

Patch-based Models For Visual Object Classes

Jania Aghajanian

A dissertation submitted in partial fulfilment

of the requirements for the degree of

Doctor of Philosophy

at

University College London

Department of Computer Science

University College London

February 24, 2011

*This thesis is dedicated to the loving memory of my grandfather
Abraham Aghajanian*

I, Jania Aghajanian, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

This thesis concerns models for visual object classes that exhibit a reasonable amount of regularity, such as faces, pedestrians, cells and human brains. Such models are useful for making “within-object” inferences such as determining their individual characteristics and establishing their identity. For example, the model could be used to predict the identity of a face, the pose of a pedestrian or the phenotype of a cell and segment parts of a human brain.

Existing object modelling techniques have several limitations. First, most current methods have targeted the above tasks individually using object specific representations; therefore, they cannot be applied to other problems without major alterations. Second, most methods have been designed to work with small databases which do not contain the variations in pose, illumination, occlusion and background clutter seen in ‘real world’ images. Consequently, many existing algorithms fail when tested on unconstrained databases. Finally, the complexity of the training procedure in these methods makes it impractical to use large datasets.

In this thesis, we investigate patch-based models for object classes. Our models are capable of exploiting very large databases of objects captured in uncontrolled environments. We represent the test image with a regular grid of patches from a library of images of the same object. All the domain specific information is held in this library: we use one set of images of the object to help draw inferences about others. In each experimental chapter we investigate a different within-object inference task. In particular we develop models for classification, regression, semantic segmentation and identity recognition. In each task, we achieve results that are comparable to or better than the state of the art. We conclude that patch-based representation can be successfully used for the above tasks and shows promise for other applications such as generation and localization.

Acknowledgements

First and foremost, I express my deepest thanks to my supervisor, Dr. Simon Prince, for his invaluable advice, support and encouragement throughout the last four years.

I am grateful to EPSRC and UCL-ORS for funding my studies. I am in particular grateful to Prof. Bernard Burton, who kindly helped provide funding for the first year of my studies.

I would like to thank my second supervisor Prof. Daniel Alexander for his suggestions and guidance. I would also like to thank my undergraduate supervisors Dr. Andreas Lanitis and Dr. Tasos Falas who first sparked an interest in me to pursue research in computer vision.

Many thanks to my colleagues at UCL for being such great company in this experience. In particular, I would like to thank Alastair Moore, Yun Fu, Umar Mohammed, Jonathan Warrell, Peng Li, Thomy Mertzaniidou, Oisin Mac Aodha, Sara Vicente and Laura Panagiotaki. I am thankful to them for pointing out the strengths and weaknesses of my work, for helping me understand difficult concepts and for reading earlier drafts of this thesis.

I would like to thank my examiners, Prof. Tim Cootes and Dr. Jan Kautz for their time, and constructive feedback.

Special thanks go to my brothers Jarnis and Jansel who have always believed in me and to Dumebi Okwechime for his kind and reassuring words when I most needed them.

Finally, to my parents Janet and Vigen, whose hard work and determination has taught me to never give up. Thank you. I am forever in your debt for your love and support throughout my studies. You have always been an inspiration to me.

Contents

1	Introduction	15
1.1	A Taxonomy of Object Model Tasks	15
1.2	Challenges Involved in Object Modeling	17
1.3	Constrained vs Unconstrained Databases	19
1.4	Problem Statement	20
1.5	Research Objective	20
1.6	Contributions	21
1.7	Report Structure	21
2	Background	23
2.1	Representation	23
2.1.1	Global Representations	24
	Subspace models	24
	Active Appearance Models	25
	Discussion	27
2.1.2	Local Representations	27
	Bag-of-words models	28
	Part-based models	29
2.1.3	Patch-based Representation	33
2.1.4	Summary and Discussion	34
2.2	Learning and Inference	35
2.2.1	Discriminative Models	36
2.2.2	Generative Models	37
2.2.3	Generative vs. Discriminative Approaches	38
2.2.4	Summary and Discussion	39

3	Patch-based Within-Category Classification	41
3.1	Introduction	41
3.2	Motivation	41
3.3	Method	44
3.3.1	Inference	45
3.3.2	Training	47
3.3.3	Calculation of Likelihood Integral	49
3.4	Databases	49
3.4.1	Faces	49
3.4.2	Cells	50
3.4.3	Pedestrians	51
3.5	Experiments	52
3.5.1	Experiment 1: Choosing The Best Preprocessing	52
3.5.2	Experiment 2: Parameter Selection	52
3.5.3	Experiment 3: Gender Classification	55
3.5.4	Experiment 4: Eyewear Classification	56
3.5.5	Experiment 5: Age Classification	57
3.5.6	Experiment 6: Cell Phenotype Classification	58
3.5.7	Experiment 7: Pedestrian Pose Classification	60
3.5.8	Experiment 8: Comparison to Other Algorithms	62
3.6	Summary and Discussion	63
4	Patch-based Regression	67
4.1	Introduction	67
4.2	Motivation	67
4.3	Method	69
4.3.1	Inference	70
4.3.2	Training	73
4.4	Databases	74
4.4.1	The CUBiC FacePix(30) Database: Images Captured in Controlled Environments	74
4.4.2	The UCL Database: Images Captured in Uncontrolled Environments	74
4.5	Experiments on the Constrained Database	76
4.5.1	Experiment 1: Varying the Patch-Grid Resolution	76

4.5.2	Experiment 2: Varying the parameters of the Radial Basis Functions (RBFs)	77
4.6	Experiments on the Unconstrained Database	78
4.6.1	Experiment 3: Varying the Patch-Grid Resolution	78
4.6.2	Are 6×6 pixel patches sufficient for face pose estimation?	79
4.6.3	Experiment 4: Varying the parameters of the Radial Basis Functions (RBFs)	80
4.7	Summary and Discussions	81
5	Patch-based Semantic Segmentation	86
5.1	Introduction	86
5.2	Motivation	87
5.3	The shiftmap prior: a prior over permutations of class labels	90
5.4	Semantic segmentation using the shiftmap prior	92
5.5	Optimizing the full posterior probability	94
5.5.1	Graph Construction	96
5.5.2	Graph construction for patches	99
5.5.3	Using multiple library labelmaps	101
5.6	Databases	102
5.6.1	Faces	103
5.6.2	Human brains	103
5.7	Experiments on semantic segmentation of faces	104
5.7.1	Experiment 1: Varying the parameter λ_1	105
5.7.2	Experiment 2: Varying the parameter λ_2	107
5.7.3	Experiment 3: Varying the number of top matched library labelmaps	109
5.7.4	Experiment 4: Varying the term $\theta_{correct}$	110
5.7.5	Sample Results	112
5.8	Experiments on semantic segmentation of brains	112
5.8.1	Experiment 5: Varying the parameter λ_1	114
5.8.2	Experiment 6: Varying the parameter λ_2	114
5.8.3	Experiment 7: Varying the number of top matched library labelmaps	115
5.8.4	Experiment 8: Varying the term $\theta_{correct}$	116
5.8.5	Sample Results	117
5.9	Summary and Discussion	117

6 Patch-based Identification	120
6.1 Introduction	120
6.2 Motivation	121
6.3 Latent Identity Variables (LIVs)	123
6.4 Mosaicface Model	125
6.4.1 Experiment 1 - Mosaicface Identification	126
6.4.2 Experiment 2 - Mosaicfaces with Gabor Features	128
6.5 Mosaicface Model with Multiple Patch Appearances	129
6.5.1 Learning Multiple Appearances	130
6.5.2 Experiment 3 - Modeling Illumination Change	132
6.5.3 Experiment 4 - Expression Variation	135
6.5.4 Experiment 5 - Large Image Variation	136
6.6 Summary and Discussion	136
7 Conclusions	139
7.1 Summary	139
7.2 Contributions	141
7.3 Limitations	141
7.4 Future work	142
Appendices	144
A Calculating The Dirichlet Posterior Integral	144
Bibliography	145

List of Figures

1.1	A Taxonomy of Object Model Tasks	16
1.2	Challenges involved in object recognition	18
1.3	Constrained vs Unconstrained Databases	19
2.1	Global Representation: Eigen Appearance Model	24
2.2	Active Appearance Models	26
2.3	Local Representation: Bag of Words Models	28
2.4	Local Representation: Constellation Models	30
2.5	Local Representation: Pictorial Structures	31
2.6	Local Representation: Fragment-based Models	32
2.7	Patch-based Image Representation	33
2.8	Graphical Representation of Generative and Discriminative Models	37
2.9	Generative vs Discriminative Object Class Modeling	38
3.1	Sliding Window Object Detection	42
3.2	Examples of Within-Category Classification in Uncontrolled Environments	43
3.3	Inference For Patch-based Within-Category Classification	45
3.4	A graphical representation of the patch-based classification model	46
3.5	Training for Patch-based Within-Category Classification	47
3.6	Face Preprocessing Steps	49
3.7	Cell Preprocessing Steps	50
3.8	Preprocessing Method and Parameter Selection	51
3.9	Parameter Selection for Patch-based Within-Category Classification	52
3.10	Comparison of Original Images and Best Approximations From Library Patches	54
3.11	Gender Classification Results on Manually Detected Faces	55
3.12	Eyewear Classification Results	57
3.13	Per-Patch Classification Performance for Gender, Eyewear and Pedestrian Pose	58
3.14	Cell Phenotype Classification Results	60

3.15 Pedestrian Pose Classification Results	61
4.1 Estimation of Face Pose: A Classification vs Regression Problem	68
4.2 Inference for Patch-based Continuous Face Pose Estimation	70
4.3 Training for Patch-based Continuous Face Pose Estimation	71
4.4 Radial Basis Functions(RBFs) used in Regression	72
4.5 Human vs. Computer Pose Estimation Error	75
4.6 Some example images along with the average human estimate of pose	76
4.7 Varying the Parameters of Pose Estimation on Constrained Database	77
4.8 Face Pose Estimation Results on a Constrained Database Using 10×10 Patches.	78
4.9 Posterior over Pose for Example Images from the Constrained Database	79
4.10 Example Images from the Constrained Database with Estimated Pose	79
4.11 Comparison of Performance on Pose Estimation by Varying the Parameters	80
4.12 Reconstruction of the original images using the closest patches from the library	81
4.13 Scatter Plot of the Face Pose Estimation Results on an Unconstrained Database	82
4.14 Results on unconstrained database by Varying σ	82
4.15 Average Error for Each Pose Angle	83
4.16 Example Images with True and Estimated Pose	83
5.1 Examples of Semantic Segmentation.	87
5.2 Overview of The Shiftmap Prior for Semantic Segmentation	88
5.3 Shiftmaps for Image Retargeting to Reduce Width.	90
5.4 Class Label Priors Conditioned on the Shiftmap	93
5.5 The Complete Semantic Segmentation Algorithm	95
5.6 Alternating Optimization of the Energy for Vertical and Horizontal Shifts	96
5.7 Graph Construction for Shiftmap Prior	97
5.8 Imposing Sequential Constraints	98
5.9 Patch-based Shiftmaps for Image Retargeting to Reduce Width.	99
5.10 Graph Construction for the Patch-based Model	100
5.11 Face Library Images and Labelmaps	103
5.12 Brain Library Images and Labelmaps	104
5.13 The Effect of the parameters λ_1 and λ_2 on Face Part Segmentation	106
5.14 Qualitative Results of Varying the Parameters λ_1 on Semantic Segmentation of Faces	107
5.15 Qualitative Results of Varying Patch Sizes on Semantic Segmentation of Faces	108

5.16	Qualitative Results of Varying the Parameters λ_2 on Semantic Segmentation of Faces	109
5.17	The Effect of the Number of Top Matched Library Images and the term $\theta_{correct}$ on Semantic Segmentation Faces	110
5.18	Example Results on Semantic Segmentation of Faces using the Shiftmap Prior .	111
5.19	The Effect of the parameters λ_1 and λ_2 on Semantic Segmentation of Human Brains	113
5.20	The Effect of the Number of Top Matched Library Images on Semantic Segmentation of Brains	115
5.21	Example Results on Semantic Segmentation of Brains using the Shiftmap Prior	116
6.1	Novel Patch-based Representation for Face Recognition	122
6.2	Face Identification by Model Comparison	124
6.3	Learnt Patch Library for Face Identification	125
6.4	Maximum Likelihood (ML) Reconstruction of Probe and Gallery Images . . .	127
6.5	Face Identification Results on XM2VTS Database	128
6.6	Identification Performance for Mosaicface Model on Gabor-filtered XM2VTS Database	129
6.7	Face Identification by Model Comparison	130
6.8	Identification Results for Multiple-Appearance Model on XM2VTS Database .	131
6.9	Mosaicface Multiple Appearance Model: Illumination Variation	133
6.10	Results on XM2VTS Lighting Database	134
6.11	Mosaicface Multiple Appearance Model: Expression Variation	134
6.12	Identification Performance of Multiple-Appearance Model on FERET fa/fb dataset	135

List of Tables

2.1	Areas of successful application of patch-based representation with generative inference.	39
3.1	Confusion matrix for age classification	59
3.2	Comparison of performance for age classification	59
3.3	Confusion matrix for pedestrian pose classification.	62
3.4	Confusion matrix for pedestrian pose classification.	63
5.1	The average $F_{0.5}$ measure for the results on semantic segmentation of faces across 50 test images.	112
5.2	The average $F_{0.5}$ measure for results of semantic segmentation of brains across 50 test images.	117

Publications

Parts of this thesis have been published in peer-reviewed conferences. The full list of publications is as follows:

Chapter 3

J. Aghajanian, J. Warrell, S.J.D. Prince, P. Li and J.L. Rohn and B.Baum “Patch-based Within-Object Classification,” *ICCV*, 2009

Chapter 4

J. Aghajanian and S.J.D. Prince, “Face pose estimation in uncontrolled environments”, *BMVC*, 2009

P. Li, J. Warrell, J. Aghajanian and S.J.D. Prince, “Context based additive logistic model for facial keypoint localization,” *BMVC*, 2010

Chapter 6

J. Aghajanian and S.J.D. Prince, “Mosaicfaces: a discrete representation for face recognition,” *WACV*, 2008.

S.J.D. Prince, J. Aghajanian, U. Mohammed and M. Sahani, “Latent Identity Variables: Biometric Matching without Explicit Identity Estimation,” *ICB*, 2007.

Chapter 1

Introduction

“*Vision is the process of discovering from images what is present in the world, and where it is*” (David Marr). Computer vision practitioners are interested in writing computer algorithms to discover objects present in images by mimicking human vision. For a computer to be able to detect, identify and draw inferences about an object, a statistical model that describes the visual appearance of the object is required. This thesis concerns statistical models of this type.

1.1 A Taxonomy of Object Model Tasks

We aim to develop a single object model that is flexible enough to tackle a wide variety of object related tasks. Consequently, we first describe these tasks in turn using the example of a face model.

The problem of object *Detection* is to determine the presence of an object in an image. Typically, object detectors consider small image windows at all locations and scales and perform a binary detection for each. The output of a detector is usually a bounding box around the object of interest; e.g. for the face class, we might require the system to return a bounding box around each face region (figure 1.1a).

The goal of *Segmentation* is to provide the precise location of the object boundaries, by dividing the image into foreground (the desired object) and background (everything else). For example, given an image containing a face, find the best division of face pixels from the background (figure 1.1b). Subsequently, *semantic image segmentation* is to label different parts of a single object. For example, given a face image, label the pixels as either eyes, nose, mouth, skin etc. (figure 1.1c).

The goal of *Identity recognition* is to recognize the identity of a particular instance of an object. For example, given an individual’s face image, the model would find the closest match in a large database (figure 1.1d). *Invariant recognition* is the ability to identify an object regardless of possible variations in different images of the same object; e.g. to recognize a face,

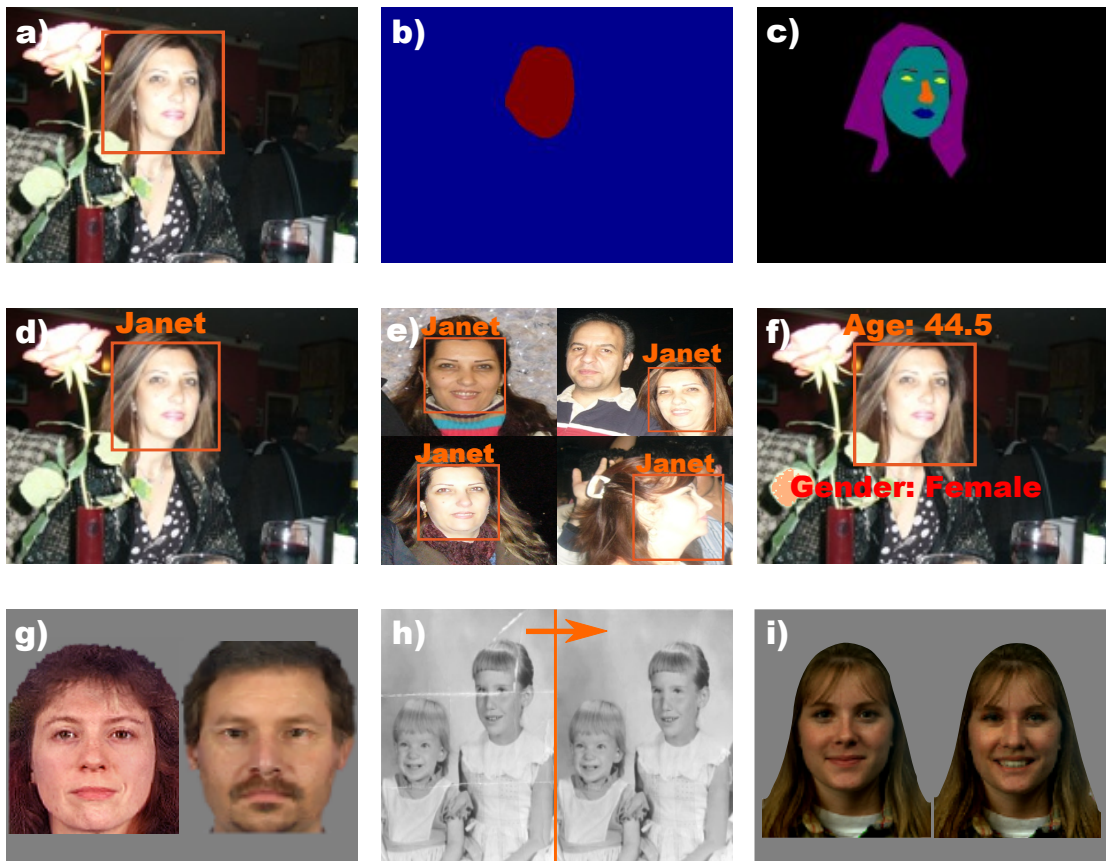


Figure 1.1: *Object-related tasks with examples from the face class* a) *Detection*: place a bounding box around the faces. b) *Segmentation*: divide the image into face (foreground) and non-face (background) pixels. c) *Semantic segmentation*: label different parts of the face e.g. nose, mouth, etc. d) *Identity recognition*: recognize the identity of the given face. e) *Invariant recognition*: recognize the person despite the changes in appearance. f) *Regression*: predict continuous attributes e.g. age. *Classification*: assign the face to one of many classes, e.g. gender classification. g) *Generation*: generate new photorealistic faces. h) *Restoration*: given a noisy face in an old photograph (left), generate a clean face (right). i) *Attribute interchange*: given a neutral face, produce a smiling face. Images in (g,i) adapted from [125] and (h) adapted from [148].

in different poses (figure 1.1e).

Regression is the process of finding a functional relationship between the appearance of the object and its continuous attributes. For example, given a face, the model would predict the age of the person (figure 1.1f). *Classification* is the task of assigning an object to one of many discrete classes. For instance, to assign an object as belonging to either of the car, face or human body categories. Alternatively, one can classify attributes *within a category*, such as predicting gender for a face image (figure 1.1f).

Generation is the process of generating new instances of the object from the learned model, which do not exist in the training database. An example would be to generate random photo-realistic images of new faces which were not in the training set (figure 1.1g).

Image *Restoration* algorithms enhance the condition of the object in an image. For example, given a noisy or low resolution face in an old photograph, generate a clean and high resolution face (figure 1.1h). *Attribute interchange*, is the process of changing an attribute such as the pose, lighting or material of the object. For example, in face images we might learn the relationship between neutral and smiling faces and use this to turn a neutral face into a smiling face (figure 1.1i).

1.2 Challenges Involved in Object Modeling

An ideal object model would support the tasks described in section 1.1. Unfortunately, this is extremely challenging for a number of reasons which we now explore.

