, where the value of a region is the sum of all pixel values within that region. A three-rectangle feature is the difference between the sum of the values of the two outside rectangles and the value of the middle one. Finally a four-rectangle feature is the difference between two diagonal pairs. The base learner used is also the single-coordinate learner as described in previous experiments.

Figure 6.11 and 6.12 show the result on the two datasets after 150 classifiers have been learned. The quadratic boosting algorithm performs better than the AdaBoost algorithm on both datasets. With 150 classifiers, the AdaBoost algorithm has an error rate of 0.33% on the pedestrian dataset, while the quadratic boosting algorithm achieves 0.17%. On the car dataset, the error is 0.31% for the AdaBoost algorithm

Figure 6.9: Examples of the MIT pedestrian dataset, courtesy of [64].

Figure 6.10: Examples of the MIT car dataset, courtesy of [64].

Figure 6.11: Result on the MIT pedestrian dataset. For better visualization we provide the figure on the right with the number of classifiers ranges from 50 to 150.

and 0.15% for the quadratic boosting algorithm.

In both experiments the training time of the quadratic boosting algorithm is about ten times that of the AdaBoost algorithm. Again, as we have discussed, the runtime of the two methods are equivalent. Note that in both figures 6.11 and 6.12, 100 base classifiers combined by quadratic boosting can achieve the same accuracy as combining 150 base classifiers by linear boosting. Thus at this level of accuracy, substantial gain in classification time is obtained by quadratic boosting.

Figure 6.12: Result on the MIT car dataset. Again, the figure on the right shows the curve with the number of classifiers ranges from 50 to 150.

6.6 Discussion and conclusion

We addressed the problem of improving the accuracy of learning algorithms with a quadratic combination of base classifiers. This type of combination can be achieved with the construction of an intermediate base learner minimizing the empirical error. This step is difficult because as with other boosting methods, we face the problem that the base classifiers are not available directly. Thus it is not possible to use standard techniques for the minimization problem.

We derive a new algorithm solving the above problem. The method consists of calling the input learner after randomizing the labels of the dataset in the first round and subsequently calling it with a systematic update of the labels. This method is in contrast to the AdaBoost algorithm that uses reweighting of training examples. Together they form a powerful combination that makes intensive use the given base learner by both reweighting and relabeling the original training set.

We analyzed the algorithm, showing that under the condition that the input base learner minimizes the empirical error for any set of weights, the new base learner converges to a local minimum. We also study the generalization error of the new algorithm by upper bounding the VC-dimension of the new classifier space.

We performed extensive experiment with the quadratic boosting algorithm. It takes more time to train with the quadratic boosting algorithm than with the AdaBoost algorithm, typically from five to twenty times. This factor is dependent on the internal correlation of the data, but independent of the data size. The difference in classification time of the two algorithms, however, is marginal.

The experiment on synthetic datasets shows the clear improvement of the quadratic combination. In the experiment on standard machine learning datasets, we observed that the training error is almost always smaller for the quadratic boosting algorithm. For small datasets, however, quadratic boosting tends to overfitting more often than the AdaBoost algorithm. Nevertheless, the result on large datasets shows

that quadratic boosting outperforms AdaBoost. The experiment on object recognition datasets shows the error of quadratic boosting is approximately half that of the AdaBoost algorithm, which is a significant improvement.

We also performed an experiment with a third order polynomial combination on 18 machine learning datasets (data not shown). As expected, the experimental results show a worse performance than quadratic boosting on small size datasets. On large datasets, we observe a minor improvement in comparison with the result of the quadratic combination.

In conclusion, we have achieved a new boosting algorithm with a quadratic combination of base classifiers. The algorithm is particularly suited for large datasets. The algorithm requires more time than the AdaBoost algorithm in training, but not in classification, which provides an attractive choice for practical application.

Chapter 7

Summary and discussion

7.1 Summary

This thesis deals with the problem of recognizing objects from pictorial examples. Specifically, we aim to exploit the spatial relations within object appearance for this purpose. We explore new statistical models that improve on existing ones in the modeling power, learning time, or recognition time. We also study the problem of missing parts and false alarms in the part-based approach. The class of part-based methods is important because the structure of the objects is modeled explicitly, giving rise to elaborate and efficient recognition. Finally, we investigate a new representation combining multiple local image signs.