A major complication in object recognition is the *intra-class variability*. The visual properties of different objects belonging to the same class can vary considerably. Chairs are good examples of such class of objects (figure 1.2a).

Another impeding factor is *illumination changes*. The appearance of most objects varies significantly with changes in illumination. For example, the same face looks very different under different lighting conditions (figure 1.2b). Another issue is the *scale* at which objects appear in the image. A vision algorithm may struggle to detect or recognize objects if it analyses them at the wrong scale.

Other major issues include *viewpoint or pose variation*. When 3D objects are projected onto a 2D image, they appear quite differently depending on the position of the camera. A car for instance, may look very different from different viewpoints (see figure 1.2c).

Occlusion is another obstacle for object detection / recognition. Occluded object parts can cause problems for vision algorithms. For example in a snapshot of a tennis match, the tennis racket may partially occlude the face of the player. Consequently it may be difficult to detect or

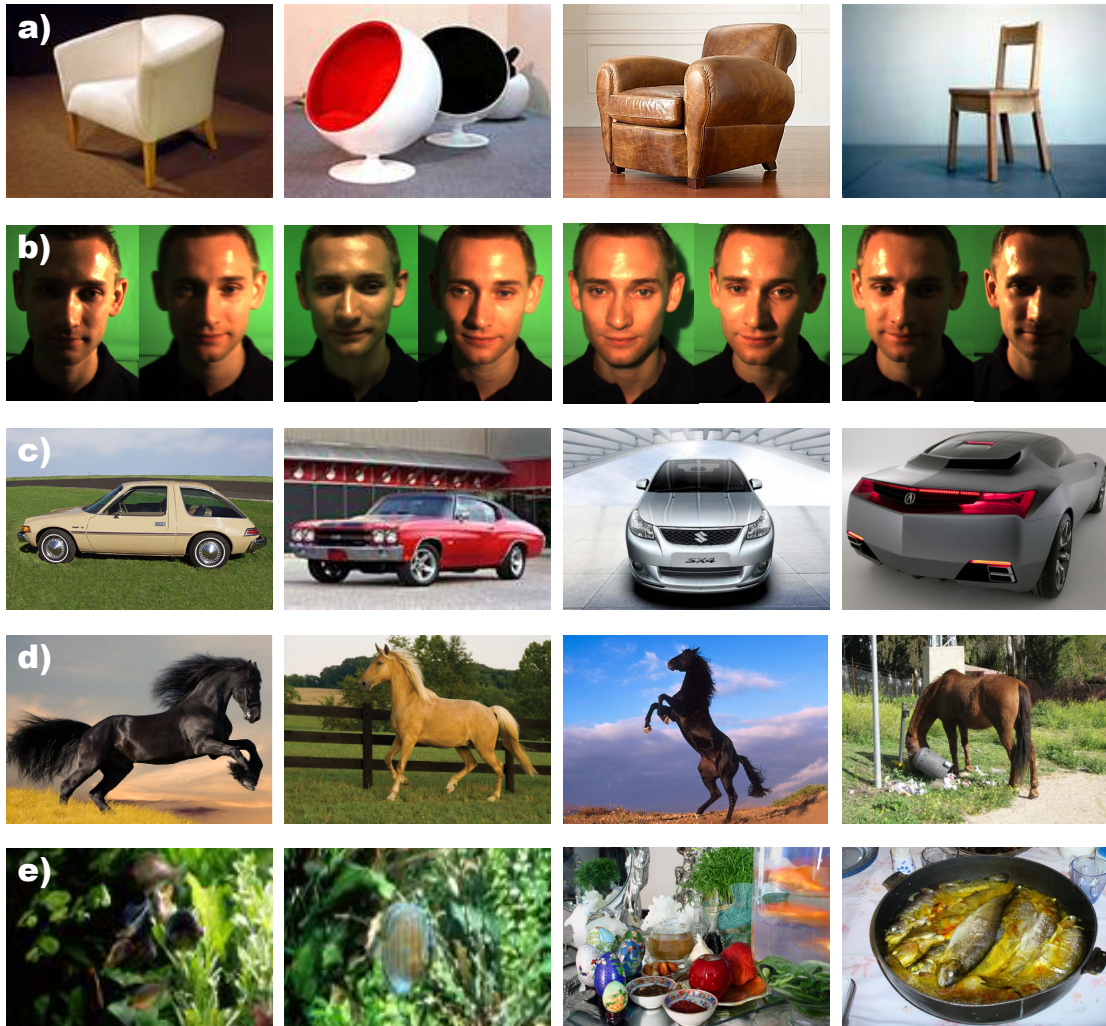


Figure 1.2: Challenges involved in object modeling: a) Intra-class variability: objects from the same class can vary considerably. b) Illumination variations: the appearance of objects varies significantly with changes in illumination. c) Pose variation: objects appear quite differently depending on the position of the camera. d) Deformation and self occlusion: some articulated objects can self occlude due to their geometric properties. e) Background clutter: cluttered background impedes vision tasks.

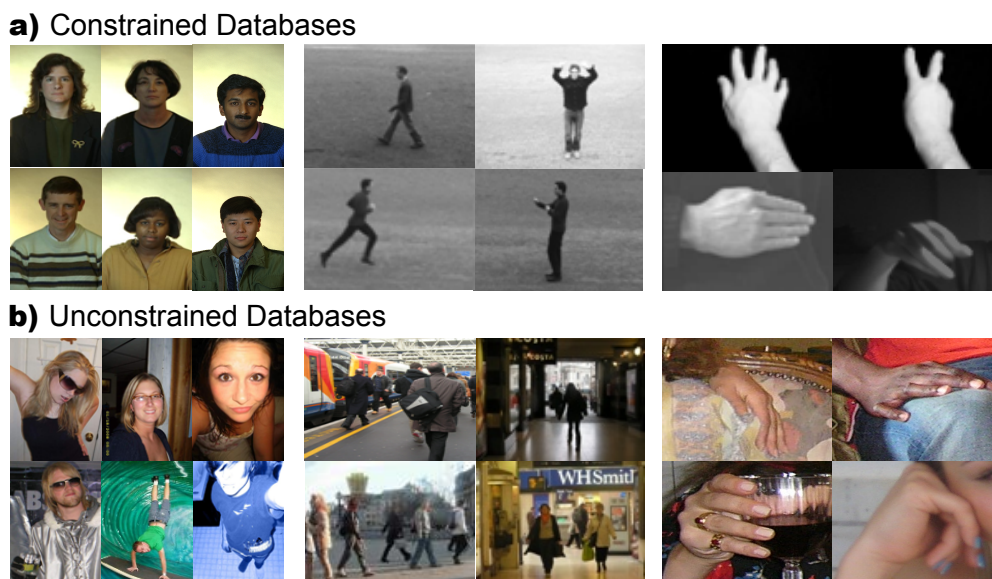


Figure 1.3: Constrained vs unconstrained databases a) In the past, models have mainly been tested on constrained databases. They contain high quality images, with limited variation in pose, orientation and illumination, and exclude occlusions or background clutter; e.g. FERET face database [137], KTH action recognition database [152] and hand databases in [19, 63]. b) Contrastingly, real world images collected from the internet present large variations in pose, scale, image quality and background clutter. This makes object modelling challenging. Here are some such databases we have collected for faces, pedestrians and hands.

identify the player's face. Some articulated objects such as horses can even self occlude (figure 1.2d). Such non-rigid *deformation* of the object can also hinder vision algorithms.

Finally, objects usually appear against a cluttered background. For example, a traffic scene would contain cars, traffic lights and pedestrians as well as buildings. Thus it may not be easy to detect a face in the scene. *Background clutter* makes any of the tasks in section 1.1 much harder, particularly detection of objects. (figure 1.2e).

An ideal object model, would support performance in all of the tasks in section 1.1 despite the above challenges.

1.3 Constrained vs Unconstrained Databases

Visual algorithms are usually evaluated on standard databases. The database provides a sample space on which a particular object model is trained and tested. The choice of the database directly effects the performance and generalization ability of a visual model. For example, the model may struggle to predict the gender of an Asian person if it has only been trained on images of Caucasian men and women. Thus, it is crucial that the model can cope with a large variety of examples.

Until recently, databases typically presented simplified images to help develop fundamental models for object detection or identification. These ‘constrained’ databases, usually contain high quality images, with limited variation in pose, orientation and illumination, and exclude occlusions or background clutter. Examples of such databases are shown in figure 1.3a, including the FERET [137] for faces, the KTH [152] for action recognition and the hand databases used in [19, 63].

Interest has now shifted towards solving more advanced problems which require dealing with “realistic” looking images. Such images are taken in unconstrained environments and represent large variations in background clutter, pose, scale and image quality. This variation makes object modelling highly challenging (see figure 1.3b).

Most current methods get impressive results on constrained databases in areas of recognition [174, 50], classification [124, 176] and regression [102]. Unfortunately, the performance of these algorithms drops sharply [120, 154] when tested on real world images.

Ultimately, an ideal object model should have reliable performance on unconstrained databases, to facilitate real world applications. Unfortunately, efficient modelling of objects using real world images still remains a challenge and there has been limited research on this subject ([98, 159]). Many tasks such as gender classification, face pose estimation, and object generation remain largely unexplored.

1.4 Problem Statement

The problem we address is to find a general object representation model such that (i) it is suitable for multiple object classes, i.e. it is not specific to a single class, (ii) it is robust to the challenges described in section 1.2 and (iii) is flexible enough to tackle a wide variety of object related tasks as described in section 1.1.

1.5 Research Objective

In this thesis we investigate *patch-based* models as a general representation for visual object classes. We restrict our investigation to object classes that exhibit a reasonable amount of regularity, because (i) it is easier to model such objects and (ii) we can collect large databases of training examples. These include faces, pedestrians, cells, and human brains.

In particular we aim to explore the following aspects of patch-based representation models:

- usability across a variety of object related tasks.
- the ability to represent multiple object classes.
- the ability to handle large databases of “real world” images.

- robustness to the challenges in section 1.2

1.6 Contributions

We demonstrate that patch-based representation can be used in a wide variety of object related tasks such as: classification, regression, semantic segmentation and identification. More specifically, the contributions of this thesis are:

- A patch-based method for within-category classification applied to multiple object classes such as: faces, pedestrians and cells. We demonstrate that patch-based representation is robust to challenges such as: background clutter, scale, occlusion and variation in pose and illumination.
- A patch-based method for regression which we apply to face pose estimation. We demonstrate that patch-based representation can be used to estimate pose in “real world” environments.
- A patch-based method for face recognition across large image variations such as changes in illumination and facial expression.
- A patch-based method for semantic segmentation of objects, which we apply to faces and human brains. We demonstrate the capabilities of patch-based representation in a challenging vision task.
- Collection and annotation of a very large database ($\sim 70,000$ images) of faces in uncontrolled environments. This dataset provides realistic scenarios for testing current and future methods for a variety of classification tasks.
- Collection of a large database ($\sim 15,000$ images) of pedestrians in uncontrolled environments and annotation of pose for each pedestrian. This database can be used to test the ability of methods to handle large number of real world examples.
- Collection of a large face pose database ($\sim 12,000$ images) and providing human estimates for yaw as ground truth pose. This is to our knowledge the first large annotated database of poses in uncontrolled environment and has received wide interest from the vision community.

1.7 Report Structure

Chapter 2 contains background on the main components of object modelling: representation, learning and inference, and a review of current techniques used for each component. Chapter 3 describes a new method for patch-based within category classification, with experiments and

results on faces, pedestrians and human brain. In chapter 4 we introduce a patch-based method for regression applied to estimating face pose. In chapter 5 we describe a new method for semantic segmentation of objects, with experiments and results on hands, faces and human brains. Chapter 6 describes a patch-based generative method for identity recognition, with experiments and results on face recognition across varying illumination and expression. In the final chapter, we present our conclusions and discuss potential extensions to this work.

Chapter 2

Background

In this chapter we present the main components of visual object modelling: *representation, learning and inference*. A visual object model describes the statistical relationship between the observed data and the world state. The representation is the form that the visual data takes within this model. For example, it might simply consist of the concatenated RGB values from a region of the image. The model will inevitably depend on some parameters. Learning is the process of fitting these parameters. Inference is the task of predicting the world state given a new observation of the specific object. Section 2.1 covers several models for representation and a discussion of their merits and limitations. Different algorithms for inference and learning are described in section 2.2.

2.1 Representation

The first and arguably the most important step in object class modelling is choosing a *representation*. The representation is the format in which the object measurements are extracted and stored. The goal is to describe the object in a compact form that retains sufficient information to perform visual tasks. An ideal representation should (i) capture the object's unique structure to allow the model to distinguish it from other instances of the object, (ii) retain common characteristics between different instances and (iii) be robust to the challenges described in section 1.2.

Object representations can be divided into two groups:

- Global representations: models that use statistics about the whole object.
- Local representations: models that use local statistics about different object parts.

In the following sections, we consider each of these representations in turn.

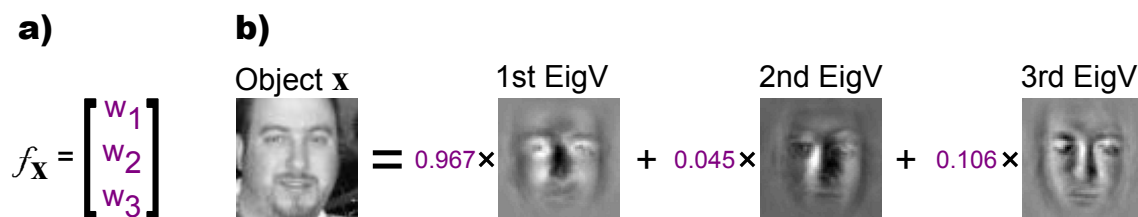


Figure 2.1: Global representation - Eigen appearance model (a) In eigen-appearance models the object x is represented as a vector of weights. Here f_x denotes this representation. These weights are obtained by projecting the observed data onto a lower dimensional space using dimensionality reduction techniques such as PCA or LDA, etc. (b) An object is described as a weighted sum of basis functions of the observed data. For example a face is described by the weighted sum of the first three eigen-vectors of the training data (note we have assumed a zero mean data in this example).

2.1.1 Global Representations

Global representations store statistical information about the whole object in a simple vector, resulting in a compact representation. Prototypical examples include (i) *subspace models* and (ii) *active appearance models*, which we will now consider in turn.

Subspace models

A dominant subspace model represents the object as a weighted sum of basis functions. Consequently, the object is represented as a vector of weights. These weights are obtained by projecting the data onto a lower dimensional space called “feature space” using dimensionality reduction methods such as principal component analysis (PCA) [161] or linear discriminant analysis (LDA) [14] (figure 2.1a). When PCA is used for projection of the data onto the feature space, the basis functions become the eigenvectors of the observed data. Thus, the model is also known as the *eigen-appearance model* (figure 2.1b). If the data is sufficiently spatially correlated, only a small number of these basis functions are needed to represent the object, giving a compact representation. Variations of this model have been applied to face recognition [161, 14, 69], and object recognition across pose and illumination [131, 105, 20].

The eigen-appearance model is conceptually simple and easy to implement but has several limitations. First, it uses global statistics, which makes it very susceptible to occlusion and large non-linear transformations such as changes in illumination. Second, it implicitly describes the joint statistics of all the pixel values with a Gaussian model. In practice this means that although large-scale variation is well described, local texture variation is not. Consequently, images generated from these models are blurry and cannot accurately model the high frequency parts of the image.

To overcome blurriness, Frey and Jojic [64] proposed a method called *transformed com-*

ponent analysis, which incorporates a discrete hidden variable in an EM framework to account for possible transformations of the observed data. This model results in sharper images, but unfortunately increases the computational burden. Moreover, this model maintains the Gaussian model of the data manifold, which means the model is unimodal. Nevertheless, the object data manifold is not in general unimodal. For example, faces with and without glasses form two separate clusters.

The eigen-appearance scheme has been modified to model the joint pixel probability distribution with non-Gaussian models [85] and mixture models [33]. Unfortunately, these extensions have their limitations. Non-Gaussian models still suffer from unimodality. Mixture models require knowing the approximate position, scale and orientation of the object. Other complexities include obtaining a large training set and reliable estimation of the parameters.

Finally, eigen-appearance models require the images to be registered to a common template. For example, they need accurate locations of key facial features such as eyes, nose, and mouth to normalize the detected face. These models struggle when the appearances of the features change significantly, for example, closed eyes, eyes with glasses, open mouth, etc. Consequently, the recognition is very much dependent on the registration process, and is not robust to shape changes. We will now consider a more powerful object representation based on *Active Appearance Models* that are much more robust in terms of handling variations in image intensity and feature shape.

Active Appearance Models

Active Appearance Models (AAMs) [32], first proposed by Cootes et al. [32], and the closely related concepts of Active Shape Models [34], Active Blobs [153] and Morphable Models [164, 90, 21], are non-linear, generative, and parametric models of a visual object.

There are two components to an Active Appearance Model: shape and appearance. The shape of an AAM is a vector of the x and y -coordinates of a set of vertices that make up a mesh. AAMs allow linear shape variation; i.e. the shape can be expressed as a base shape plus a linear combination of n shape vectors (figure 2.2b). AAMs are normally computed by applying PCA to the training meshes [32]. The base shape is the mean shape and the rest of the shape vectors are the n eigenvectors of the training meshes corresponding to the n largest eigenvalues. The appearance of an AAM is an image defined over the pixels in the base mesh. AAMs allow linear appearance variation; i.e. the appearance can be expressed as a base appearance plus a linear combination of m appearance images (figure 2.2a). The base appearance is set to be the mean image and the rest of the appearance images are the m eigenimages of the training image covariance corresponding to the m largest eigenvalues.

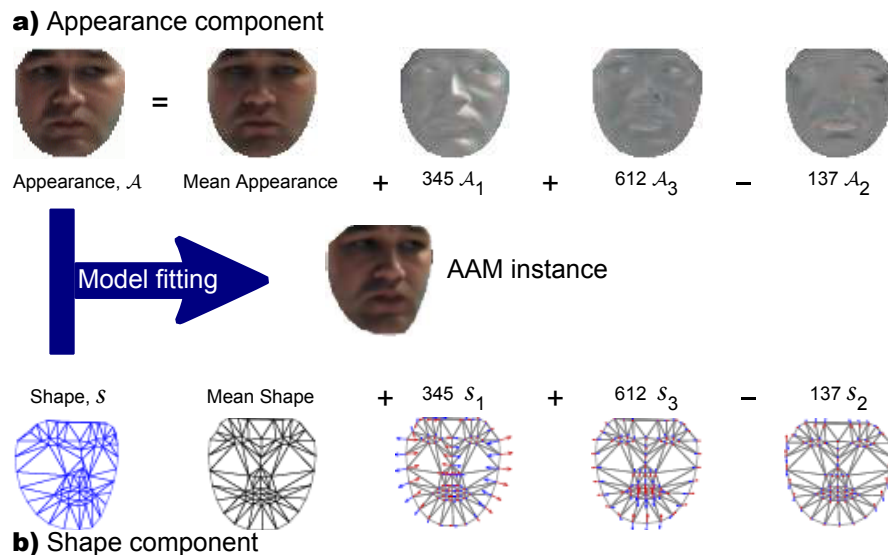


Figure 2.2: Global representation- Active Appearance Models: There are two components to an Active Appearance Model: shape and appearance. (a) The appearance of an AAM is an image defined over the pixels in the base mesh. This is expressed as a base appearance plus a linear combination of m appearance images. (b) The shape of an AAM is a vector of the x and y -coordinates of a set of vertices that make up a mesh. This is expressed as a base shape plus a linear combination of n shape vectors, which are normally the eigenvectors and the eigen values of the training data respectively. (adapted from [123])

In the above representation, shape and appearance were modelled independently [123]. Alternatively, one can parameterize shape and appearance with a single set of linear parameters [32]. Under these circumstances, the model usually needs less parameters to represent the same visual object to the same degree of accuracy.

Although linear in both shape and appearance, AAMs are nonlinear parametric models in terms of the pixel intensities. An AAM is fitted to an image by minimizing the error between the input image and the closest model instance; i.e. solving a nonlinear optimization problem.

The active appearance model (AAM) is an elegant model which has been widely used to fit statistical models of shape and appearance to images. AAMs have been successfully used in segmentation [9], tracking [70], classification [150] and face recognition [43]. Unfortunately, there are some limitations to this model.