Chapter 2 presents a method for face detection. It learns a face classifier using a large set of face and non-face examples. The classifier is built from an aggregation of Bayesian network classifiers. The use of classifier aggregation is suited for the detection problem where a very low false alarm rate is required. Our Bayesian network method is superior to a related method using a Markov process model in that it offers a wider class of dependences among input variables with a forest-structured network while having a better algorithmic complexity of the learning procedure and identical recognition time. The face detection system performs equally well in comparison with other state-of-the-art methods in terms of accuracy.

Chapter 3 presents a method for the recognition of object classes once parts have been detected. The recognition task is formulated as a graph problem searching for the characteristic geographical arrangements of (possibly missing) parts. The objective function is Bayesian maximum a posteriori estimation, integrating the image likelihood as a posteriori probability of the part detectors. The variability in the arrangement of object parts is captured by a Gaussian distribution after translation normalization. The advantage of our approach over previous part-based methods is twofold. First, by employing two special properties of a Gaussian distribution, we are able to deal with missing parts including the situation where the origin for normalization is not detected. Second, we are able to find the optimal solution for the graph search problem with an elegant A^* algorithm. Experiments performed on

both synthetic and real data have demonstrated good results and fast performance of the recognition.

Chapter 4 studies two types of spatial relationships that can be learned from training examples for object recognition. The first one presented in chapter 2 describes fine-grain spatial relation between pixels. The second one presented in chapter 3 models coarse-grain spatial relation between object parts. Two datasets are used, one being human faces in frontal view as an example for the class of objects with a fixed spatial layout, and the other being horses faced to the right with large variations in action as an example of the articulated object class. By imposing (almost) the same amount of annotation of training data, which can be seen as sending knowledge to the learning algorithm, we identify the strengths and weaknesses of each method. We can also conclude that both methods are superior to strictly spatial matching by template and strictly non-spatial classifiers.

Chapter 5 offers a sparse, multiscale representation of objects. It captures the object appearance by selection from a very large dictionary of Gaussian differential basis functions. The learning procedure results from the matching pursuit algorithm, while the recognition is based on polynomial approximation to the bases, turning image matching into a problem of polynomial evaluation. The method is suited for coarse recognition between objects, and by adding more bases, also for fine recognition of the object pose. The advantages over the common representation using PCA include storing sampled points for recognition is not required, adding new objects to an existing dataset is trivial because re-training other object models is not needed, and significantly in the important case where one has to scan an image over multiple locations in search for an object, the new representation is readily available as opposed to PCA projection at each location. The experimental result on the COIL-100 dataset demonstrates high recognition accuracy with real-time performance.

Chapter 6 presents a strategy to improve the AdaBoost algorithm with a quadratic combination of base classifiers. We observe that learning this combination is necessary to get better performance and is possible by constructing an intermediate learner operating on the combined linear and quadratic terms. This is not trivial, as the parameters of the base classifiers are not under direct control, obstructing the application of direct optimization. We propose a new method realizing iterative optimization indirectly. First we train a classifier by randomizing the labels of training examples. Subsequently, the input learner is called repeatedly with a systematic update of the labels of the training examples in each round. We show that the quadratic boosting algorithm converges under the condition that the given base learner minimizes the empirical error. We also give an upper bound on the VC-dimension of the new classifier. Our experimental results on 23 standard problems show that quadratic boosting compares favorably with AdaBoost on large datasets at the cost of training speed. The classification time of the two algorithms, however, is equivalent.

7.2 Discussion

We set out to study spatial relationships within the object appearance for recognition. In particular, the problem is investigated under a paradigm of learning from pictorial examples. We consider this computer vision problem at the frontier of pattern recognition research due to the complexity of the decision boundary, the data representation, the high dimensionality of the data, the huge amount of data, and constraints on learning and recognition time.

The spatial relationship in object recognition is about combining local image signs. The image signs vary from the simple pixel intensity to a complex model learned from examples. The type of spatial relationships also appears correspondingly in different forms, from a statistical model of a fixed arrangement to a deformable one. The study of spatial relationships should not be separated from that of local image signs. In particular, we have shown that learning both local image signs and spatial relations simultaneously yields an efficient recognition method.

We consider unsupervised learning and interactive learning as the next challenge in object recognition, with respect to both local image signs and spatial relationships. It has been shown in this thesis and elsewhere that what appears important as a single entity in our intuition might not be relevant in relation with other signs. Thus, a new learning paradigm is required for this purpose.

To conclude, we have advanced the understanding of spatial relationships in an endeavor for automated object recognition.

Bibliography

- [1] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In Anders Heyden, Gunnar Sparr, Mads Nielsen, and Peter Johansen, editors, *Proceedings of the 7th European Conference on Computer Vision*, volume 4, pages 113–130, 2002.
- [2] M. Anthony and N. Biggs. *Computational Learning Theory*. Cambridge University Press, 1992.
- [3] A. J. Baddeley and M. N. M. van Lieshout. Recognition of overlapping objects using markov spatial processes. Technical report, CWI, 1991. BS-R9109, ISSN 0924-0659, 35 pages.
- [4] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting and variants. *Machine Learning*, 36(1/2):105 – 139, 1999.
- [5] E. B. Baum and D. Haussler. What size net gives valid generalization? *Neural Computation*, 1(1):151–165, 1989.
- [6] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.
- [7] Y. Bengio, J.-F. Paiement, P. Vincent, O. Delalleau, N. Le Roux, and M. Ouimet. Out-of-sample extensions for lle, isomap, mds, eigenmaps, and spectral clustering. In *Advances in Neural Information Processing Systems 16*, 2004.
- [8] C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998. University of California, Irvine, Dept. of Information and Computer Sciences.
- [9] F. L. Bookstein. Size and shape spaces for landmark data in two dimensions (with discussion). *Statistical Science*, 1(2):181–242, 1986.
- [10] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.
- [11] L. Breiman. Arcing classifiers (with discussion). *The Annals of Statistics*, 26(3):801–849, 1998.
- [12] H. Bunke. Graph matching for visual object recognition. *Spatial Vision*, 13:335–340, 2000.
- [13] M. Burl and P. Perona. Recognition of planar object classes. In *Proc. of CVPR’96*, pages 223–230, 1996.

- [14] S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal of Scientific Computation*, 20(1):33–61, 1998.
- [15] A. Colmenarez and T. Huang. Face detection with information-based maximum discrimination. In *Proc. of CVPR'97*, pages 782–787, 1997.
- [16] T.F. Cootes, C.J. Taylor, D.H. Cooper, and J. Graham. Training models of shape from sets of examples. In D. C. Hogg and R. D. Boyle, editors, *Proc. British Machine Vision Conference*, pages 9–18. Springer-Verlag, 1992.
- [17] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.
- [18] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal statistical Society, Series B*, 39(1):1–38, 1977.
- [19] T. G. Dietterich. Machine learning research: Four current directions. *AI Magazine*, 18(4):97–136, 1997.
- [20] I. L. Dryden and K. V. Mardia. General shape distributions in a plane. *Adv. Appl. Probab.*, 23:259–276, 1991.
- [21] R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, Inc., 1973.
- [22] M. Egenhofer. Query processing in spatial-query-by-sketch. *Journal of Visual Languages and Computing*, 8(4):403–424, 1997.
- [23] P. Felzenszwalb and D. Huttenlocher. Efficient matching of pictorial structures. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 66–75, 2000.
- [24] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 264–271, 2003.
- [25] M. A. Fischler and R. A. Elschlager. The representation and matching of pictorial structure. *IEEE Trans. on Computers*, 22(1):67–92, 1973.
- [26] D. Forsyth and M. Fleck. Automatic detection of human nudes. *International Journal of Computer Vision*, 32/1:63–77, 1999.
- [27] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the 13th International Conference on Machine Learning*, pages 148–156, 1996.
- [28] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.
- [29] J. H. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 28(2):337–374, 2000.