One of the limitations of AAMs is that they are unimodal. They may fail in cases when there are large non-linear variations. This problem is partly solved by combining the model with kernel based PCA [147] and mixture of Gaussians [33]. Moreover, these models are based on eigendecomposition, hence, learning is expensive and difficult to generalize to the case where both the number of training examples and the data dimension is high.

Another disadvantage of AAMs is that they require a training set of annotated images where corresponding points (vertex locations) have been marked on each example. This is very time consuming when the training dataset is large. Unfortunately, performance of the standard AAM tends to be poor when these landmarks undergo large position changes or some features (e.g. an eye in a face image) are missing typically due to occlusion or large viewpoint changes.

Extensions of this work to handle occlusion and missing features, include view-based active appearance models [35] and layered models [89]. In [35], multiple distinct active appearance models are built, where occluded landmarks do not appear in some models. The appropriate model is selected based on pose. This approach is practical but unattractive since it involves building several models of the same object. Jones and Soatto [89] proposed a layered active appearance model where each layer is associated with one feature or part of an object. For example, they learn a seven-layered model of a car including a separate layer for the headlights, windscreen, etc. This model allows for missing features, occlusion and spatial rearrangements. However, it requires manually defining and labeling of parts or layers which is tedious and time consuming.

Discussion

To summarize, global representations work very well for registered objects with limited variation, but start to fail with real-world data, captured in uncontrolled environments. For example, the PCA based Eigenfaces model [161] achieves around 96% correct recognition when applied to registered faces with no variation in pose or scale. However, the performance drops to around 60% when applied to the LFW database [84] which contains images with large variation in pose, illumination and scale.

Moreover, models using global representation cannot deal with occlusions and geometric transformations. For example, it may be very difficult to model a highly articulated object such as a horse using a global representation model. Finally, using global statistics could be computationally expensive and slow (since they may involve calculating a full covariance matrix). Thus, may not be suitable for real time applications.

2.1.2 Local Representations

Local representations store local image statistics, and represent an object as a vector of local features. To form a feature vector, first, a set of ‘keypoints’ are detected. Keypoints are some interesting points on the image that are unique in both position and scale such as edges, corners, etc. The keypoints are commonly extracted from the image using some interest point detectors [27, 114, 60]. Once these keypoints are detected, a local image descriptor is used to character-

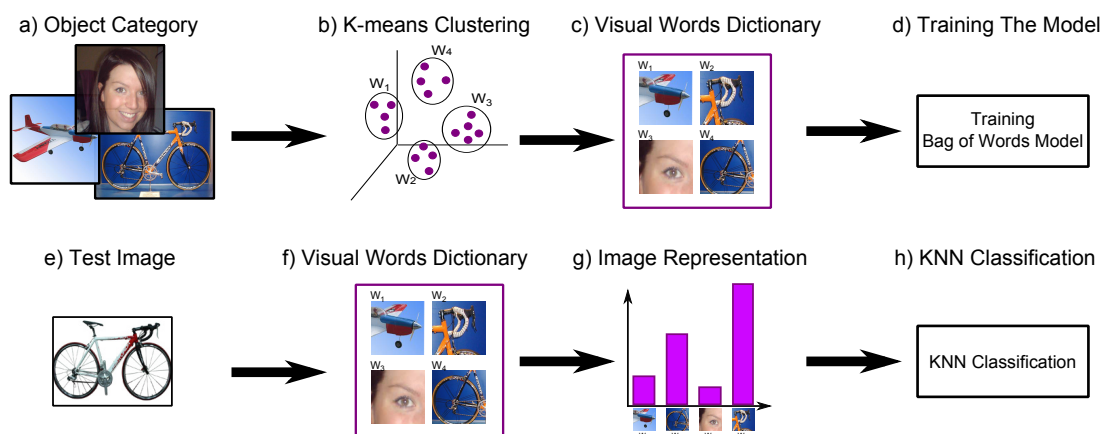


Figure 2.3: Local representation- bag of words model: (a) Bag of words model is used for classifying an object as belonging to one of many object categories. (b-c) First a large number of local feature vectors are extracted and clustered into visual words to build a dictionary. (d) The Bag of words model is trained by repeating this process for every training image. (e-g) A test image is represented by a histogram over this dictionary of visual words. (h) Finally a multi-class classifier is used to determine which category or categories to assign the image to based on the histograms.

ize the region around each point, forming the representation vector. Popular local descriptors include SIFT descriptors [115] and steerable filters [61].

We will now describe two prototypical examples that use a local representation: (i) *bag of words* models which ignore the spatial configuration of features and (ii) *part-based models* which accommodate spatial information.

Bag-of-words models

The *bag-of-words* or *bag-of-keypoints* is a representation model motivated from text categorization which was first used for object recognition by Csurka et al. [38]. A bag of words corresponds to a histogram of the number of occurrences of a particular pattern in an image. First, a set of features such as Harris affine [38, 171, 158] are detected and described using SIFT descriptors [115]. These features are then clustered into visual words using k-means to build a dictionary of selected size (figure 2.3 b-c). Given a new image, each descriptor extracted from this image is labelled with the visual word (cluster centre) to which it lies closest in feature space. Subsequently, the image is represented by histograms over this dictionary of visual words (figure 2.3 g).

The bag-of-words representation is an interesting model, which has been successfully used for visual object categorization. The model has achieved good results on simultaneously classifying several semantic visual categories [38]. The performance has been shown to be largely unaffected by position and orientation of object in image. However, there are some limitations

to the bag-of-words model which we will now discuss.

Unfortunately, clustering methods such as k-means could create generic visual words that may not be representative of a specific object class. Winn et al. [171], extend this work by automatically learning the optimal visual words and dictionary, in a supervised manner. This is done by pair-wise merging of visual words from an initially large dictionary. The final visual words are described by Gaussian Mixture Models (GMMs), where the histogram is treated as a continuous multivariate quantity. This is a more systematic way of choosing the visual vocabulary. However, this model is still based on appearance alone and ignores the spatial relationship between features. Therefore, it is unlikely that high discriminability can be achieved if the object categories are increased [157].

Sivic et al. [158], extend the bag-of-words representation to allow the learning of a model for several object categories in an unsupervised manner. This was motivated from the statistical models from text literature: probabilistic latent semantic analysis (pLSA) and latent Dirichlet allocation (LDA). The new representation extends the visual vocabulary to include words that encode spatially local co-occurring regions. Unfortunately, this model confuses objects in different categories with a similar background. Consequently, the performance decreases when more object categories are added.

The bag-of-words representation is an interesting model, because it is simple, and intrinsically invariant, due to its affine invariant features. However, this way of representing objects is not suitable for other vision tasks such as semantic image segmentation.

Another disadvantage of this model is that it relies on detection of informative features, which highly depends on the proportion of the object to background ratio, and background clutter. For instance, if the object is too small in an image, or the background is highly textured, the feature detector might pick points in the background. This model is not usually engineered to eliminate such features.

Finally a major drawback of the bag-of-words model is that, it offers a rather impoverished representation of the object, because it ignores any spatial relationships between the features.

Part-based models

Part-based models represent the object as a collection of features or parts. Each part has a distinctive appearance and spatial position. Well known examples are constellation models [25, 170, 57], pictorial structures [96, 54, 97] and fragment-based models [162, 156] which we present in turn.

Constellation Models

The *constellation* model introduced by Burl et al. [25] consists of a number of parts, each

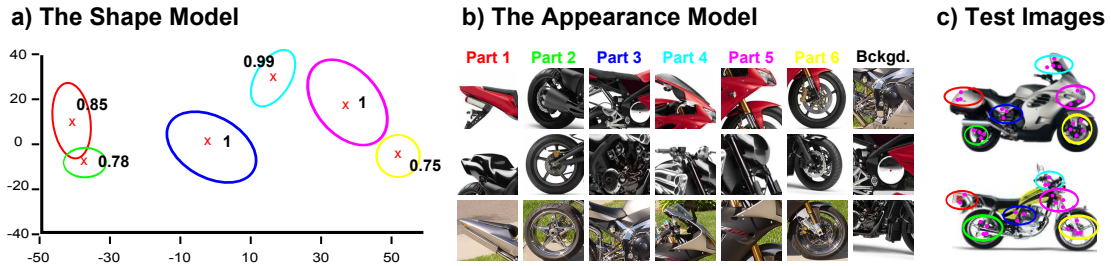


Figure 2.4: Local representation- constellation model: Constellation model of a motorcycle model with 6 parts (adapted from [57]). (a) The shape model. The ellipses represent the variance of each part. (b) Three patches closest to the mean of the appearance density for each part. (c) Some sample test images. The pink dots are features found on each image and the colored circles indicate the features of the best hypothesis in the image. The size of the ellipses indicates the score of the hypothesis (the bigger the better).

encoding information on both the shape and appearance. The original work of Burl et al. [25] takes a supervised approach where parts are learned from a labelled training set. Weber et al. [170] extend this work to allow unsupervised part detection and hypothesis creation in a maximum-likelihood manner.

While both of these approaches model explicitly shape variability, they do not model the variability of appearance. Furthermore, their experiments are limited in terms of scale-invariance. Fergus et al. [57] extended the model to take these aspects into account. In their model, each part has an appearance, relative scale and can be occluded or not. Shape is represented by the mutual position of the parts (figure 2.4). The entire model is generative and probabilistic, so appearance, scale, shape and occlusion are all modelled by probability density functions. To model an object category first, parts and their scales are detected. Then the correspondence between the features and the parts are learned in an unsupervised manner. Finally, the parameters of the above densities are estimated from these parts by maximizing the likelihood of the training data.

Unfortunately constellation models have several limitations. First, they use a sparse representation of the object and therefore cannot be applied to some vision tasks such as generation. Second, they require objects with distinguishable parts in a relatively fixed spatial configuration. Hence, they are not robust to large viewpoint and scale changes. Finally, they are not well suited for representing articulated objects, as a joint Gaussian distribution cannot capture multiple articulation points.

Pictorial Structures

Some part-based models use *pictorial structures* pioneered by Fischler and Elschlager [59]

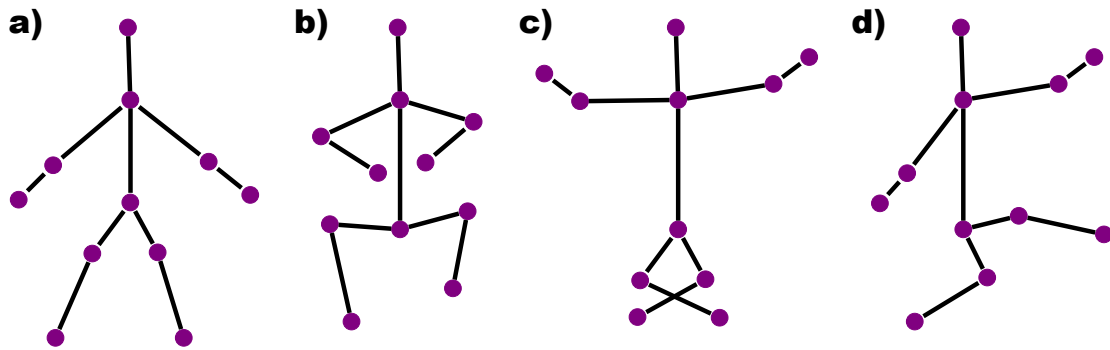


Figure 2.5: Local representation- pictorial structures: A pictorial structure model for an object is expressed in terms of an undirected graph, where the vertices (purple dots) correspond to the n parts, and there is an edge for each pair of connected parts. (a) An example of a pictorial structure for the human body which is represented as a set of joints connecting different body parts. (b-d) Several configurations of the pictorial structure to represent different poses.

to represent an object. A pictorial structure model for an object is given by a collection of parts with connections between certain pairs of parts. This model is expressed in terms of an undirected graph $G = (V, E)$, where the vertices $V = \{v_1, \dots, v_n\}$ correspond to the n parts, and there is an edge $E(v_i, v_j)$ for each pair of connected parts v_i and v_j . An instance of the object is given by a configuration $L = (l_1, \dots, l_n)$, where each l_i specifies the location of part v_i . This representation is particularly interesting for modelling articulated objects such as the human body (figure 2.5).

In the original work by Fischler and Elschlager [59], the problem of matching a pictorial structure to an image is defined in terms of an energy function to be minimized. The cost or energy of a particular configuration depends both on how well each part matches the image data at its location, and how well the relative locations of the parts agree with the deformable model. In this work they only consider the problem of finding the best match of a pictorial structure model to an image. Unfortunately, this energy function has many free parameters. For each different object, one has to construct a model, which includes picking appearance parameters for each part, a set of edges connecting pairs of parts and the characteristics of the connections.

Felzenszwalb and Huttenlocher [54] proposed an algorithm for automatically learning these parameters from examples. They formulate a statistical framework such that energy minimization is replaced by finding the maximum a posteriori estimate of the object configuration given an observed image. To gain computational efficiency, the connections between parts are restricted to a tree structure. The model is tested on object detection and localization of faces and human bodies. Kumar et al. [96], extend this work where parts are connected to each other to form a complete graph instead of a tree structure. This proved to be advantageous over the

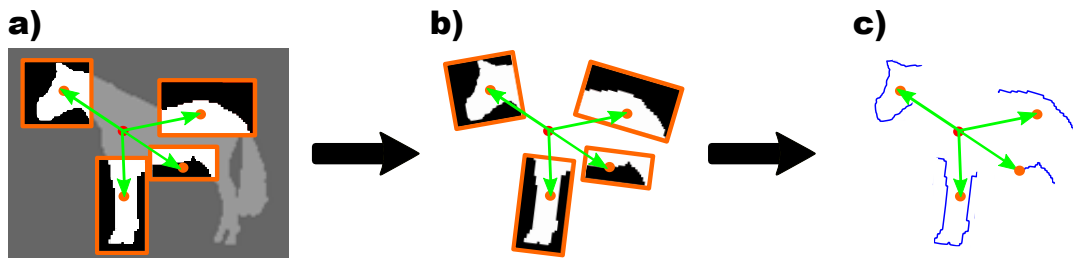


Figure 2.6: *Local representation- fragment-based model:* objects within a class are represented in terms of common image fragments that are used as building blocks for representing objects that belong to a common class. For example, (a) Random rectangular regions are paired with their positions (green arrows) relative to the object centroid (red circle). (b) These are randomly perturbed. (c) Edges are calculated to form contour fragments. (adapted from [156])

tree structure for recognition tasks. A variation of this model known as OBJCUT [97], extends the standard pictorial structures to handle partial self occlusion and uses them as a prior for object category segmentation. The parts in OBJCUT are learned from video sequences. Unfortunately, this requires either complex tracking of video sequences or manual labeling of parts.

A crucial step in pictorial structures representation is the reliable detection of the parts. The work of Felzenszwalb and Huttenlocher [54] is based on a simple appearance model requiring background subtraction, which is not always appropriate for real world scenarios. Extensions of this model [7] to work without background subtraction rely on a discriminative appearance model, using image features such as Gaussian derivatives. Further improvements include extracting more powerful templates [144] and integrating features from an automatic foreground segmentation step [58].

A major drawback of the pictorial structures is that they assume an object specific representation. The model is tailor-made to one specific class of objects at a time and, a new set of pictorial structures are designed for each new object category. For example, a pictorial model learned for a human body cannot be used for other objects such as cars, horses etc. This may be highly inconvenient when modelling large number of object classes that do not share a similar structure.

Fragment-based Models

Fragment-based models, first proposed by Ullman et al. [162], represent objects within a class in terms of common image fragments. These are used as building blocks for representing objects that belong to a common class. Note, the fragments mainly exploit the contours of an object, as opposed to the texture-based parts extracted from the interior of an object in the pic-

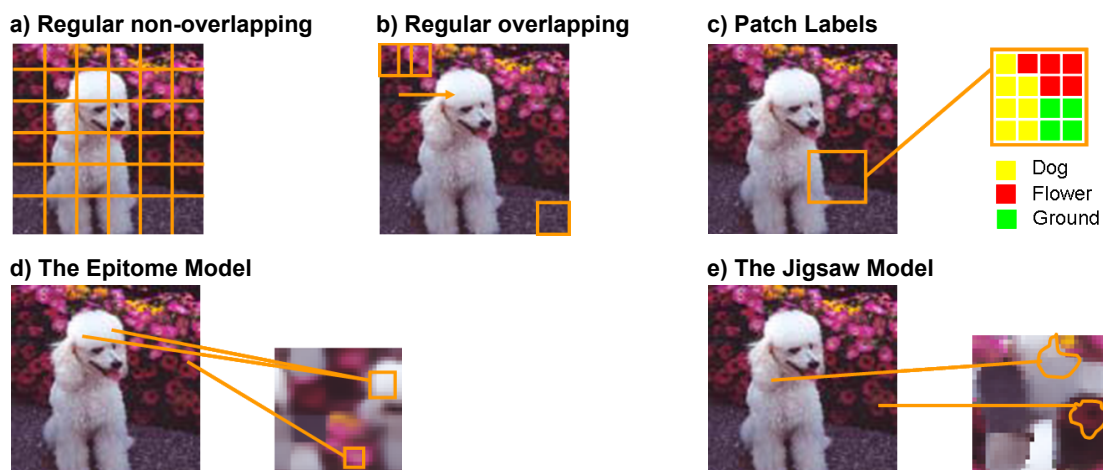


Figure 2.7: (a) Regular grid of fixed size (square) non-overlapping patches. (b) Regular grid of fixed size (square) overlapping patches (c) Each patch is associated with one of a predefined set of labels. (d) The epitome of an image is a miniature, condensed version containing textual and shape components of the image. The epitome is built from the (usually square) patches of various sizes from the input image and a mapping is defined from the epitome to the image pixels (image adapted from [88]). (e) The jigsaw of an image contains all the necessary image pieces that can be used to generate the target image set. Irregularly shaped and size patches are extracted from the jigsaw to reconstruct the target image.

torial structures (there are some exceptions [96, 97]). The fragments are selected in a supervised manner from a training set based on a criterion of maximizing the mutual information of the fragments and the class they represent. This representation was tested on detection of faces and cars. A partially-supervised version was also proposed by Shotton et al. [156] where a boosting method was used to extract image contours, and applied to detection of horses, cars, faces and motorbikes (figure 2.6).

Unfortunately, fragment-based models still use a sparse representation and cannot be applied to tasks like object generation. Moreover, the object-specific fragments make it difficult to extend the model to multiple object classes, without building separate models for each class. Finally, they usually require high level of segmentation in training, which could be very laborious.

2.1.3 Patch-based Representation

Bridging the gap between local and global representation is a *patch-based* representation, where an object is represented as a collection of patches. These may occur in various configurations such as grids of overlapping or non-overlapping patches (figure 2.7a,b). Patch-based representation has several nice properties (i) it provides a dense representation of an object, (ii) covariance

is only modelled over a certain scale (which varies with the patch size) and (ii) it retains some spatial resolution.

Patch-based models have recently been very successful in other areas of computer vision. For instance, Efros and Freeman [46] proposed a powerful model for texture synthesis that uses a simple representation based on overlapping patches. Another popular application area is semantic image segmentation [77, 169], where patches are used to encode label information (figure 2.7c). Patches have also been successfully used for image denoising [148, 41] and generation of photorealistic face images [126].

Jojic et al. [88], learned a representative palette of intensity/shape known as the ‘epitome’ from which patches are extracted (figure 2.7d). The epitome is constructed as a generative model of patches of various sizes from the input image and a mapping is defined from the epitome to the image pixels. Epitomes have been used for image segmentation [88], denoising [28] and object detection [29]. This model has been extended to extract irregularly shaped and size patches known as jigsaws [93] (figure 2.7e), and was used for part detection.

Despite the success of patch-based models in the tasks above, they have rarely been used to represent object classes. Recently, Lucey and Chen [118] proposed a patch-based representation to model faces and used it for face verification in the presence of large pose differences. They divide each of the gallery images into a non-overlapping grid of patches which are assumed to be independent. They demonstrate that using patches to represent the gallery image, improves the performance of their algorithm on face verification using FERET face database [137].

2.1.4 Summary and Discussion

Many traditional models in computer vision have used a global representation of an object. Using global statistics is ideal in that it allows a dense representation of the object which is suitable for tasks such as generation. However, it cannot deal with occlusions and geometric transformations. For example, it may be very difficult to model a highly articulated object such as a horse with a global representation model. Models that use global statistics could be computationally expensive and slow (since they may involve calculating a full covariance matrix). Hence, may not be suitable for real time applications. Moreover, global representations do not model fine local texture variations and may fail in problems such as face recognition under varying expression or illumination.