- [30] D. M. Gavrilă and V. Philomin. Real-time object detection for smart vehicles. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 87–93, 1999.
- [31] D. Geman and B. Jedynak. An active testing model for tracking roads in satellite images. *IEEE PAMI*, 18(1):1–14, 1996.
- [32] S. Geman and D. Geman. Stochastic relaxation, Gibbs distribution and the Bayesian restoration of images. *IEEE PAMI*, 6(6):721–741, November 1984.
- [33] J. M. Geusebroek, A. W. M. Smeulders, and J. van de Weijer. Fast anisotropic gauss filtering. *IEEE Trans. Image Processing*, 12(8):938–943, 2003.
- [34] S. Ghebreab and A.W.M. Smeulders. Strings: Variational deformable models of multivariate ordered features. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(11):1–14, 2003.
- [35] M. Gondran and M. Minoux. *Graphs and Algorithms*. John Wiley & Sons, 1984.
- [36] U. Grenander and A. Srivastava. Probability models for clutter in natural images. *IEEE PAMI*, 23(4):424–429, 2001.
- [37] W. E. L. Grimson. *Object Recognition by Computer: The Role of Geometric Constraints*. The MIT Press, 1990.
- [38] R. C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11(1):63–90, 1993.
- [39] A. K. Jain, R. P. W. Duin, and J. Mao. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):4–37, 2000.
- [40] M. I. Jordan, editor. *Learning in Graphical Models*. MIT Press, 1999.
- [41] D. G. Kendall. A survey of the statistical theory of shape. *Statistical Science*, 4(2):87–120, 1989.
- [42] E. Klassen, A. Srivastava, W. Mio, and S. H. Joshi. Analysis of planar shapes using geodesic paths on shape spaces. *IEEE PAMI*, 26(3):372–383, 2004.
- [43] D. E. Knuth. *The Art of Computer Programming: Seminumerical Algorithms*, volume 2. Addison-Wesley, Reading, Massachusetts, 3rd edition, 1997.
- [44] J.J. Koenderink and A.J. van Doorn. Representation of local geometry in the visual system. *Biological Cybernetics*, 55:367–375, 1987.
- [45] S. Kullback. *Information Theory and Statistics*. John Wiley, 1959.
- [46] G. H. Landeweerd, T. Timmers, E. S. Gelsema, M. Bins, and M. R. Halie. Classification of normal and abnormal samples of peripheral blood by linear mapping of the feature space. *Pattern Recognition*, 16:319–326, 1983.
- [47] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

- [48] T.K. Leung, M.C. Burl, and P. Perona. Finding faces in cluttered scenes using random labeled graph matching. In *Proc. of ICCV'95*, pages 637–644, 1995.
- [49] X. Liu, A. Srivastava, and K. Gallivan. Optimal linear representations of images for object recognition. *IEEE Transactions on PAMI*, 26(5):662–666, 2004.
- [50] D. G. Luenberger. *Linear and Nonlinear Programming*. Addison-Wesley, 1984.
- [51] J. MacCormick and A. Blake. A probabilistic contour discriminant for object localisation. In *Proc. 6th Int. Conf. on Computer Vision*, pages 390–395, 1998.
- [52] S. Mallat. *A wavelet tour of signal processing*. Academic Press, 1999.
- [53] S. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.
- [54] M. I. Miller, U. Grenander, J. A. O'Sullivan, and D. L. Snyder. Automatic target recognition organized via jump-diffusion algorithms. *IEEE Transactions on Image Processing*, 6(1):157–174, January 1997.
- [55] M. Minsky and S. Papert. *Perceptrons*. MIT Press, 1969.
- [56] B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *IEEE PAMI*, 19(7):696–710, 1997.
- [57] A. Mohan, C. Papageorgiou, and T. Poggio. Example-based object detection in images by components. *IEEE PAMI*, 23(4):349–361, 2001.
- [58] S. Mukherjee and S. K. Nayar. Automatic generation of RBF networks using wavelets. *Pattern Recognition*, 29:1369–1383, 1996.
- [59] H. Murase and S. Nayar. Visual learning and recognition of 3-D objects from appearance. *International Journal of Computer Vision*, 14:5–24, 1995.
- [60] S. A. Nene, S. K. Nayar, and H. Murase. Columbia object image library (COIL-100). Technical Report Technical Report CUCS-006-96, Columbia University, 1996.
- [61] H. Ney. On the probabilistic interpretation of neural network classifiers and discriminative training criteria. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):107–119, 1995.
- [62] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In *Proc. of CVPR'97*, Puerto Rico, 1997.
- [63] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In *Proc. of CVPR'97*, pages 130–136, Puerto Rico, 1997.
- [64] C. Papageorgiou and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1):15–33, 2000.
- [65] V. Pavlovic, A. Garg, J. Rehg, and T. Huang. Multimodal speaker detection using error feedback dynamic Bayesian networks. In *Proceedings of CVPR*, volume II, pages 34–41, 2000.