Local models on the other hand, treat different parts of the image as independent or conditionally independent and may not be globally coherent. Moreover, these models use a sparse representation of the object, throwing away a lot of potentially useful information. Consequently, they are not suitable for generating new instances of objects.

Pictorial structures are more powerful local representation models, as they can cope with articulated objects and occlusion. The major limitation however, is that they build object specific models which cannot be easily used for a different class of objects without rebuilding the model.

Alternatively, the patch-based representation is a simple yet very flexible representation which allows us to exploit the advantages of both global and local representation models. This representation has been successful in other areas of computer vision, but unfortunately, there has been a limited investigation on the use of patch-based representation in object modelling. Thus, many of the vision tasks mentioned in section 1.1 such as identification, regression and within-category classification, remain largely unexplored.

The shortcomings described above motivate our work. We present a patch-based representation model that has the following properties:

- It can capture accurate and selective information about the object class in hand.
- It has a dense representation, and retains global structure of an object making it suitable for synthesis and generation tasks.
- It does not make any object-specific assumptions, making it a generic representation which can be used across multiple object classes.
- It is (potentially) multi-modal, allowing a wide range of objects to be modelled under various imaging conditions.

Subsequently, we explore our patch-based representation model for a variety of vision tasks such as identity recognition, classification, regression and semantic object segmentation on several classes including: faces, pedestrians, hands and human brains.

2.2 Learning and Inference

An object class model mathematically relates the visual data \mathbf{x} and the world state y . The model specifies a family of possible relationships between \mathbf{x} and y and the particular relationship is determined by the model parameters θ . Once a suitable representation is decided for the visual data, the subsequent steps in developing an object class model are: *learning* and *inference*. Learning refers to the process of fitting the parameters θ using paired training examples $\{\mathbf{x}_i, y_i\}$ where both the measurements and the world state are known. Inference is the task of predicting the world state given a new observation of the specific object. Inference can involve an algorithm that takes a new observation \mathbf{x} and uses the model to return the posterior $Pr(y|\mathbf{x})$ over

the world state y . Alternately, it might return the maximum a posteriori (MAP) solution or draw samples from the posterior. Inference can take many forms such as classification or regression depending on whether the world state to be predicted is discrete or continuous.

The two main paradigms used for relating the data \mathbf{x} to the world y are:

1. **Discriminative:** which models the contingency of the world on the data $Pr(y|\mathbf{x})$
2. **Generative:** which models the joint occurrence of the world and the data $Pr(\mathbf{x}, y)$

We will now consider these types of models in turn and discuss learning and inference in each.

2.2.1 Discriminative Models

Probabilistic discriminative models are built to capture the boundaries between the different possible output states of a system without taking interest in modelling the distribution of the inputs. They focus on directly modelling the probability distribution over the world state y conditioned on the input data \mathbf{x} . First an appropriate form for the distribution $Pr(y)$ is chosen over the world state y . Then the distribution is formed as a function of the data denoted as $Pr(y|\mathbf{x})$. To determine this distribution a parametric model is chosen governed by a set of parameters θ . The conditional distribution is then defined as $Pr(y|\mathbf{x}, \theta)$, where θ are the parameters of the model.

The goal of the learning algorithm is to fit the parameters θ using paired training data $\{\mathbf{x}_i, y_i\}$. This can be done using maximum likelihood (ML), maximum a posteriori (MAP) or Bayesian approaches.

The goal of inference is to find a distribution over the possible world states y for a particular observation \mathbf{x} . In this case, this is easy since we have directly constructed an expression for the posterior distribution $Pr(y|\mathbf{x}, \theta)$. Sometimes instead of a probability distribution, a *discriminant function* $f(\mathbf{x})$ is designed which returns one of the possible world states y . A graphical representation of a basic discriminative model is shown in figure 2.8a.

Popular discriminative models include: logistic regression [18], Gaussian processes [145], support vector machines (SVM)[163] and neural networks (NN) [17]. Among traditional discriminative methods, neural networks and support vector machines have been particularly popular for modelling object classes. Some examples include use of neural networks for object detection [149], and classification [67, 37, 86, 175]. Similarly SVMs have been used in a discriminative approach for object detection [127, 39, 133, 134] and classification [38, 124, 116, 36, 98, 15, 68, 116, 150, 12].

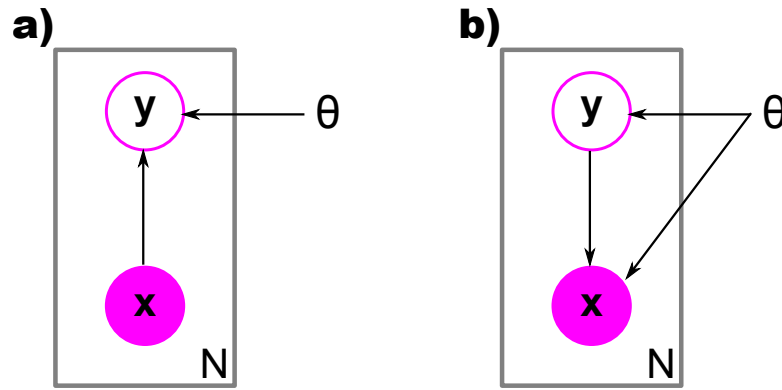


Figure 2.8: Graphical representation of (a) a basic discriminative model and (b) a basic generative model. Where we have N data points $x_1 \dots x_N$ together with corresponding labels $y_1 \dots y_N$ and θ represents other parameters of the model. The arrows indicate conditional probability relations.

2.2.2 Generative Models

Probabilistic generative models are designed to model interactions between all of the variables in the system. They maintain probability models over the data which can be used to generate new observations. Generative models describe the joint probability distribution $Pr(\mathbf{x}, y)$ of the world y and the data \mathbf{x} . The goal of learning is to use paired training examples $\{x_n, y_n\}$ to fit the parameters θ of the distribution. The goal of inference is to compute the posterior distribution $Pr(y|\mathbf{x})$, this is done using the Bayes' rule:

$$Pr(y|\mathbf{x}) = \frac{Pr(\mathbf{x}, y)}{\int Pr(\mathbf{x}, y) dy} \quad (2.1)$$

Alternatively, generative approaches can model the contingency of the data \mathbf{x} on the world y : $Pr(\mathbf{x}|y)$. Now the distribution $Pr(\mathbf{x})$ depends on both the world state and the model parameters, we write this as $Pr(\mathbf{x}|y, \theta)$ and refer to it as the *likelihood*. As before in inference, we aim to compute the posterior distribution $Pr(y|\mathbf{x})$. To this end we specify a prior $Pr(y)$ over the world state and use Bayes' rule:

$$Pr(y|\mathbf{x}) = \frac{Pr(\mathbf{x}|y)Pr(y)}{\int Pr(\mathbf{x}|y)Pr(y) dy} \quad (2.2)$$

The latter formulation of generative models is particularly interesting, as it allows injection of prior knowledge in the system. Experts also use their experience of the problem to choose suitable distributions and reasonable conditional independencies. The relevant graphical model for this definition is shown in figure 2.8b.

Popular generative models include naive Bayes, hidden Markov models (HMM) [143], Markov random fields (MRF) [108] and Gaussian mixture models (GMM) [18]. The generative

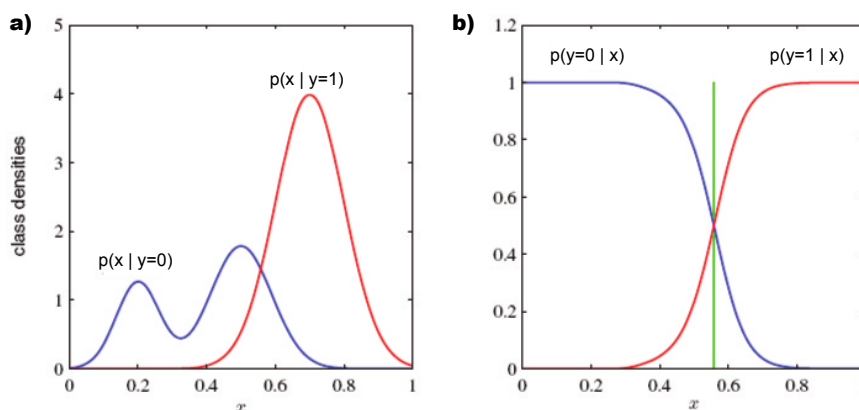


Figure 2.9: Generative vs. discriminative models for binary classification (a) Generative models, model the class-conditional densities for two classes $Pr(x|y=0)$ and $Pr(x|y=1)$ for an input variable x (b) Discriminative models, directly model the probability distribution over the world state y conditioned on the input data x . Thus, they capture the decision boundary (the green vertical line) between the different possible output states of a system without taking interest in modelling the distribution of the inputs. Note that the left-hand mode of the class-conditional density $p(x|y)$, shown in blue on the left plot, has no effect on the posterior probabilities (shown on the right). (adapted from [18])

Bayesian framework has been used for object detection [51], recognition [53, 57, 170, 52], and content based image retrieval [79]. Note some generative methods have used a patch based representation (discussed in section 2.1.3). Some areas of successful application of patch-based representation with generative inference are listed in Table 2.1.

2.2.3 Generative vs. Discriminative Approaches

So far we have described the principles of generative and discriminative models. We will now discuss the strength and weaknesses of the two approaches.

Some vision tasks such as object recognition involve learning several object categories. In such tasks generative models have an advantage over discriminative ones, as the model is learnt independently for each category. This one to one mapping, between the model and a category, makes it very easy to add categories. Conversely, because discriminative models are concerned with boundaries between the categories, all the categories need to be considered jointly. Therefore adding a new category requires the entire model to be rebuilt.

Vision algorithms often deal with incomplete or unlabelled data. In such scenarios it is very helpful to have access to the distribution from which the data was drawn i.e. $Pr(x)$. This can be obtained by marginalizing out the world state y from the joint distribution $Pr(x, y)$ in the generative approach. This highlights the modelling ability of the generative approach. On the contrary, discriminative models cannot easily handle incompleteness since the distribution

Task	Inference Method
Segmentation	Generative Latent Variable Model [93] Generative epitome model [88]
Image Denoising	Generative Probabilistic Model [88], Field of Experts [148] Directed models [42]
Image Editing / Inpainting	Markov random fields [30], Directed models [42]

Table 2.1: Areas of successful application of patch-based representation with generative inference.

of the data is not explicitly modelled.

A fundamental property of the generative approach is that it allows incorporation of expert knowledge in the form of a prior, and models the distribution of the system parameters. Discriminative models on the contrary, lack this flexibility and are often used as a ‘black box’. Moreover, modelling the likelihood of the data in the generative approach mirrors the actual way that the data was generated: the world state created the data. This is particularly desirable for tasks such as generation and synthesis. Furthermore, the generative approach allows integration of several models in a single Bayesian framework.

The discriminative approach is appealing in classification problems since it directly models the boundary between object classes. Hence, it does not waste resources on modelling the class-conditional density (like the generative approach) which could potentially be a harder problem. Moreover, if the input data \mathbf{x} is high dimensional, modelling the class density may have very little effect on the posterior probability. This is illustrated in figure 2.9. Another advantage of the discriminative approach is their speed. They are generally faster since $Pr(y|\mathbf{x})$ is directly modelled.

Complementary properties of generative and discriminative methods have subsequently lead to combining these two approaches in a hybrid model, which has been applied to: image retrieval [112], object recognition [81, 57, 82, 103].

2.2.4 Summary and Discussion

We have reviewed two main approaches to statistical inference. When dealing with a classification problem, discriminative classifiers seem more attractive. This is because they directly model the boundary between object classes and are typically very fast at making predictions for new data points. When dealing with missing or partially labelled data however, a generative approach is favorable, since it explicitly models the distribution from which the data was drawn.

Subsequently, this allows new samples to be generated from this distribution.

In conclusion, there is no general agreement as to which of the generative or discriminative approaches is superior. However, in the spirit of wanting models that can achieve all of the tasks in the first chapter we choose a generative framework.

There have only been a few generative methods that use patches (see table 2.1), and have not addressed problems such as classification, regression, identification and semantic segmentation. In this thesis, we aim to build generative object class models using a patch-based representation, which we will apply to all of these problems.

Chapter 3

Patch-based Within-Category Classification

3.1 Introduction

Recent advances in computer vision have allowed us to reliably detect objects with limited variation in structure such as faces, pedestrians and cars in real time [165, 55]. A typical approach is to use *sliding window* object detectors such as the work of Viola and Jones [165] and [111, 80, 55]. Sliding window object detectors consider small image windows at all locations and scales, and perform a binary detection at each position. The output of a sliding window object detector is a bounding box around the object of interest (see figure 3.1).

The success of these techniques allows us to collect large databases of such objects. Subsequently, it would be useful to describe their characteristics (attributes). For example, we might classify gender in face images or phenotype in cell images. This “within-category” classification task has quite different characteristics to other forms of object recognition. All of the examples have a great deal in common and we aim to classify quite subtle differences (see figure 3.2).

In this chapter we present a novel patch-based image representation and we investigate its application to within-category classification.

3.2 Motivation

Within-category classification has widespread applications including targeted advertising, consumer analysis, and medical image analysis. Examples include biological classification, where we might automatically screen cell cultures for diseases, and gender classification which could be used as a preprocessing step in face recognition.

A large body of research has investigated different learning algorithms for particular within-category classification tasks including neural networks [67, 24], support vector machines [150, 124, 72] and AdaBoost [154, 11]. Most methods use tailor-made representations specific to the object of interest. For example, Brunelli and Poggio [24], extract geometric features from faces

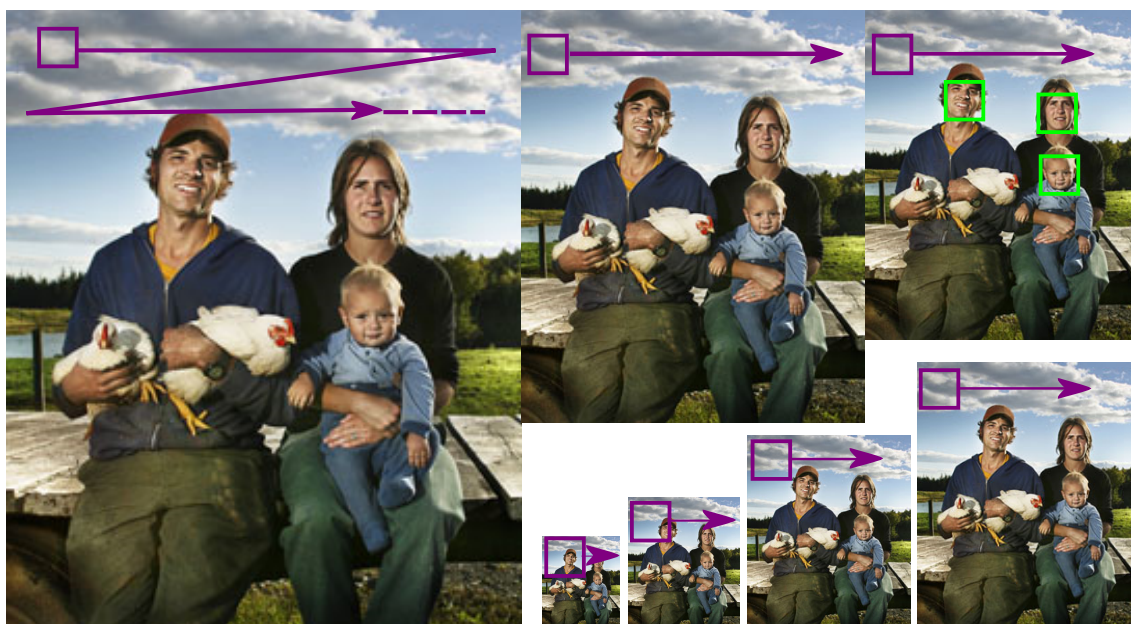


Figure 3.1: *Sliding window object detection:* Sliding window object detectors consider a small and fixed size image window at all locations and scales of the image and perform a binary detection for each. The output of a sliding window object detector is a bounding box around the object of interest.

such as pupil to eyebrow ratio, eyebrow thickness, and nose width, as input to a neural network to perform gender detection. Saatci and Town [150] use Active Shape Models to represent faces for gender classification. Similarly, 2D contours and stick figures have been used to represent human bodies for motion analysis [106] and action recognition [45]. Domain specific features are also used in cell screening. These include features such as the size, perimeter and convexity of cells [91] as well as the size and shape of the nuclei [26].

These techniques have several disadvantages. First, object specific representations cannot be applied to other problems without major alteration: most techniques have only been applied to a single class. Second, most methods do not exploit the large amounts of available training data (there are some exceptions, e.g. [98]). This is partly due to the computational complexity of these methods. For example, to find an optimal solution with non-linear SVMs generally involves solving a quadratic problem which has complexity on the order of $O(n^3)$ where n is the size of the training set (this drops to $O(n^2)$ for approximate solvers). Instead, these methods have mostly been investigated using small databases some of which contain images that are not typical of the real environment. For example, in gender classification, the FERET database is often used, although it does not contain the variations in pose, illumination, occlusion and background clutter seen in figure 3.2. As a result, the performance of most methods drops sharply when tested on images captured in uncontrolled environments [120].

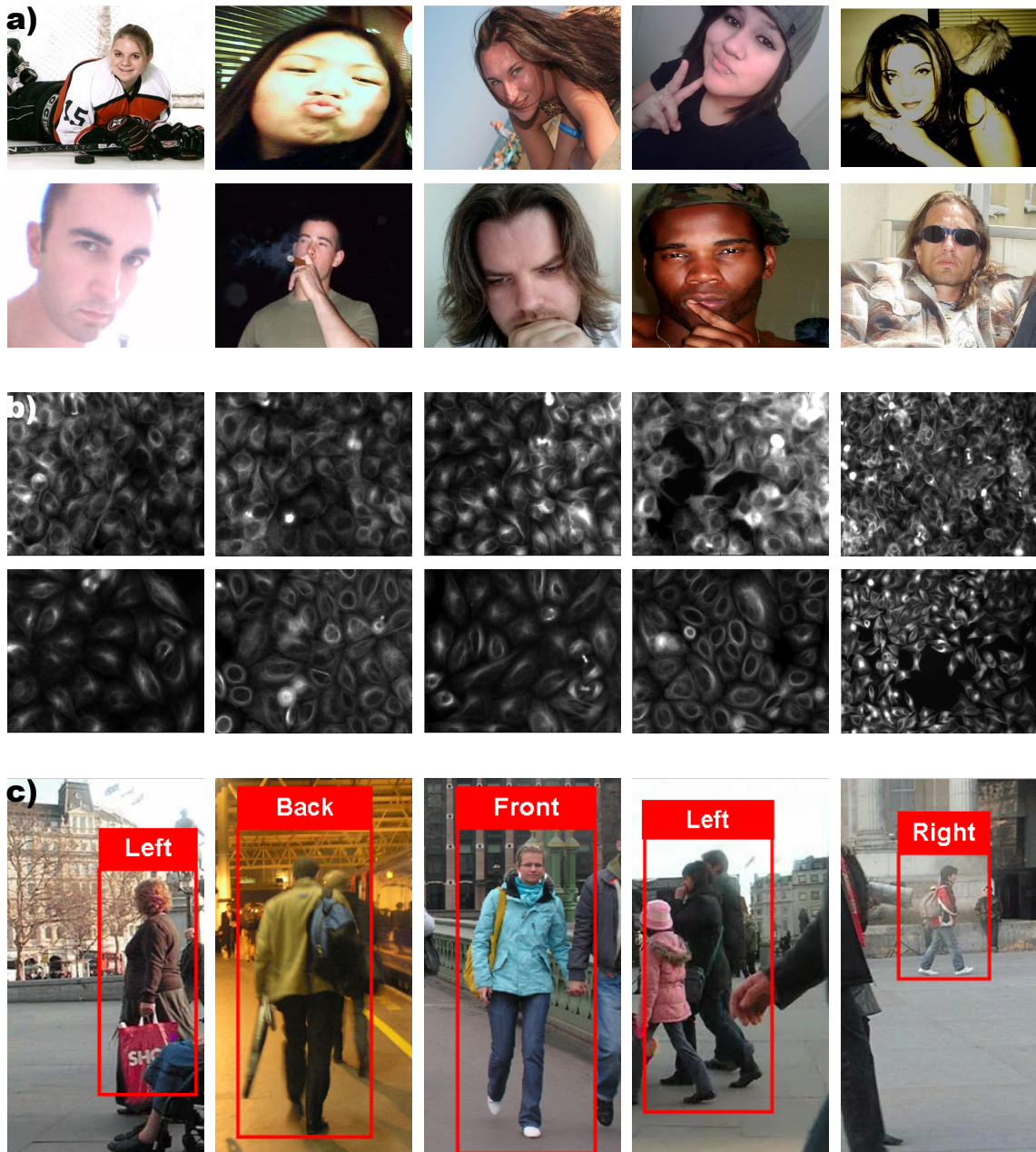


Figure 3.2: Within-Category classification: We address the problem of within-category classification on images captured in uncontrolled environments, where large within-category variations are present. For example, we classify (a) gender in face images, (b) phenotype in cell images and (c) pose in pedestrian images. These examples were all correctly classified.

In this chapter we propose a Bayesian framework for within-category classification that exploits very large databases of objects and can be used for disparate object classes. We build a non-parametric generative model that describes the test image with patches from a library of images of the same object. All the domain specific information is held in this library: we use one set of images of the object to help classify others. We test our algorithm on large real-world databases of faces, pedestrians and human cells.

3.3 Method

Our approach breaks the test image into a non-overlapping regular grid of patches. Each is treated separately and provides independent information about the class label. At the core of our algorithm is a predefined library of object instances. The library can be considered as a palette from which image patches can be taken. We exploit the relationship between the patches in the test image and the patches in the library to determine the class.

The idea behind our model is conceptually simple. Consider the toy problem of assigning a test patch as belonging to either class A or class B, where we have a small library containing only two patches. We also define a mapping between a test patch and a library patch based on a similarity measure. Let's assume we have a simple scenario where all of the training patches in class A map (i.e. closely match) to the first library patch, and all of the training patches in class B map to the second library patch. Classifying a new data point is now very easy. By just looking at the frequency of the library patches used for each class we can say that if the test patch maps to the first library patch then it belongs to class A, otherwise it belongs to class B.

In reality however, the situation is not so simple. The library is usually much larger, without a clear mapping pattern, and we replace simple frequencies with probability distributions over the library patches (see sections 3.3.1-3.3.2). Nevertheless, the core idea remains the same. Our algorithm can be understood in terms of either *generation* or *inference* and we will describe each in turn.

First, let us look at *generation* from this model. For example, consider the generative process for the top-left patch of a test image. The true class label induces a probability distribution over all the patches in the library based on the learned parameters for that class. We choose a particular patch using this probability distribution and add independent Gaussian noise at each pixel to create the observed data. In inference we invert this generative process using Bayes' rule to establish which class label was most likely to be responsible for the observed data.

Alternatively, in *inference* (see figure 3.3), the test image patch is approximated by a patch from the library \mathcal{L} . The particular library patch chosen can be thought of as having a different

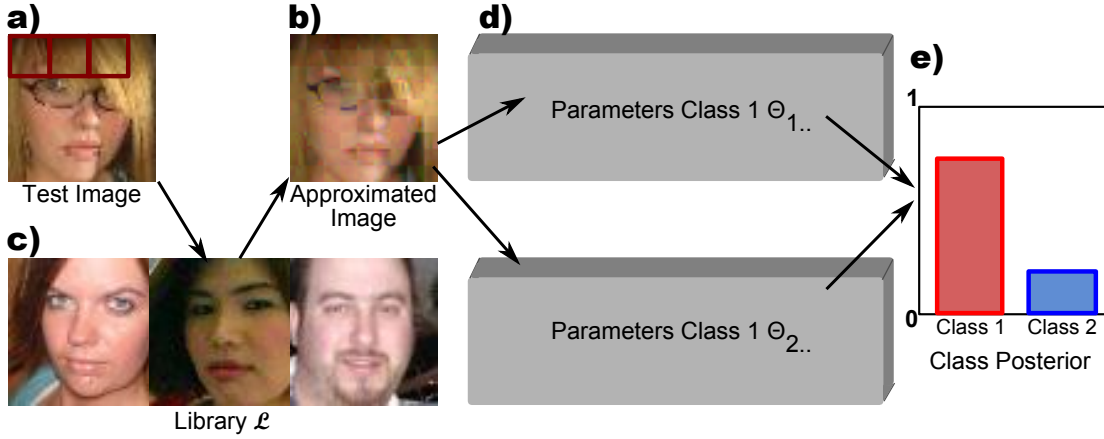


Figure 3.3: Inference: (a) A test image \mathbf{Y} is decomposed into a regular patch grid. (b) A large library \mathcal{L} is used to approximate each test image patch. (c) The choice of library patch provides information about the class. (d) Parameters θ associated with each class are used to interpret these patch choices and (e) used in a Bayesian framework to calculate a posterior over classes.

affinity with each class label. These affinities are learned during a training period and are embodied in a set of parameters θ associated with each class. The relative affinity of the chosen library patch for each class is used to determine a posterior probability over classes. In other words, we employ a voting scheme where each patch independently casts a vote for each class. We will now consider the inference process in more detail.

3.3.1 Inference

Consider the task of assigning a class label \mathcal{C} to a test image, where there are K possible classes i.e. $\mathcal{C} \in \{1 \dots K\}$. The test image \mathbf{Y} is represented as a non-overlapping grid of P patches $\mathbf{Y} = [\mathbf{y}_1 \dots \mathbf{y}_P]$. The model is trained from I training examples \mathbf{X}_c from each of the K classes. Each training image is also represented as a non-overlapping grid of patches of the same size as the test data. We denote the p^{th} patch from the i^{th} training image of the c^{th} class by \mathbf{x}_{icp} (figure 3.5a).

We also have a library \mathcal{L} of images that are not in the training or test set and would normally contain examples of all classes. We will consider the library as a collection of patches \mathcal{L}_l where $l \in \{1 \dots N\}$ indexes the N possible sites from which we can take library patches (see figure 3.5b). These patches are the same size as those in the test and training images but may be taken from anywhere in the library (i.e. they are not constrained to come from a non-overlapping grid). In other words, the sites l denote every possible pixel position in the library images.

Each library patch has a different affinity with each class label which is embodied in the model parameters θ . For example, the parameter θ_{cpl} represents the probability of the patch from library site l to be picked when considering patch p of an example image of class c . The

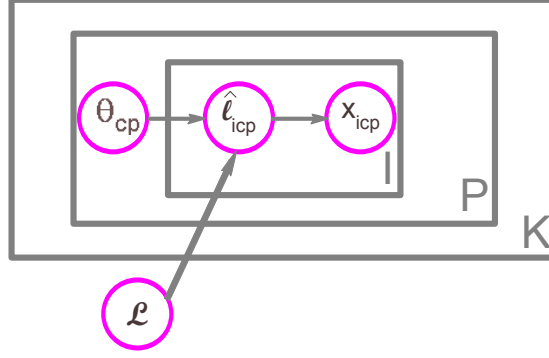


Figure 3.4: Graphical representation: Generation process of a patch p from the test image: the true class label c induces a probability distribution over all the patches in the library \mathcal{L} based on the learned parameters for that class θ_{cp} . We choose a particular library patch \hat{l}_{icp} using this probability distribution and add independent Gaussian noise at each pixel to create the observed data \mathbf{x}_{icp} . The terms I , P and K denote the number of training images, the total number of patches and the number of classes, respectively.

final solution however, does not depend on these parameters, and we will later show how to marginalize them out.

The output of our algorithm is a posterior probability over class label \mathcal{C} . We calculate this using Bayes' rule

$$Pr(\mathcal{C} = c | \mathbf{Y}, \mathbf{X}) = \frac{\prod_{p=1}^P Pr(\mathbf{y}_p | \mathcal{C} = c, \mathbf{x}_{\bullet cp}) Pr(\mathcal{C} = c)}{Pr(\mathbf{Y})} \quad (3.1)$$

where we have assumed that the test patches \mathbf{y}_p are independent. The notation \bullet indicates all of the values that an index can take, so $\mathbf{X} = \{X_1 \dots X_K\}$ denotes the training images from all of the K classes and $\mathbf{x}_{\bullet cp}$ denotes the p^{th} patch from all I training images from the c^{th} class.

Although the likelihood in Equation 3.1 depends on the library, it is not conditioned on the parameters of the model θ . We take a Bayesian approach and marginalize over the model parameters, so the likelihood terms have the form:

$$Pr(\mathbf{y}_p | \mathcal{C} = c, \mathbf{x}_{\bullet cp}) = \int Pr(\mathbf{y}_p | \theta_{cp\bullet}) Pr(\theta_{cp\bullet} | \mathbf{x}_{\bullet cp}) d\theta_{cp\bullet} \quad (3.2)$$

where $\theta_{cp\bullet}$ are all of the parameters associated with the p^{th} patch for the c^{th} class. To calculate the likelihood, we first find the index l^* of the library site that most closely matches the vectorized pixel data from the test patch \mathbf{y}_p . We are assuming that the test patch is a Gaussian corruption of the library patch and we can find the most likely site to have been responsible using maximum a posteriori estimation

$$l^* = \arg \max_l \mathcal{G}_{\mathbf{y}_p}[\mathcal{L}_l; \sigma^2 \mathbf{I}] \quad (3.3)$$

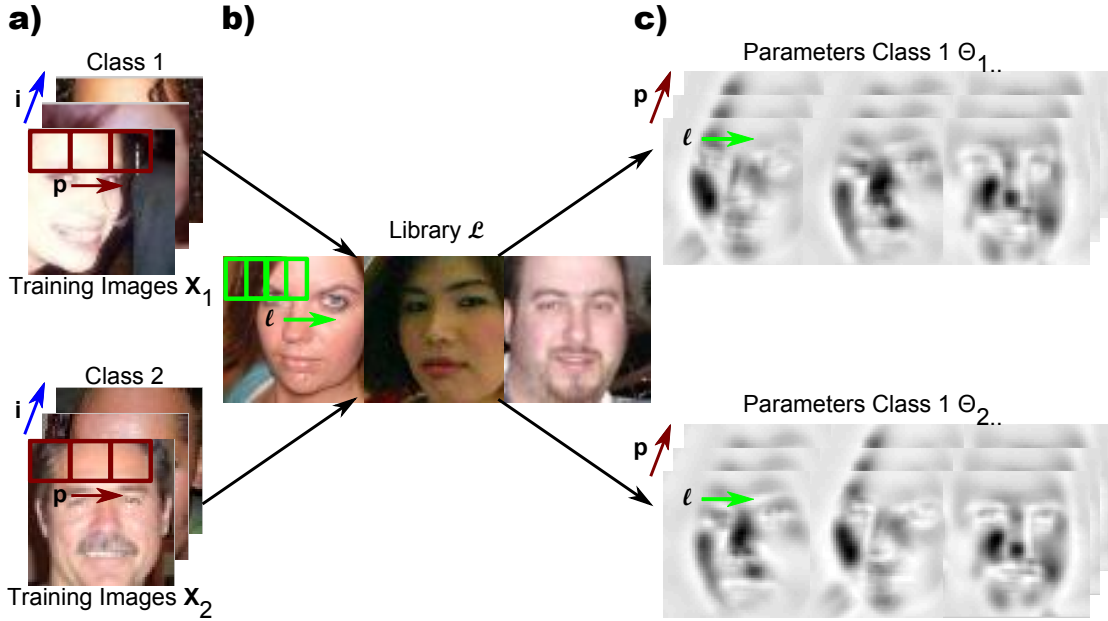


Figure 3.5: Training: (a) The model is trained from I training examples from each of the K classes. Each training image is represented as a non-overlapping grid of patches denoted by p . (b) We also have a library \mathcal{L} of images that are not in the training or test set and contain examples of all classes. The library is considered as a collection of patches \mathcal{L}_l where $l \in \{1..N\}$ indexes the N possible sites. (c) The parameter θ_{cpl} represents the probability of the patch from library site l to be picked when considering patch p of an example image of class c .

where \mathcal{L}_l is the vectorized pixel data from site l of the library \mathcal{L} . We define the likelihood to be

$$Pr(\mathbf{y}_p | \theta_{cp\bullet}) = Pr(l^* | \theta_{cp\bullet}) = \theta_{cpl^*} \quad (3.4)$$

From this it can be seen that the parameter θ_{cpl} represents the tendency for the patch from library site l to be picked when considering patch p of an example image of class c . This can be visualized as in figure 3.5c. A graphical model relating all of the variables is illustrated in figure 3.4.

3.3.2 Training

In this section, we consider how to use the training data $\mathbf{x}_{\bullet cp}$ from the p^{th} patch of all images belonging to the c^{th} class to learn a posterior distribution $Pr(\theta_{cp\bullet} | \mathbf{x}_{\bullet cp})$ over the relevant parameters $\theta_{cp\bullet}$. These parameters define a probability distribution over the library sites when considering patch p of an example image of class c . In section 3.3.3 we show how to use this distribution to calculate the integral in Equation 3.2.

We calculate the posterior distribution over the parameters $\theta_{cp\bullet}$ using a second application of Bayes' rule:

$$Pr(\theta_{cp\bullet} | \mathbf{x}_{\bullet cp}) = \frac{Pr(\mathbf{x}_{\bullet cp} | \theta_{cp\bullet}) Pr(\theta_{cp\bullet})}{Pr(\mathbf{x}_{\bullet cp})} \quad (3.5)$$

To simplify notation, we describe this process for just one of the P regular patches and one of the K classes and drop the indices c and p . Equation 3.5 now becomes:

$$Pr(\theta_{\bullet}|\mathbf{x}_{\bullet}) = \frac{Pr(\mathbf{x}_{\bullet}|\theta_{\bullet})Pr(\theta_{\bullet})}{Pr(\mathbf{x}_{\bullet})} \quad (3.6)$$

where $\mathbf{x}_{\bullet} = \mathbf{x}_1 \dots \mathbf{x}_I$ is all of the training data for this patch and this class and $\theta_{\bullet} = \theta_1 \dots \theta_N$ is the vector of N parameters associated with each position in the library for this patch and this class.

To calculate the likelihood of the i^{th} training example \mathbf{x}_i given the relevant parameters θ_{\bullet} we first find the closest matching library patch \hat{l}_i where

$$\hat{l}_i = \arg \max_l \mathcal{G}_{\mathbf{x}_i}[\mathcal{L}_l; \sigma^2 \mathbf{I}] \quad (3.7)$$

The data likelihood is a categorical distribution (one sample from a multinomial) over the library sites so that

$$Pr(\mathbf{x}_i|\theta_{\bullet}) = Pr(\hat{l}_i|\theta_{\bullet}) = \theta_{\hat{l}_i} \quad (3.8)$$

Now consider the entire training data \mathbf{x}_{\bullet} . The likelihood now takes the form

$$Pr(\mathbf{x}_{\bullet}|\theta_{\bullet}) = \prod_{i=1}^I Pr(\mathbf{x}_i|\theta_{\bullet}) = \prod_{i=1}^I \theta_{\hat{l}_i} = \prod_{l=1}^N (\theta_l)^{f_l} \quad (3.9)$$

where f_l is defined as

$$f_l = \sum_{i=1}^I \delta_{\hat{l}_i=l} \quad (3.10)$$

and $\delta_{\hat{l}_i=l}$ returns one when the subscripted expression $\hat{l}_i = l$ is true and zero otherwise. In other words, f_l is the frequency of the library site l being the closest matching patch, during the training process.

We also need to define the prior over the parameters θ in Equation 3.6. We choose a Dirichlet prior as it is conjugate to the categorical likelihood so that

$$Pr(\theta_{\bullet}) = \frac{\Gamma(\sum_l \alpha_l)}{\prod_l \Gamma(\alpha_l)} \prod_{l=1}^N (\theta_l)^{\alpha_l-1} \quad (3.11)$$

where Γ denotes a Gamma distribution and $\{\alpha_1 \dots \alpha_N\}$ are the parameters of this Dirichlet distribution. These are learned from a validation set (see section 3.5.2).

Substituting the likelihood (Equation 3.9) and the conjugate prior term (Equation 3.11) into Bayes' rule (Equation 3.6) we get an expression for the posterior distribution over the parameters which has the form of a Dirichlet distribution:

$$Pr(\theta_{\bullet}|\mathbf{x}_{\bullet}) = \frac{\Gamma(\sum_l (\alpha_l + f_l))}{\prod_l \Gamma(\alpha_l + f_l)} \prod_{l=1}^N (\theta_l)^{f_l + \alpha_l - 1} \quad (3.12)$$

We compute one of these distributions for each of the P patches in the regular grid and for each of the K classes.



Figure 3.6: Face preprocessing: (a) We manually place 2 landmarks on the face at the bridge of the nose and the top of the mouth. (b) We then warp the images to a 60x60 template using a Euclidean transformation. (c) We then band-pass filter the images using lower and upper cutoff frequencies of 2.5 and 25 cycles per image respectively and weight the pixels using a Gaussian function centered on the image. Finally, each image is normalized to have zero mean and unit standard deviation.

3.3.3 Calculation of Likelihood Integral

Finally, we substitute the posterior distribution over the parameters $Pr(\theta_{cp} | \mathbf{x}_{cp})$ (now resuming use of the indices c and p) into Equation 3.2 and integrate over θ_{cp} to get an expression for the likelihood¹ of observing test data patch \mathbf{y}_p given that the object class is c :

$$Pr(\mathbf{y}_p | \mathcal{C}=c, \mathbf{x}_{cp}) = \frac{f_{cpt^*} + \alpha_{l^*}}{\sum_l (f_{cpl} + \alpha_l)} \quad (3.13)$$

Note, Equation 3.13 can be thought of as a histogram over the library sites where the bins have been initialized with the values of α_l which are the hyper-parameters of the Dirichlet prior.

3.4 Databases

We test our algorithm on three datasets: faces, cells and pedestrians. We will describe each of these in the following sections.

3.4.1 Faces

We harvested a large database of images of men and women from the web. These were captured in uncontrolled environments and exhibit wide variation in illumination, scale, expression and pose as well as partial occlusion and background clutter (see figure 3.2a). Faces were detected using two methods: first, we used a commercial frontal face detector. Second, we manually labelled two landmarks. The former method does not localize the faces accurately and misses many of the harder non-frontal faces (it detected about 70% of the faces). The latter method localizes the images very accurately but includes all examples in the database regardless of their pose or quality.

For both methods, the images were subsequently transformed to a 60x60 template using a Euclidean warp (figure 3.6a,b). Note, sometimes the warping introduces blank areas in the

¹See Appendix A for derivation details.

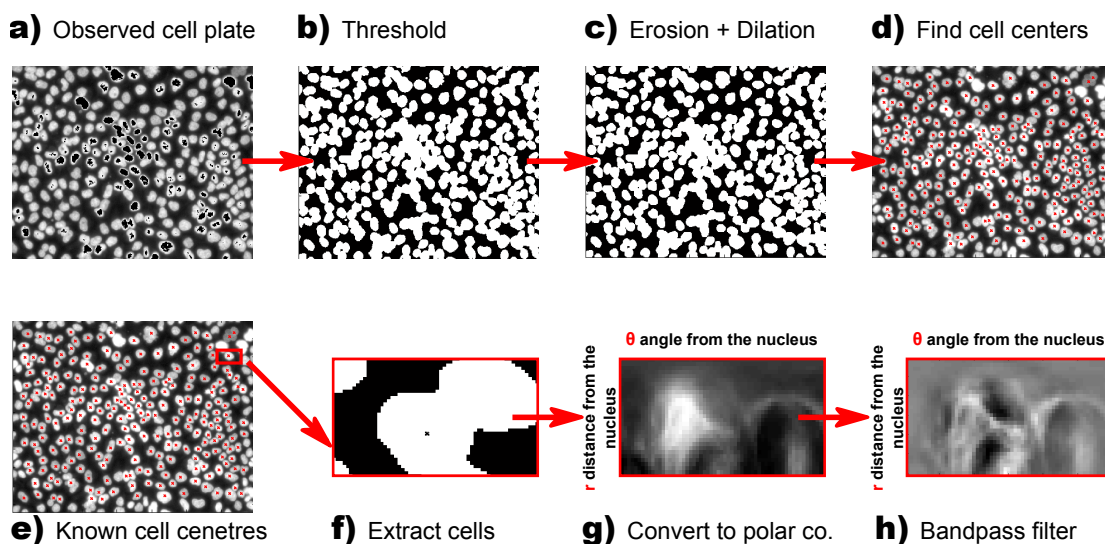


Figure 3.7: Cell preprocessing: (a) A plate of cells. (b) Threshold. (c) Perform morphological operations: erosion and by dilation. (d) Use connected component analysis to extract cell centers. (e) Place a 30x60 bounding box centered at each nucleus. (f) A 30x60 image containing a single cell. (g) Convert the cell image to polar coordinates by coping the pixel intensity at image x - y position in cartesian system to the new image in polar system. (h) Preprocess the image by Bandpass filtering.

image (i.e. when the warped position of a pixel falls outside the original image). We fill these pixels with 127 intensity in all three RGB channels resulting in gray regions in the final image. For example, see the images 3,6 and 7 in figure 3.11a. We band-pass filtered the images using lower and upper cutoff frequencies of 2.5 and 25 cycles per image respectively and weighted the pixels using a Gaussian function centered on the image. Each image was normalized to have zero mean and unit standard deviation (figure 3.6c). This preprocessing was chosen as it showed superior results on a validation set compared to the results of images with no preprocessing and an alternative preprocessing algorithm proposed by Tan and Triggs [160] (see section 3.5.1 for the details of our preprocessing experiment).