- [66] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA, 1988.
- [67] T. V. Pham and A. W. M. Smeulders. Statistical strategy for object class recognition using part detectors. In A. Leonardis and H. Bischof, editors, *Proc. of Statistical Learning in Computer Vision*, pages 33–48, 2004.
- [68] T. V. Pham, M. Worring, and A. W. M. Smeulders. Face detection by aggregated Bayesian network classifiers. *Pattern Recognition Letters*, 23(4):451–461, 2002.
- [69] P. J. Phillips. Matching pursuit filters applied to face identification. *IEEE Transactions on Image Processing*, 7(8):1150–1164, 1998.
- [70] T. Poggio and S. Edelman. A network that learns to recognize three-dimensional objects. *Nature*, 343:263–266, 1990.
- [71] T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990.
- [72] M. Pontil and A. Verri. Support vector machines for 3-D object recognition. *IEEE Transactions on PAMI*, 20(6):637–646, 1998.
- [73] W. K. Pratt. *Digital image processing*. Wiley, second edition, 1991.
- [74] J. R. Quinlan. Bagging, boosting, and c4.5. In *Proceedings 13th American Association for Artificial Intelligence National Conference on Artificial Intelligence*, pages 725–730, 1996.
- [75] J. Rocha and T. Pavlidis. A shape analysis model with applications to a character recognition system. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(4):393–404, 1994.
- [76] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.
- [77] H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE PAMI*, 20(1):23–38, 1998.
- [78] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 1995.
- [79] N. Sauer. On the density of families of sets. *Journal of Combinatorial Theory (series A)*, 13:145–147, 1972.
- [80] J. L. Schafer. *Analysis of Incomplete Multivariate Data*. Chapman & Hall, 1997.
- [81] F. Schaffalitzky and A. Zisserman. Viewpoint invariant texture matching and wide baseline stereo. In *ICCV*, pages 636–643, 2001.
- [82] R. E. Schapire. The boosting approach to machine learning: An overview. In *MSRI Workshop on Nonlinear Estimation and Classification*, 2002.
- [83] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics*, 26(5):1651–1686, 1998.

- [84] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.
- [85] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–535, 1997.
- [86] H. Schneiderman and T. Kanade. A statistical method for 3d object detection applied to faces and cars. In *Proceedings of CVPR*, pages 746–751, 2000.
- [87] B. Schölkopf, A. J. Smola, and K. R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.
- [88] T. B. Sebastian, P. N. Klein, and B. B. Kimia. Recognition of shapes by editing their shock graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5):550–571, 2004.
- [89] K. Siddiqi, A. Shokoufandeh, S. Dickinson, and S. Zucker. Shock graphs and shape matching. *International Journal of Computer Vision*, 30:1–24, 1999.
- [90] A. J. Smola and B. Scholkopf. From regularization operators to support vector kernels. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10, pages 343–349. The MIT Press, 1998.
- [91] M. S. Srivastava and C. G. Khatri. *An Introduction to Multivariate Statistics*. North Holland, 1979.
- [92] L. Staib and J. Duncan. Boundary finding with parametrically deformable models. *IEEE PAMI*, 14(11):1061–1075, 1992.
- [93] J. Sullivan, A. Blake, M. Isard, and J. MacCormick. Bayesian object localisation in images. *International Journal of Computer Vision*, 44(2):111–135, 2001.
- [94] K. K. Sung and T. Poggio. Example-based learning for view-based human face detection. *IEEE PAMI*, 20(1):39–51, 1998.
- [95] H. Tagare, F. Vos, C. C. Jaffe, and J. S. Duncan. Arrangement: A spatial relation between parts for evaluating similarity of tomographic section. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(9):880–893, 1995.
- [96] R. Tarjan. Finding optimum branchings. *Networks*, 7:25–35, 1977.
- [97] D. M. J. Tax and R. P. W. Duin. Support vector domain description. *Pattern Recognition Letters*, 20(11-13):1191–1199, 1999.
- [98] J. B. Tenenbaum and V. de Silva and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.
- [99] K. Tieu and P. Viola. Boosting image retrieval. In *Proceedings of CVPR*, volume I, pages 228–235, 2000.
- [100] M. A. Turk and A. P. Pentland. Face recognition using eigenfaces. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