3.4.2 Cells

The MRC cell database [13] contains images of human cancer cells (HeLa-Kyoto) displaying a large variety of morphological phenotypes after the individual knockdown of approximately 500 genes. Of these many morphological changes, we were interested specifically in two phenotypes: (i) when the borders of the cell change significantly in response to the knockdown, and produces a ‘triangular’ phenotype with sharply-edged borders, and (ii) when knocking down a gene had no effect on the cell, leaving its phenotypic appearance as non-triangular/amorphous (‘normal’).

Each image contains 3 color channels: W1,W2,W3, each of which represents a different

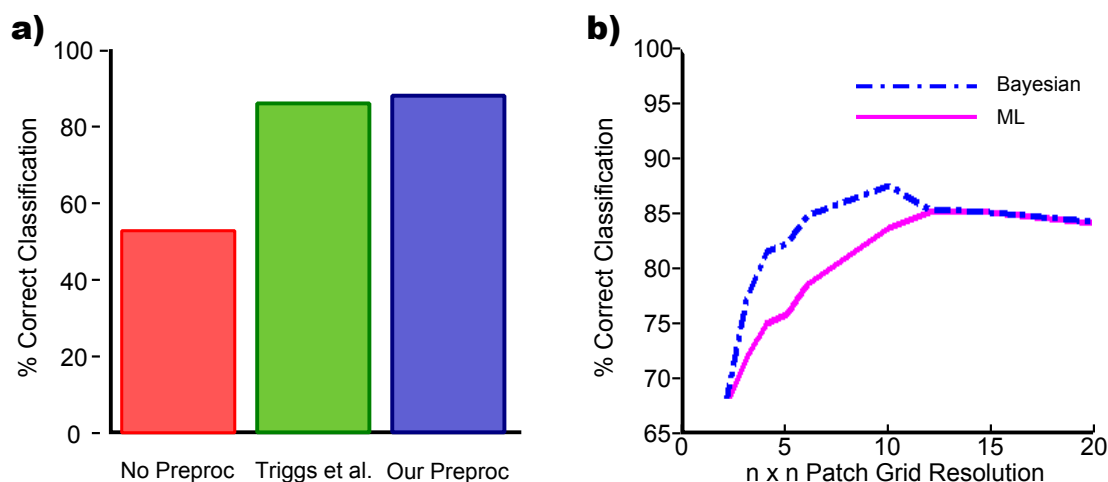


Figure 3.8: Preprocessing method and parameter selection: (a) The performance of the gender classification algorithm was tested with three different preprocessing methods. The results show that our preprocessing algorithm described in sections 3.4.1 and 3.4.2 for faces and cells, outperforms other methods. (b) To find the best patch size, gender classification is carried out on the validation set on several patch grid sizes. The performance peaks at the 10x10 grid resolution.

fluorescent stain. We use the W1 channel to find the nuclei: we threshold the image (figure 3.7b) and then use morphological opening to remove noise (figure 3.7c). We find connected regions and take their centroids to represent the nucleus position (figure 3.7d). We place a 60×60 pixel bounding box around the center and extract the data from the W2 channel for classification (figure 3.7e,f). Since cells exhibit radial symmetry we convert the images to a 60×30 polar representation, so that the horizontal coordinate of the new image contains the angle from the nucleus center and the vertical coordinate represents the distance from the nucleus. This allows us to easily constrain patches from the library to only match to patches at similar radii without regard for their polar angle (figure 3.7g). These radial images were band-pass filtered and normalized to have zero mean and unit standard deviation (figure 3.7h).

3.4.3 Pedestrians

We collected a large database of urban scenes. Pedestrians were automatically detected using the method of [55]. The images were then manually labelled for pose: pedestrians facing front, back, left and right. The bounding box around each pedestrian was re-scaled to create a 60×120 image. The pedestrian images were band-pass filtered and normalized as with the face and cell images.

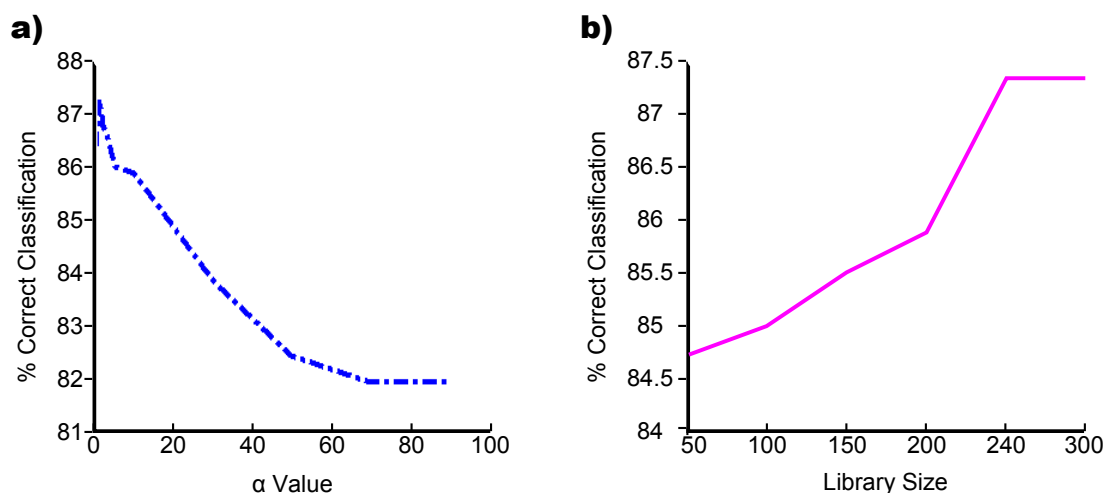


Figure 3.9: Parameter selection: (a) The α the hyper-parameter of the Dirichlet prior is chosen empirically by testing the performance of the algorithm on the validation. The performance peaks at the value of $\alpha = 2$. (b) To find the best library size, gender classification is applied on a validation set, where we vary the size of the library by keeping the patch grid fixed at 10×10 and the alpha = 2. The peak performance is reached at a library of size 240.

3.5 Experiments

3.5.1 Experiment 1: Choosing The Best Preprocessing

To choose a suitable preprocessing method, we performed a gender classification experiment, where we varied the preprocessing method through three different states by keeping all the other variables constant. We used a training set of 8,000 male and 8,000 female and a validation dataset of 400 male and 400 female images. First, the algorithm was tested on data with no preprocessing, i.e. the raw pixel intensities. Second, the preprocessing described in section 3.4 was performed on the data before testing. Third, the preprocessing algorithm proposed by Tan and Triggs [160] was performed before testing. This method involved a sequence of Gamma correction, difference of Gaussian (DoG) filtering, masking and contrast equalization steps. The results are plotted in Figure 3.8. It is clear that performing preprocessing improves the results significantly when compared to using raw pixel intensities. The other two preprocessing methods perform almost equally, with our method producing a modest improvement. We expect these results to generalize on our other data sets. Thus, in the remaining experiments in this chapter we will apply our preprocessing method as explained in section 3.4.

3.5.2 Experiment 2: Parameter Selection

In this experiment, we use a validation set to investigate the effect of patch grid resolution and the Dirichlet parameters $\{\alpha_1 \dots \alpha_N\}$ for gender classification in the face images. We used a

training set of 8,000 male and 8,000 female faces, a validation set of 400 male and 400 female images, and a library of 240 images uniformly sampled from both classes.

Figure 3.8b shows the percentage correct classification as a function of the patch grid resolution. The results show that performance increases as the patch grid gets finer, peaking at a 10×10 grid (6×6 pixels) and then declining. When the patches are very small, they are probably not sufficiently informative. For comparison, we also plot results from a maximum likelihood approach where we form a point estimate of the parameters θ_{cpl} . We note that this approach produces noticeably worse results. This is because our training set is not large enough to give a fair probability distribution over the library patches. Consequently, most of the library patches ($\sim 70\%$) end up having a zero probability given each of the classes, which hinders the predictions for test images. Conversely, in a Bayesian approach the α hyper-parameter solves this problem by allowing an initial frequency on each of the library patches.

In figure 3.10, we verify that 6×6 pixel patches are sufficient to capture information about gender, by reconstructing real images using the closest patches l^* from the library. It is still easy to identify the gender of the images using the approximated versions. We also show reconstructed images of pedestrians with the same size patches and observe that the poses of the pedestrians can easily be seen in these images.

Figure 3.9a shows the percentage correct classification using 6×6 pixel patches as a function of the Dirichlet hyper-parameters $\{\alpha_1 \dots \alpha_N\}$ which are constrained to all be the same value, since we do not have a particular preference (or prior knowledge) as to which library site should be weighted higher. The results show a significant jump in performance when the α value changes from 1 to 2 but then decline. This is also a confirmation of the Bayesian inference being beneficial, since the maximum likelihood solution can be seen as a special case of Bayesian inference when $\alpha = 1$. For the rest of this chapter we adopt these optimal parameters: we use a patch resolution of 6×6 and set $\{\alpha_1 \dots \alpha_N\} = 2$.

Finally, to find the best library size we repeat the above experiment where we vary the library size from 50 images to 300 images, while keeping the patch grid fixed at 10×10 and the α value fixed at 2. Figure 3.9b shows that the performance increases with the size of the library, reaching a peak at 240 and converging afterwards. Due to time limitations the maximum number of library images tested was 300. Moreover, this experiment was performed by randomly selecting a set of images only once for each library size. Ideally, to get an estimate of the uncertainty in the library size, one should repeat this experiment several times for each library size and report the mean percentage correct classification as well as the standard deviation at each size. A library of size 240 seems to be sufficient for our current experiments. However,

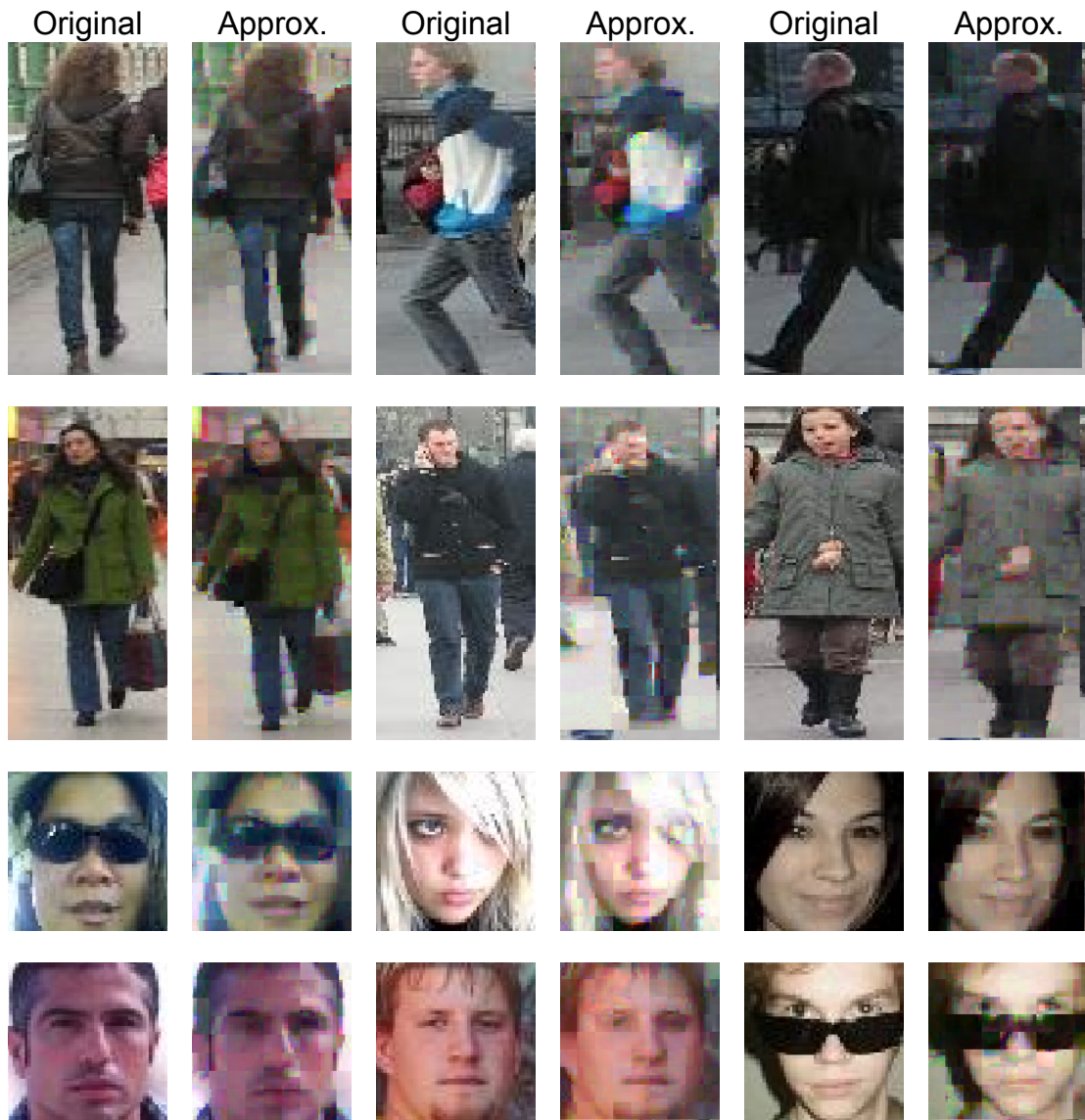


Figure 3.10: *Patch approximation:* Comparison of original images and best approximations l^* from library patches.

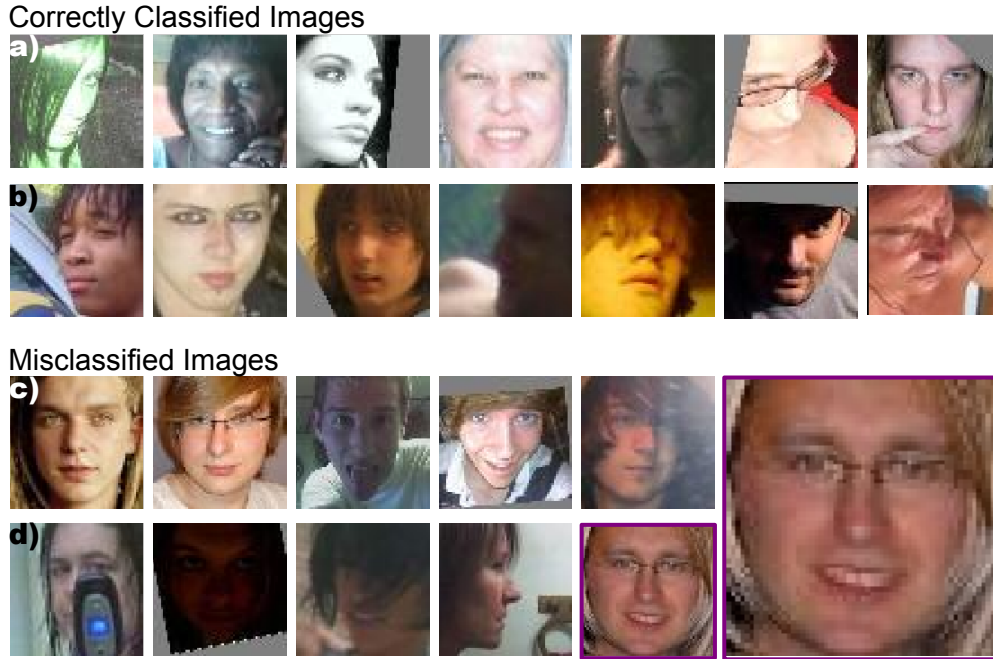


Figure 3.11: Gender classification Results on manually detected faces: We achieve 89% correct classification. (a) Correctly classified females. (b) Correctly classified males. (c) Males misclassified as female. (d) Females misclassified as male. (e) Close up: interesting misclassified case.

if the training data is increased to for example, a million images instead of a few thousand, we believe that the library size should also increase accordingly.

In Equation 3.7 we defined a hard assignment (MAP) of training patch \mathbf{x}_i to library index \hat{l}_i . In principle it would be better to marginalize over possible values of \hat{l}_i , but this is intractable. We found experimentally that it was possible to slightly improve performance by using a soft assignment of library sites, replacing Equation 3.10 with

$$f_l = \sum_i \frac{\mathcal{G}_{\mathbf{x}_i}[\mathcal{L}_l; \sigma^2 \mathbf{I}]}{\sum_{j=1}^N \mathcal{G}_{\mathbf{x}_i}[\mathcal{L}_j; \sigma^2 \mathbf{I}]}, \quad (3.14)$$

We have done this for all results in this chapter. In practice, we also restrict the possible indices l to a subset corresponding to a 6×6 pixel window around the current test patch position in each library image so patches containing eyes are only approximated by other patches containing eyes etc. This reduces the number of parameters, and the model only allows plausible deformations.

3.5.3 Experiment 3: Gender Classification

In this experiment we investigate gender classification for both the manually and automatically detected face datasets. In each case, we use a training set of 16,000 male and 16,000 female images. The test set contains 500 male and 500 female faces and the library is made up of 120 male and 120 female images.

We achieve an 89% correct recognition rate on the manually detected dataset. Figures 3.11a and b show correctly classified female and male examples respectively. Note that the images contain large pose variations ranging from -90° to $+90^\circ$. Figure 3.11c shows typical examples of male images misclassified as female. Notice that these images have no facial hair and some have long hair. The third person is pulling a face which was seen more often in female training examples. Figure 3.11d shows typical examples of female images misclassified as male. Many of these images are partially obscured or low quality. The fourth image (blown up in 3.11e) is particularly interesting. This was tagged as female but we suspect it is a man in a wig! Gender classification was performed with a similar protocol on automatically detected images. Here, we achieve 90% correct classification. This dataset shows less variation in head orientation but the position of the face varies more in each image. Examples of correctly classified faces were shown in figure 3.2a.

We compared our gender classification results with the performance of 10 human subjects on the same test set. For this purpose each subject was shown the same test images as used for our gender classification experiment on manually detected dataset. The images were 60×60 in size and grayscale but did not undergo further preprocessing. The average human performance was 95.6% which is only 7% higher than what our algorithm achieves.

We also tested the classification ability of each patch individually. Figure 3.13b shows the percentage correct classification for each patch as a gray level image (the higher the performance, the lighter the patch). Notice that there are no dominant patches with high discriminative power. Instead a collective decision based on all of the patches is made for classification of gender.

3.5.4 Experiment 4: Eyewear Classification

We also investigate the task of determining whether people were wearing glasses. The training set for this experiment contains 8,000 images with glasses and 8,000 images without. The library contained 120 images with glasses and 120 without. The algorithm was tested on 400 images with glasses and 400 without. We achieve 84% correct classification. Figure 3.13c shows the percentage correct achieved based on each patch alone. As expected, there is far more discriminatory power in the top half of the image. We repeated the experiment using only the top half of each image and achieved 91.2% classification.

Figure 3.12a shows images correctly classified as ‘without glasses’ despite some images being dark (images 1,2) or the eyes being covered by hair (images 3-5). Figure 3.12b shows images correctly classified as wearing glasses, despite the images being very bright (image 4), blurry (images 5,6), or non-frontal (image 3). Misclassified images are shown in figures 3.12c

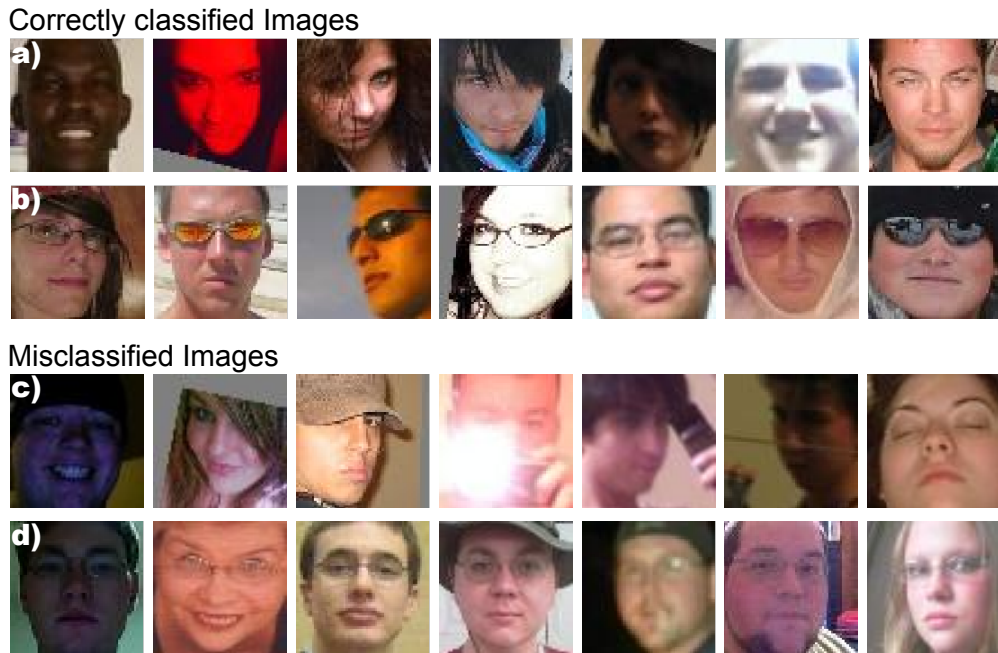


Figure 3.12: Eyewear classification results: We achieved 91.2% correct classification of the presence of eyewear. (a) Correctly classified as without glasses. (b) Correctly classified as wearing glasses. (c) Without glasses but misclassified as wearing glasses. (d) Faces with glasses but misclassified as not wearing glasses.

and d. Many of the images misclassified as wearing glasses (figure 3.12c) have obscured eyes or were wearing a cap. Most of the images misclassified as being without glasses (figure 3.12d) were wearing frameless reading glasses which are difficult to distinguish.

We compared our results with the performance of 10 human subjects on the same test set. Their average performance was 96.79% correct classification, which is only around 6% higher than what our algorithm achieves. We note that most of the images misclassified by humans as without glasses were also wearing frameless reading glasses.

3.5.5 Experiment 5: Age Classification

In this experiment we study the problem of age classification. Given a test image the goal is to assign it as belonging to one of four age categories: (i)18-25, (ii)26-32, (iii)33-40, (iv) 40Plus. The training set contained 5500 images of men and women from each age group. The library contained 240 images uniformly collected from each category. The algorithm was tested on a set of 1000 images equally distributed in each category. We also asked 10 human subjects to predict age in the same 1000 test images. The results were stored in a confusion matrix, where correct vs estimated categories are shown. Table 3.1(b), summarizes the results for both our algorithm and human subjects. The table shows that, for most age groups, our algorithm assigned the highest probability to the correct category, whereas the human subjects consistently

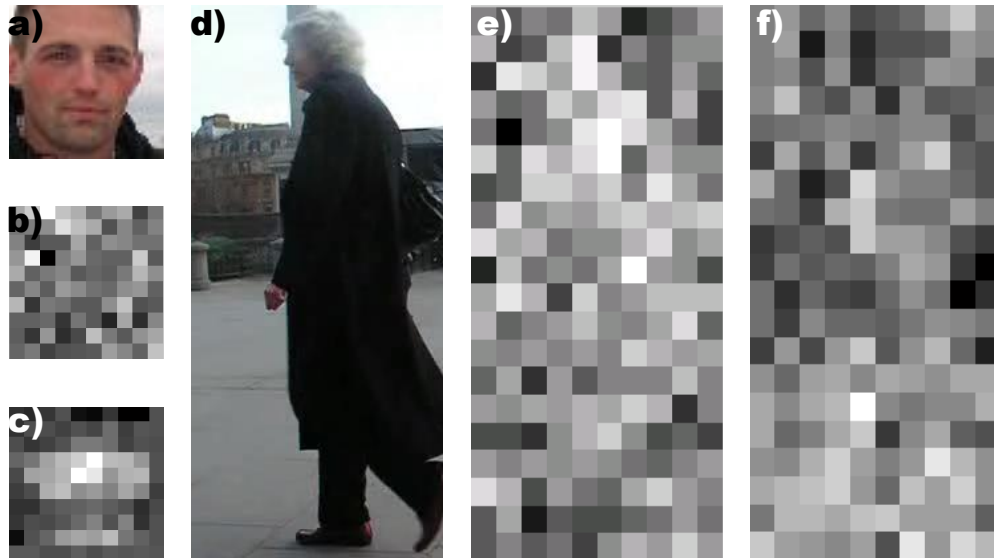


Figure 3.13: Per-Patch classification results: Classification performance was tested per patch. The images (b,c,e,f) show % correct classification as a gray scale image i.e the brighter the image, the higher the % correct. (a) Sample face image.(b) % correct performance per patch for gender (c) % correct performance per patch for eyewear. (d) Sample pedestrian image. (e) % correct performance per patch for classifying pedestrians as facing front vs. back. and (f) % correct performance per patch for classifying pedestrians as either facing front & back vs. facing left & right.

assigned the highest probability to the second age group. It is interesting to observe that despite the low performance (on average 42.5%), our algorithm has managed to learn an association between an image and its age group, each time giving the correct category a higher probability, as opposed to our human subjects who have mainly gone for the ‘safe option’ of the middle category and scored an average of 36.5% correct classification.

To examine this issue further, a second evaluation metric was used, where we considered correct classification if the image was classified as either the correct age group or the adjacent age groups (table 3.1(a)). Using this second metric, shows that our performance is comparable to human performance with average performance of $\sim 70\%$ for both our method and human subjects (table 3.1(b)). Overall, this experiment suggests that age estimation is a very challenging task and may not be suitable for a classification algorithm. In fact, age is really a continuous quantity and regression methods should be applied for this task. Unfortunately, we do not have the ground truth to do this.

3.5.6 Experiment 6: Cell Phenotype Classification

In this section, we apply our algorithm to a second object class with completely different properties. We classify human cells from a subset (141 plates) of the MRC cell database [13] as being either ‘triangular’ or ‘normal’. During training, 12500 cells from 125 plates were used

		Our Method				Human Performance			
True \ Est.	18-25	26-32	33-40	40 Plus	18-25	26-32	33-40	40 Plus	
	18-25	44%	25.5%	16.5%	16%	23.8%	37.1%	26.7%	12.4%
26-32	23.0%	46.5%	16%	14.5%	22.7%	37.9%	28.2%	11.2%	
33-40	16.0%	26.5%	26.5%	31%	23.6%	36.5%	27.2%	12.7%	
40 Plus	16.5%	19.5%	16.5%	47.5%	24%	35.1%	29.8%	11.1%	

Table 3.1: Age classification confusion matrix: Comparison of results between our algorithm and human performance. The rows show the correct category and the columns show percentage classification as belonging to each of the categories.

		(a)				(b)		
True \ Est.	18-25	26-32	33-40	40 Plus	True Class	Ours	Human	
	18-25	44%	25.5%	16.5%	16%	18-25	69.5%	60.9%
26-32	23.0%	46.5%	16%	14.5%	26-32	85.5%	88.8%	
33-40	16.0%	26.5%	26.5%	31%	33-40	84.0%	76.4%	
40 Plus	16.5%	19.5%	16.5%	47.5%	40 Plus	64.4%	40.9%	

Table 3.2: (a) A second metric used to evaluate age classification. For each row if the age group adjacent to the correct age group is picked (highlighted areas), it is considered as correct classification. (b) Comparison of our method and human performance on age classification using this metric.

for each class. The library contained 120 cells from each class.

The first experiment tests the ability of our algorithm to classify single cells. We used a test set of 500 normal and 500 triangular cells. The method achieved 70% correct classification. We note that (i) this is a very difficult task as the cell images contain significant within class variation (see figure 3.14) (ii) not all cells in a given plate are affected by the experimental conditions that cause changes in cell shape so we do not expect perfect performance and (iii) biologists are usually interested in classifying entire plates (images) each of which contains 50-150 cells.

This motivates our second experiment in which we classify the entire plates. Due to limited amount of data there were only 16 plates (8 from each class) that were not used either in the training set or the library. We break these plates into 4 sub-images, resulting in 64 sub-plates which were used in testing. We treat each cell within each subplate as providing independent

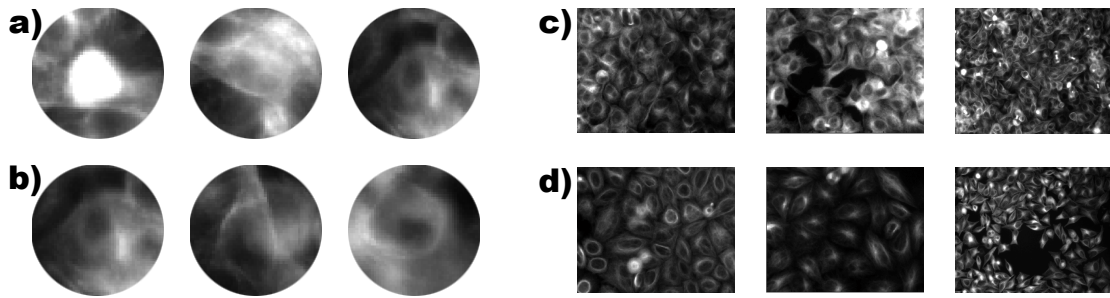


Figure 3.14: Cell classification results: *W2 channels of individual cells that were correctly classified as (a) normal and (b) triangular. (c) A correctly classified normal plate. (d) A correctly classified triangular plate. Interestingly, biologists usually classify cells based primarily on the W3 channel. Our algorithm seems to be exploiting information that is not particularly salient to human experts.*

information about the plate class. Under these conditions, the algorithm managed to achieve 100% correct classification rate by classifying all 64 sub-plates correctly. Example classifications for cells are shown in figure 3.14a-b. Subplate classification is shown in figures 3.2b and 3.14c.

3.5.7 Experiment 7: Pedestrian Pose Classification

Finally, we test our algorithm on classifying pose in pedestrian images. In training, we use 3000 images of pedestrians from each of the four classes (i) facing front, (ii) facing back, (iii) facing left and (iv) facing right. We use a library of 240 images (60 from each class). We devise four separate experiments.

In the first experiment we do multi-class classification using a test set of 1200 images (300 per class). In this experiment we classify a test image as belonging to one of the four classes. We achieve 67% correct classification overall. Table 3.3 shows the confusion matrix where each row shows the true label and each column shows the estimated label. It is notable that left facing pedestrians are most confused with right facing ones and front facing pedestrians are most confused with back facing ones. Figure 3.15 shows examples of correct and wrong classifications. In the second experiment we examine binary classification to distinguish only front-facing from back-facing examples. We tested on 600 images (300 back, 300 front) and we achieve 75% correct classification. This is quite a challenging task as these two classes are largely distinguishable only from the facial area. This is verified when we examine the per patch classification (see figure 3.13e): patches in the top center of the image are most informative.

In the third experiment we classify test images as either facing left or facing right. We achieve 81.2% correct classification. Finally we test our algorithm on classifying pedestrians as either facing left/right, or facing front/back. In this experiment we achieve 85.3% correct clas-

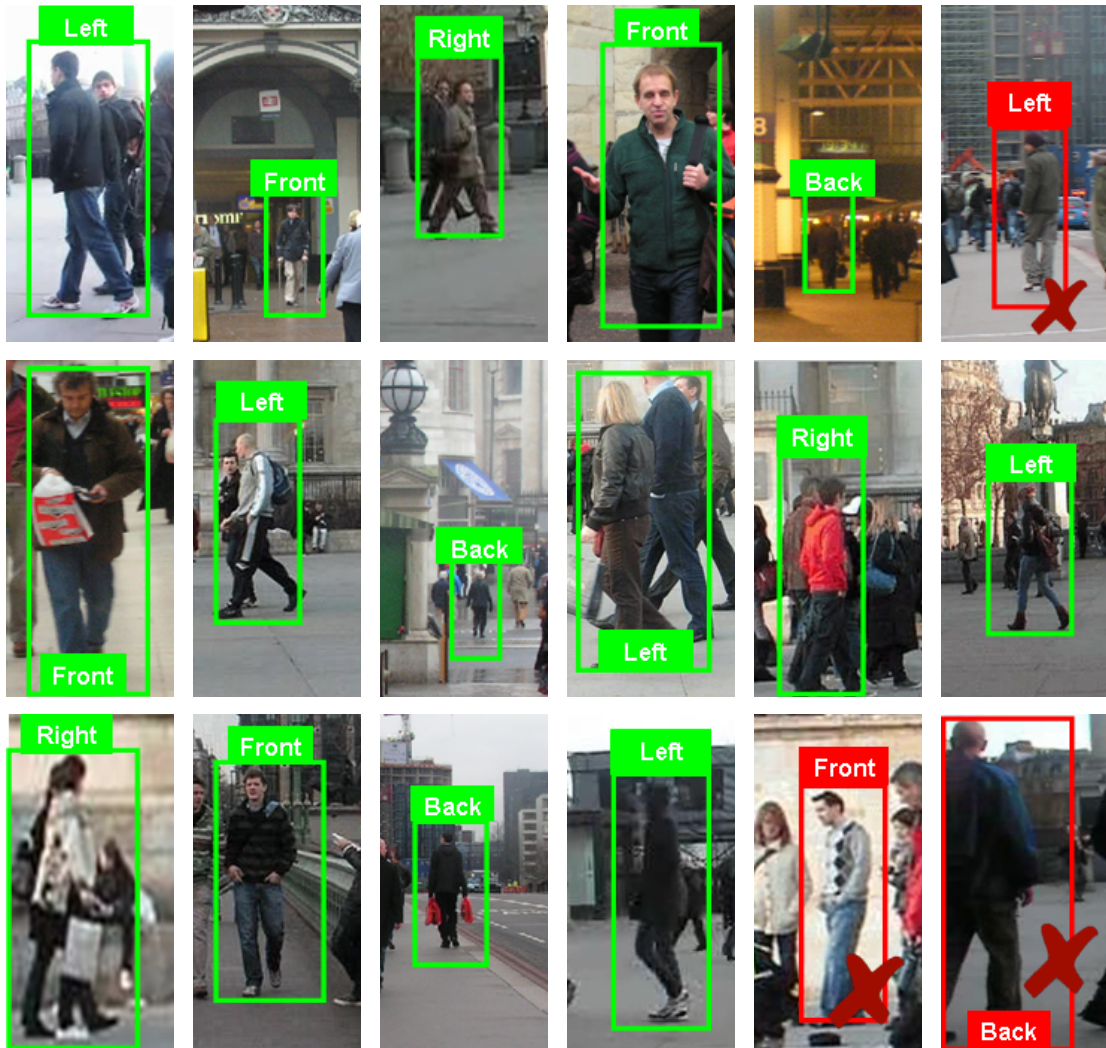


Figure 3.15: Pedestrian pose classification results: Example results from the pedestrian pose classification experiment. We show the predicted label for each pedestrian. The two images marked by a red cross have been misclassified, but the remaining images show correctly classified pedestrians.

Est. \ True	Back	Front	Left	Right
Back	77.7%	10.0%	5.6%	6.7%
Front	35.6%	53.7%	6.7%	4.0%
Left	7.3%	6.7%	71.0%	15.0%
Right	12.0%	8.3%	15.0%	64.7%

Table 3.3: Confusion matrix for pedestrian pose classification.

sification. We plotted the per patch classification as a grayscale image in 3.13f. Unsurprisingly, this figure shows that the most discriminative patches for this task are ones towards the bottom of the image: the legs are the most distinctive part of the image to distinguish these classes.

3.5.8 Experiment 8: Comparison to Other Algorithms

We compare the results of our algorithm on gender classification with a recent method proposed by Prince and Aghajanian [138] based on additive logistic models. This is a discriminative classifier, which is based on an additive sum of non-linear functions of one-dimensional projections of the data such as the arc tangent and weighted sums of Gaussians. In this experiment, we used the face images from the automatically detected face dataset. We used a training set of 16,000 male and 16,000 female images, and the test set contains 500 male and 500 female faces. The discriminative classifier achieves 87.5% correct classification whereas our algorithm proves to be superior by scoring 90% correct classification when trained and tested on the same datasets.

For further validation we compare the performance of our gender classification algorithm with the manually registered dataset to that of support vector machines (SVMs) which a standard pattern recognition method. Unfortunately, SVMs were not designed to work with large databases and it is hard to train with the high resolution (60×60) images due to the memory requirements. To get the best out of these methods we have used both (i) the maximum feasible number of training images at high resolution (4000 images per class) and (ii) a larger training set (16000 images per class) of low resolution images which were subsampled to 21×12 . This is similar to images used in other gender classification algorithms such as [124]. However, such small images are unsuitable for our algorithm as we require the patches to be of a certain size (6×6) to be informative.

For the first case (4000 high resolution images per class) a linear SVM and a non-linear SVM with an RBF kernel achieved 78.8% and 77.8% performance respectively. The SVMs were trained with libsvm [2] and the parameters selected with 3-fold cross validation. When we

tested our method with only these 4000 training images we achieved 84.6% which is considerably better than either of the SVM methods.

For the second case (16000 low resolution images per class) the linear SVM achieved 78.7% performance and the non-linear SVM achieved 82.4%. For this dataset we also tried linear discriminant analysis which achieved a maximum of 78%. None of these results approach the 89% performance achieved by our algorithm (See Table 3.4).

Finally, we also compared human performance on gender classification. The average human performance was 95.6%. Although our best performance is 7% lower than this, we conclude that some of the test images are genuinely difficult to classify. Moreover, our results reveal that the human performance is not perfect (i.e. 100%) when looking at the face region only, this suggest that humans seem to rely on context from other parts of the scene, to predict gender.

Method	% Correct Classification
Lin SVM HighRes 4,000	78.8%
RBF SVM HighRes 4,000	77.8%
Our Method 4000	84.6%
LDA 16,000	78%
Lin SVM LowRes 16,000	78.7%
RBF SVM LowRes 16,000	82.4%
Our Method 16,000	89%
Human Performance	95.6%

Table 3.4: *Confusion matrix for pedestrian pose classification.*

3.6 Summary and Discussion

In this chapter, we have begun addressing our first research objective of investigating patch-based representation for a variety of vision tasks. As the first of these tasks we attempted to classify several attributes of visual objects within a category. In the past, within-category classification has mostly been tested on constrained databases, which contain simplified images with limited or no variation in lighting, pose, background clutter etc. To address this issue, we collected and annotated a large database of $\sim 70,000$ faces in uncontrolled conditions. This is a much more challenging database, with realistic images taken in everyday scenarios. We subsequently used it to predict demographic characteristics such as gender and age, and appear-

ance attributes such as eyewear. As such, we have made progress towards our other research objectives: we have demonstrated the ability of the patch-based representation to handle large databases and their robustness to visual object modelling challenges such as occlusion, scale, background clutter, and variation in pose and lighting.

We proposed a general Bayesian framework for classifying within-category attributes. Our algorithm uses a generic patch-based representation, allowing it to be used on several object classes without major alterations. We exploit this property by predicting different attributes in three distinct object classes. Besides predicting gender and eyewear in face images, we predict phenotype in cells and pose in pedestrian images. We demonstrate good performance on ‘real world’ images of all three classes. We achieve 90% correct classification of gender, and 91.2% correct classification of eyewear in face images. The results are quite satisfactory when compared to human performance on these tasks on the same database, which were 95.6% and 96.7% respectively. We achieve 67% and 100% in pedestrian pose and cell phenotype classification respectively. The promising results generated by our patch-based representation model in all three object classes contributes towards our final research objective of having a representation that is not object-specific and is applicable to multiple object classes.

Our algorithm has been designed to handle large amounts of data. In terms of scalability, it is linear with respect to the size of the library and the training data. For a library of size m and a training set of size n it scales as $O(mn)$ during training and $O(m)$ in testing. Current implementation of our code in Matlab takes about 2 seconds per image on an Intel Xeon(R) CPU E5420 @2.50 GHz machine with a 2.5 GB RAM. The most computationally expensive part of our model is the 2D convolution to map the test/train patch to the library patches which could be easily speeded up using a Graphics Processing Unit (GPU). Moreover, the same library can be used to predict more than one attribute, resulting in further savings of computational time. For example, if the training images are divided into four groups of females with and without glasses and males with and without glasses, once a test images is mapped to the library, our algorithm can predict both the gender and the eyewear of the test image.