- [101] M. Unser, A. Aldroubi, and M. Eden. B-spline signal processing: Part I - theory. *IEEE Transactions on Signal Processing*, 41(2):821–833, 1993.
- [102] M. Unser, A. Aldroubi, and M. Eden. B-spline signal processing: Part II - efficient design and applications. *IEEE Transactions on Signal Processing*, 41(2):834–848, 1993.
- [103] L. J. van Vliet, I. T. Young, and P. W. Verbeek. Recursive Gaussian derivative filters. In *Proceedings of the 14th ICPR*, volume I, pages 509–514, 1998.
- [104] V. N. Vapnik. *Estimations of dependences based on statistical data*. Springer-Verlag, 1982.
- [105] V. N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, Inc, 1998.
- [106] N. Vasconcelos, P. Ho, and P. Moreno. The Kullback-Leibler kernel as a framework for discriminant and localized representations for visual recognition. In *Proceedings of the 8th European Conference on Computer Vision*, volume III, pages 430–441, 2004.
- [107] P. Viola and M. Jones. Robust real-time object detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.
- [108] M. Weber, M. Welling, and P. Perona. Unsupervised learning of models for recognition. In *Proc. of ECCV*, pages 18–32, 2000.
- [109] G. Wei and I. K. Sethi. Face detection for image annotation. *Pattern Recognition Letters*, 20(11-13):1313–1321, 1999. Special issue on Pattern Recognition in Practice VI, Vlieland, June 99. Editors E. S. Gelsema and J. F. Veenland.
- [110] W. M. Wells. Efficient synthesis of Gaussian filters by cascaded uniform filters. *IEEE Transactions on PAMI*, 8(2):234–239, 1986.
- [111] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999.
- [112] J. Yang and A. Waibel. A real-time face tracker. In *Proc. of WACV'96*, pages 142–147, 1996.
- [113] K.C. Yow and R. Cipolla. Scale and orientation invariance in human face detection. In *Proc. 7th British Machine Vision Conference*, pages 745–754, 1996.
- [114] A. L. Yuille and J. Coughlan. An A* perspective on deterministic optimization for deformable templates. *Pattern Recognition*, 33(4):603–616, April 2000.
- [115] S. Zhu and D. Mumford. Prior learning and Gibbs reaction-diffusion. *IEEE PAMI*, 19(11):1236–1250, November 1997.

Samenvatting*

Dit proefschrift behandelt het probleem van het herkennen van objecten op basis van een verzameling voorbeeldafbeeldingen. Voor dit doel richten we ons bovenal op de spatiele relaties binnen de verschillende verschijningsvormen van objecten. We onderzoeken nieuwe statistische modellen die in vergelijking met bestaande modellen krachtiger zijn, sneller leren, of juist sneller herkennen. We bestuderen tevens het probleem van niet-zichtbare objectdelen en onjuiste herkenning. De klasse van op objectonderdelen gebaseerde methoden is belangrijk, daar de structuur van objecten expliciet wordt gemodelleerd, wat een gedetailleerde en efficiënte herkenning oplevert. Tot slot onderzoeken we een nieuwe, op meerdere lokale aanwijzingen gebaseerde objectrepresentatie.

Hoofdstuk 2 beschrijft een methode voor gezichtsherkenning. Het leert een gezichtsclassificator op basis van een grote verzameling afbeeldingen van zowel gezichten als andere objecten. De classificator wordt samengesteld door aggregatie van Bayesiaanse netwerk classificators. Het gebruik van classifier-aggregatie is geschikt voor dit specifieke herkenningsprobleem, waarin een zeer lage mate van foutclassificatie vereist is. Onze methode presteert beter dan een gerelateerde, op een Markov proces-model gebaseerde methode: het biedt namelijk een grote klasse van afhankelijkheden tussen input variabelen, met een betere algoritmische complexiteit van de leerfase en identieke herkenningssnelheid. Het gezichtsherkenningssysteem levert resultaten die net zo accuraat zijn als die van andere kwalitatief hoogstaande methoden.

Hoofdstuk 3 beschrijft een methode voor objectklasse herkenning gebaseerd op eerder gedetecteerde objectdelen. Deze herkenningstaak is geformuleerd als een graaf over de karakteristieke geografische indeling van (mogelijk ontbrekende) onderdelen. De objectieve functie is een Bayesiaanse maximale a-posteriori schatter, waarbij de beeldwaarschijnlijkheid geïntegreerd is als een a-posteriori kansverdeling van de objectdeeldetectors. De variatie in de verdeling van de objectdelen is gemodelleerd als een Gaussische verdeling na een translatie normalisatie. Het voordeel van onze aanpak boven bestaande methodes gebaseerd op objectdeel herkenning is tweeledig. Ten eerste, door twee speciale eigenschappen van de Gaussische verdeling te gebruiken kunnen we omgaan met het ontbreken van onderdelen, zelfs in situaties waar de oorsprong voor de normalisatie niet gedetecteerd is. Ten tweede kunnen we de optimale oplossing voor het graaf zoekprobleem vinden met behulp van een elegant A^* algo-

*Summary in Dutch

ritme. Experimenten op zowel synthetische als reële gegevens demonstreren goede resultaten, en een goede herkenningssnelheid

Hoofdstuk 4 bestudeert twee typen van spatiele relaties, geleerd door trainingsvoorbeelden, voor objectherkenning. Het eerste type, behandeld in hoofdstuk 2, beschrijft een fijnkorrelige spatiele relatie tussen pixels. Het tweede type, beschreven in hoofdstuk 3, modelleert grofkorrelige spatiele relaties tussen objectdelen. We gebruiken twee verschillende gegevensbanken: de ene bevat frontale aanzichten van menselijke gezichten, als voorbeeld van de objectklasse met een vaste spatiele lay-out; de andere bestaat uit naar rechts kijkende paarden, met grote variatie in hun actie, als een voorbeeld van de uit onderdelen bestaande objecten. Door (bijna) hetzelfde aantal aan annotaties van trainingsgegevens af te dwingen, wat gezien kan worden als het dirigeren van kennis naar het leeralgoritme, kunnen we de sterke en zwakke punten van elke methode identificeren. We concluderen dat beide methoden superieur zijn aan zowel een strikte spatiele herkenning volgens een vaste structuur als aan een strikte niet-spatiele klassenherkenner.

Hoofdstuk 5 behandelt een schaarse, op meerdere schalen beschreven object-representatie. Het ondervangt objectverschijningen door te kiezen uit een groot woordenboek bestaande uit basisfuncties van Gaussische afgeleiden. De leerprocedure is gebaseerd op een vergelijk-en-achtervolg algoritme, terwijl herkenning gebaseerd is op polynomiale benadering tot de basis, wat het vergelijken van beelden reduceert tot polynoom evaluatie. De methode is geschikt voor een ruwe herkenning tussen objecten en, door meer bases te gebruiken, ook bruikbaar voor gedetailleerde herkenning van de objectpose. De grootste voordelen boven de gebruikelijke representaties gebaseerd op PCA zijn driedelig. Ten eerste is het opslaan van steekproefpunten niet nodig, ten tweede is het model makkelijk uitbreidbaar, aangezien andere objectmodellen niet hertrainend hoeven te worden, en ten derde is de representatie, in tegenstelling tot PCA, al aanwezig in geval het beeld doorzocht moet worden. Experimentele resultaten op de COIL-100 gegevensbank laat zien dat we een hoge herkenningsgraad combineren met een onmiddellijke reactiesnelheid.

Hoofdstuk 6 presenteert een strategie om het AdaBoost algoritme te verbeteren met een kwadratische combinatie van basis klassen-herkenners. We observeren dat het leren van deze combinatie noodzakelijk is voor verbeterde resultaten en mogelijk wordt gemaakt door het maken van een tussen-leeralgoritme welke opereert op de gecombineerde lineaire en kwadratische termen. Dit is geen triviale taak, aangezien de parameters van de basis klassen-herkenners niet onder directe controle staan, waardoor een directe optimalisatie methode niet mogelijk is. We presenteren een nieuwe methode voor indirecte iteratieve optimalisatie. Ten eerste trainen we een klassen-herkenner op willekeurige etiketten van trainingsvoorbeelden. Vervolgens wordt het initiele leeralgoritme herhaaldelijk aangeroepen met een systematisch vernieuwing van de etiketten van de trainingsvoorbeelden in elke ronde. We laten zien dat het kwadratische boosting algoritme convergeert onder de aannamen dat het gegeven basis leeralgoritme de empirische fout minimaliseert. Bovendien geven we een bovengrens voor de VC-dimensie van de nieuwe klassenherkenner. Onze experimentele resultaten voor 23 standaard problemen tonen dat kwadratische boosting beter werkt dan AdaBoost op grote gegevensbanken, ten koste van de leersnelheid. Desalniettemin is de

voor klassenindeling benodigde tijd identiek.

Discussie

We bestuderen spatiele relaties binnen object verschijningsvormen voor een herkenningstaak. In het bijzonder is het probleem gemodelleerd naar aanleiding van automatisch leren van voorbeeldafbeeldingen. We beschouwen dit computerperceptieprobleem op de frontlinie van het patroonherkenningsonderzoek vanwege de complexiteit van de beslissingsgrens, de representatie van de gegevens, de hoge dimensionaliteit van de gegevens, de grote hoeveelheden aan gegevens en de beperkingen aan de leer- en herkenningstijdsduur.

De spatiele relatie binnen de objectherkenning wordt gelegd door een combinatie van lokale aanwijzingen. Er is grote variatie tussen verschillende afbeeldingaanwijzingen, van de simpele pixelintensiteit tot een complex model geleerd van voorbeelden. De typen spatiele relaties hebben ook verschillende verschijningsvormen, van een statistisch model van vaste posities tot een vervormbaar model. Het bestuderen van spatiele relaties dient niet onafhankelijk van de lokale afbeeldingaanwijzingen te worden gedaan. In het bijzonder, hebben we aangetoond dat het gemeenschappelijk leren van lokale afbeeldingaanwijzingen met spatiele relaties een efficiënte herkenningsmethode oplevert.

We beschouwen het leren zonder voorbeelden en interactief leren als de nieuwe uitdaging in objectherkenning, met betrekking tot lokale afbeeldingaanwijzingen als spatiele relaties. Zowel in dit proefschrift als op andere plekken is aangetoond dat wat intuïtief belangrijk lijkt als een alleenstaande entiteit niet relevant hoeft te zijn in relatie met andere aanwijzingen. Ergo, voor dit doel is een nieuw leerparadigma noodzakelijk.

Tot slot concluderen we dat we vooruitgang hebben geboekt in het begrip van spatiele relaties in een exercitie in automatische objectherkenning.

Acknowledgements

First and foremost, I thank my supervisor and promoter Arnold Smeulders who has reserved an immeasurably large amount of time for me and my scientific work. I have learned much from his penetrating insights in science and life over the years. Arnold, thank you for your support, for the many lessons, and for showing me what I was trying to do from my seemingly endless stream of random thoughts.

I thank Marcel Worrying for his guidance in the first year of my PhD research. Marcel brought me here to the ISIS group, helped me settle down in Amsterdam, and ever since has been a constant source of support. Marcel also contributed substantially to the content of chapter two of the thesis.

I wish to express my gratitude to members of my doctoral committee for reviewing this thesis.

I wish to acknowledge the financial support from our industrial partner TNO-TPD. I thank Jurgen and Nellie of TNO-TPD who together with Marcel wrote the initial research proposal. They both have been very supportive during the whole research project.

I thank my compatriots Hieu, Minh and Giang with whom I share many memories in working and living in Amsterdam. A special thank goes to Hieu who was particularly kind and helpful when I first arrived in Amsterdam.

ISIS members have contributed to this thesis in many ways, and I thank them all. My officemate Frank is always there whenever I need help. He also has to put up with all the mess around our office. Joost, Andy, Aristeidis and Jan-Mark deserve special thanks for their help, advices, and many technical and non-technical discussions. Jan wrote more than two pages in the Dutch translation of this thesis. Dennis has always been helpful, especially in building demonstrators. Marc was encouraging to ask for a copy of the thesis four years ago. Jeroen was very kind to give me his bicycle when he left. Sennay was willing to read and discuss my paper. I thank Theo for asking me to play in his football team from time to time. I thank Rein for his kind words regarding my research on many occasions. I thank Cees for his willingness to share (others') computing power. Gertjan and Michiel keep reminding me of lunch patiently even though I rarely join in. Many others I wish to mention for the many, however short, conversations: Frank Aldershoff, Carlo, Wilko, Erik, Kees, Frans, Ioannis, Edo, Leon, and Nicu. Finally, I thank Virginie for her help in practical matters.

I thank IAS members for the thought-provoking Tuesdays, in particular Ben Kröse and Nikos Vlassis for their interest in my research.

I thank all teachers I have had.

I thank my friends for the many get-togethers, and for their email messages.

Now just before the completion of this thesis, it is time to thank the ones who are most important to me. My father and my mother, Pham Van Tham and Pham Thi Thoa, have made it all possible by working so hard for so many years, and I can never thank them enough. My sister Hoa always looks after me with her wonderful personality and positive thinking. She has helped me overcome many difficult moments during this research. My sister Chi makes everyday life a bit more exciting, especially when we watch football. A special thank goes to my parents-in-law and my sister-in-law for their encouragement and kindness. And finally, to my brave and beautiful wife Duong who takes care of me with her smile, love and understanding in all steps along the way, and with who I look forward to the future, thank you.