Other properties of our method include “automatic registration” and avoiding overfitting. The patches are allowed to map to the library within a 6×6 window of the original position of the patch in the test image. This automatically finds the best horizontal and vertical shift of the original patch, compensating for possible translations. Finally, the Bayesian formulation of our method where we marginalize over the parameters helps guard against overfitting.

The algorithm has a close relationship with non-parametric synthesis algorithms such as image quilting [46] where patches from one image are used to model others. Our algorithm

works on exactly the same principles - all the knowledge about the object class is embedded in the library images. This accounts for why the algorithm works so well in different circumstances. If we have enough library images, they naturally provide enough information to discriminate the classes. To this end our algorithm works in a similar fashion as most of the data-driven algorithms such as [92, 47] where millions of images are used to derive information about other unseen images.

The algorithm also has a close relationship with bag-of-words models [38]. The library can be thought of as a structured set of visual words (textons) which are used to quantize the image patches. Unlike the standard bag-of-words models, the visual words are arranged on a regular grid, giving implicit information about the spatial configuration of the words. To the best of our knowledge the original bag-of-words representation has not been used for within-category classification. We suspect it may perform poorly, since generic visual dictionaries used in the model, may not be able to distinguish subtle differences between two instances within a category. For example, the bag-of-words model may easily distinguish a face from a car, however, it is not clear whether it can tell the difference between a male and a female face.

Our model also has connections to exemplar-based models such as ‘Visual Memex’ [121] which encode both local appearance and 2D spatial context between object instances. In the ‘Visual Memex’ model, objects are represented as a set of exemplar nodes, that are connected together based on their similarity and context. The library patches can be thought of as the exemplar nodes randomly sampled from all object classes. The mapping of each training patch to the library patch, is equivalent to the ‘similarity edge’, where the distance function in [121] is replaced by a Gaussian likelihood function. Finally, the ‘context edge’ is explicitly achieved by restricting the patches to be mapped only from a similar region in the original training patch.

Some aspects of our model remain unexplored. The model has not been tested with a very large library. We expect that a larger library will be beneficial since it will contain more variety for each class. It would be particularly interesting to learn the optimal library. The idea of learning an optimal library is very similar to learning the visual dictionary in Bag-of words models. Methods such as ‘Gaussian Mixture Models’ (GMMs) [171], or discriminative methods based on ensembles of random clustering trees such as ‘Extremely Randomized Clustering Forests’ [129], which can be used to learn a compact library which could provide faster training and testing. Finally the model can be extended to overlapping patches to gain a richer representation, which would be desirable for generation tasks.

One of the less successful problems investigated with our model was age classification. This is a genuinely difficult problem given that our results show that human performance on

this problem is $\sim 36\%$. In our experiments the age estimation was formulated as a classification problem with four age groups. However, age estimation is really a regression problem where the exact age can be predicted. Unfortunately, current format of our algorithm does not allow prediction of continuous parameters since the algorithm is designed to have discrete outputs. This limits the application of our model to classification problems only.

In the following chapter, we will extend the model to allow continuous outputs, making it suitable for regression type problems.

Chapter 4

Patch-based Regression

4.1 Introduction

In the previous chapter we showed that patch-based representation can be used in a probabilistic framework for within-category classification. Although this framework could successfully predict discrete attributes (e.g. gender), it was not designed to estimate continuous parameters.

In this chapter we present an extension to our within-category classification model, where we use a patch-based representation to estimate a continuous parameter. Upon successful completion of this task, we aim to progress towards our goal of having a representation that can be used for a variety of vision tasks. The approach considered here is a general model that is suitable for all regression type problems where the output is a continuous parameter. We will test our model on the specific problem of estimating pose. In particular, we will apply the model to face pose estimation where the pose of the face images varies from -90° to 90° , and the faces are captured in uncontrolled environments.

4.2 Motivation

Automatic estimation of head pose from a face image is a sub-problem of human face analysis with widespread applications such as gaze direction detection and human computer interaction. It can also be integrated into a multi-view face detection and recognition system. Another interesting application is driver surveillance, where the driver's head is constantly monitored for an unusual pose. For example, a warning can be issued if the head is regularly tilted down due to the driver's sleepiness. Similarly, a face pose estimation system can be used for video conferencing and designing a smart room that monitors its occupants' activities. There have been various approaches to this problem using stereo or multi-view images [122, 130], range images [23] and tracking using video sequences [78], etc. In this chapter we focus on estimating head pose from a single 2D face image.

Current methods for face pose estimation from a 2D image can be divided into two groups:

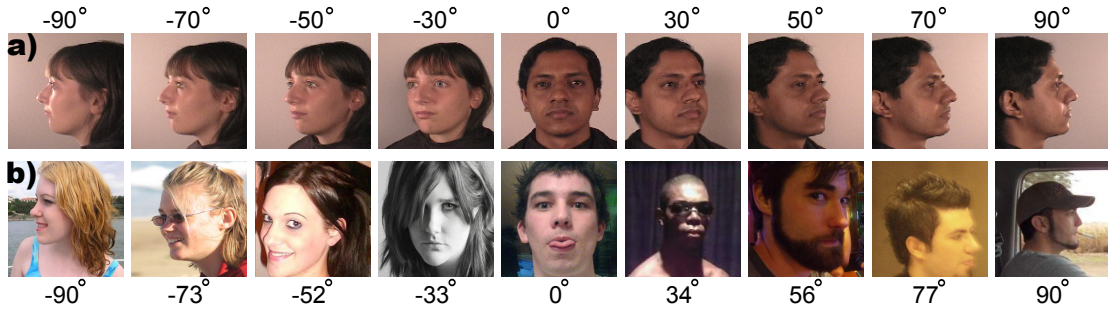


Figure 4.1: Face pose estimation: classification vs regression(a) Most current methods estimate pose in a limited range or treat pose as a classification problem by assigning the face to one of many discrete poses. Moreover they have mainly been tested on images taken in controlled environments e.g. with solid/constant background, with small or no variation in illumination and expression. (b) We address the problem of estimating pose as a continuous regression problem on “real world” images with large variations in background, illumination and expression.

(i) geometric shape or template based methods [166, 83, 177, 95] and (ii) manifold learning and dimensionality reduction based methods [10, 65, 146, 132]. The first group use the geometric information from a configuration of a set of landmarks to estimate pose. For example, some use the relative position of the eyes, mouth, nose etc. in piecewise linear or polynomial functions [83], or in an Expectation Maximization (EM) approach [31] to recover the face pose. Others fit a template to the face such as an Active Shape Model (ASM) and estimate the pose parameters using Bayesian inference [177]. Other common approaches include fitting an elastic bunch graph [95, 48] to a certain pose and using graph matching to decide on the particular pose. A major limitation of these methods is that they rely on finding the position of facial features, or fitting a tailor-made template to the face which itself is a difficult problem [117].

Alternatively, manifold learning approaches consider the high dimensional feature space of face pose as a set of geometrically related points lying on a low dimensional smooth manifold. They use linear/non-linear embedding methods to learn this lower dimensional space from the training data. Pose is then estimated in this space. Several general linear embedding methods such as PCA [132] and ICA [110] have been proposed. One limitation of these methods is that they can only recover the true structure of *linear* manifolds, while the geometry of the view-varying face manifold can be highly folded in the high-dimensional input space.

Alternatively non-linear embedding such as Isomap [10], Local Embedded Analysis (LEA) [65] and Local Linear Embedding (LLE) [10, 146], have been used to project the data onto a lower dimensional non-linear manifold. Pose is then estimated using K-nearest neighbor classification [65], or multivariate linear regression [10]. These methods are limited in that

the non-linear projection is defined only on the training data space. For each novel input (test data) the entire embedding procedure has to be repeated or other techniques like Generalized Regression Neural Networks [10] need to be designed to do this mapping for test data.

One of the limitations of current methods is that most of them estimate pose in a limited range and treat pose estimation as a classification problem by assigning the face to one of many discrete poses [95, 109, 110]. However pose estimation is really a regression problem. Consider a human computer interaction scenario where we are interested in finding the gaze direction of the user; it is much more desirable to have a continuous estimation of the head pose rather than specific head angles. Another major drawback of current methods is that they have mainly been tested on faces taken in controlled environments (there are exceptions e.g. [71]) i.e. with solid or constant background and small or no variation in illumination and expression such as the CUbiC FacePix database [113] (figure 4.1a). Ideally we should be able to estimate pose in uncontrolled environments. Unfortunately current methods are not capable of this, partly because it is very difficult to obtain the ground truth for such images.

In this chapter we address both problems of obtaining human estimates of pose (as ground truth) and automatically estimating pose in images taken in uncontrolled environments. First we collect a large database (tens of thousands) of “real world” images. This database contains faces with poses varying from -90° to 90° as well as high variation in illumination, expression and background clutter (see figure 4.1b). Then, we ask four human subjects to label these images for pose according to some reference poses. Finally, we compare our automatically estimated pose with human estimates of pose in these real world scenarios.

Unlike current methods that have tailor-made representations, we use a general representation that does not rely on locating facial features or fitting a model to the face. Instead we represent a face with a non-overlapping grid of patches. We use this representation in a generative model for automatic estimation of head pose.

4.3 Method

Our approach breaks the test image into a non-overlapping regular grid of patches. Each is treated separately and provides independent information about the true pose. At the core of our algorithm is a predefined library of object instances. The library can be considered as a palette from which image patches can be taken. This is similar to the *bag-of-words* model [38]: the library can be thought of as a structured set of textons which are used to quantize the image patches. We exploit the relationship between the patches in the test image and the patches in the library to estimate the face pose. Our algorithm can be understood in terms of either *generation*

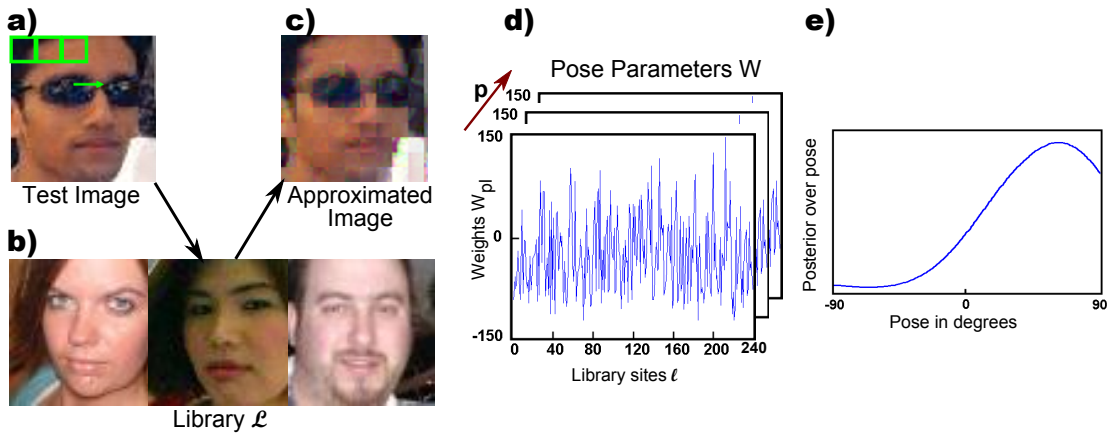


Figure 4.2: Inference (a) A test image \mathbf{Y} is decomposed into a regular patch grid. (b) Patches from a large library \mathcal{L} are used to approximate each patch from the test image. (c) The choice of library patch provides information about the true pose. (d) The model parameters \mathbf{W} are used to interpret these patch choices in a Bayesian framework to calculate a posterior over pose (e).

or *inference* and we will describe each in turn.

First, let us look at *generation* from this model. For example, consider the generative process for the top-left patch of a test image. The true pose induces a probability distribution over all the patches in the library based on the learned parameters \mathbf{W} . We choose a particular patch using this probability distribution and add independent Gaussian noise at each pixel to create the observed data. In inference we invert this generative process using Bayes' rule to establish which pose was most likely to be responsible for the observed data.

Alternatively in *inference* (see figure 4.2), the test image patch is approximated by a patch from the library \mathcal{L} . The particular library patch chosen can be thought of as having a different affinity with each pose. These affinities are learned during a training period and are embodied in a set of parameters \mathbf{W} . The relative affinity of the chosen library patch for each pose is used to determine a posterior probability over pose. We will now consider the inference process in more detail.

4.3.1 Inference

Consider the task of estimating a continuous value for the pose parameter β for a test image, where β can have values ranging from -90° to 90° . The test image \mathbf{Y} is represented as a non-overlapping grid of patches $\mathbf{Y} = [\mathbf{y}_1 \dots \mathbf{y}_P]$. The model will be trained from I training examples \mathbf{X} with labeled poses. Each training example is also represented as a non-overlapping grid of patches of the same size as the test data. We denote the p^{th} patch from the i^{th} training example by \mathbf{x}_{ip} (see figure 4.3a).

We also have a library \mathcal{L} of images that are not in the training or test set and would normally

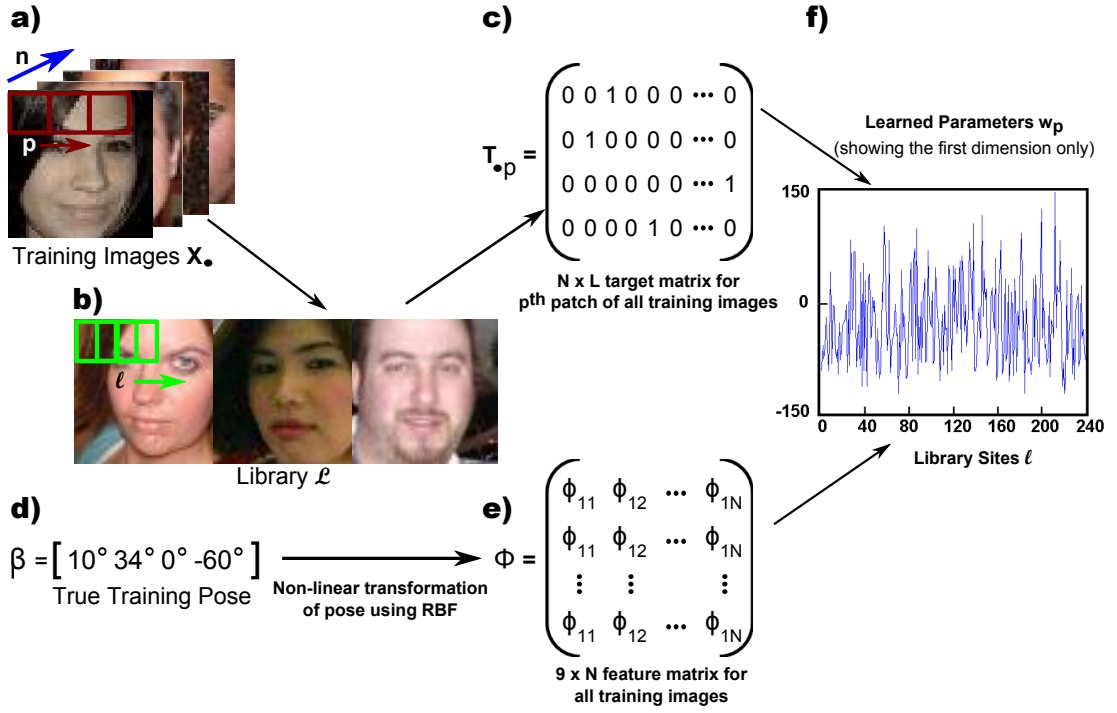


Figure 4.3: Training (a) The model is trained from I training images, each of which is represented as a non-overlapping grid of patches indexed by p . (b) The library \mathcal{L} is considered as a collection of patches \mathcal{L}_l where l indexes the N possible sites. (c) We use 1-of- N coding scheme to represent the closest library patch \hat{l} and store them in a target matrix T_p for the p^{th} patch of all training images. (d) The true poses are stored in a $I \times 1$ vector β . (e) Columns of Φ represent ϕ_i : 9D radial basis functions of pose parameter β for the i^{th} training image. (f) We learn the parameter vector w_{p1} which represents the tendency for the library site l to be picked when considering patch p of an image with pose vector ϕ_i .

contain examples with various poses. We will consider the library as a collection of patches \mathcal{L}_l where $l \in \{1..N\}$ indexes the N possible sites from which we can take library patches (see figure 4.3b). These patches are the same size as those in the test and training images but may be taken from anywhere in the library (i.e. they are not constrained to come from a non-overlapping grid).

The output of our algorithm is a posterior probability over the pose parameter β . We calculate this using Bayes' rule

$$Pr(\beta|\mathbf{Y}, \mathbf{W}) = \frac{\prod_{p=1}^P Pr(\mathbf{y}_p, l^*|\beta, \mathbf{w}_{p\bullet}) Pr(\beta)}{Pr(\mathbf{Y})} \quad (4.1)$$

where we have assumed that the test patches \mathbf{y}_p are independent. The term β represents pose, $w_{p\bullet}$ are the parameters of the model and the variable l^* is the site in the library that most closely matches the test patch \mathbf{y}_p . The notation \bullet indicates all of the values that an index can take, so $w_{p\bullet}$ denotes the parameter vector associated with the p^{th} patch in the test image and all of the sites in the library. To find the site l^* in the library we assume that the test patch is a Gaussian

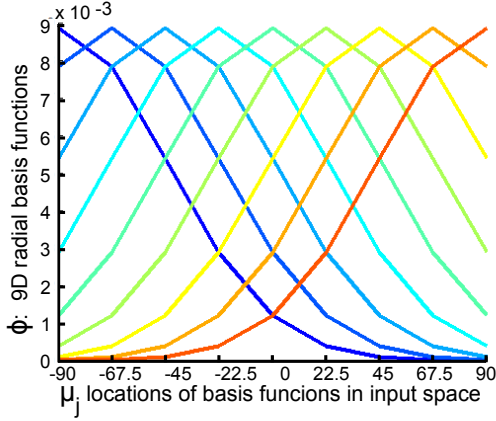


Figure 4.4: Radial Basis Functions: An illustration of the 9D radial basis functions (RBF) $\phi = \{\phi_1.. \phi_J\}$ used as the input to our algorithm for pose estimation. The x axis denotes μ_j which governs the locations of the basis functions in input space and lie within -90° to 90° . A standard deviation of $\sigma = 45$ was used to plot the RBFs.

corruption of the library patch and use

$$l^* = \arg \max_l \mathcal{G}_{\mathbf{y}_p}[\mathcal{L}_l; \sigma^2 \mathbf{I}] \quad (4.2)$$

where \mathcal{L}_l is the patch from site l of the library \mathcal{L} .

We now define the likelihood in Equation 4.1 to be a multinomial distribution on the library sites. The model takes the form of an inverse multi-class logistic regression. We use the standard logistic regression (a discriminative model) in the reverse order making it a generative model that predicts a continuous parameter i.e. the continuous pose vector ϕ from a set of discrete classes which are the library sites. Thus we define the likelihood term $Pr(\mathbf{y}_p, l^* | \beta, \mathbf{w}_{p\bullet})$ in Equation 4.1 as:

$$Pr(\mathbf{y}_p, l^* | \beta, \mathbf{w}_{p\bullet}) = \frac{\exp(\mathbf{w}_{pl^*}^T \phi)}{\sum_{l=1}^N \exp(\mathbf{w}_{pl}^T \phi)} \quad (4.3)$$

where $\phi = \{\phi_1.. \phi_J\}$ is chosen to be 9D radial basis functions (RBF) defined as

$$\phi_j = \exp\left(-\frac{(\beta - \mu_j)^2}{2\sigma^2}\right) \quad (4.4)$$

where $j \in \{1..9\}$ and μ_j govern the locations of the basis functions in input space and lie within -90° to 90° . The term σ denotes the standard deviation and is set during training (see figure 4.3e). To illustrate, we plot the 9D RBF input Φ to our algorithm in figure 4.4, corresponding to 181 poses from -90° to 90° respectively, where we used a standard deviation of $\sigma = 45$. The parameter \mathbf{w}_{pl} in equation 4.3, represents the tendency for the patch from library site l to be picked when considering patch p of an example image with pose vector ϕ . This can be visualized as in figure 4.3f. We find the maximum-a-posteriori estimate of pose by first doing a one dimensional line search of pose varying from -90° to 90° and then maximizing the posterior probability over pose $Pr(\beta | \mathbf{Y}, \mathbf{W})$ in Equation 4.1.

4.3.2 Training

In this section, we consider how to use the training data $\mathbf{x}_{\bullet p}$ from the p^{th} patch of all of the I training images to learn the relevant parameters $\mathbf{w}_{p\bullet} = \{\mathbf{w}_{p1} \dots \mathbf{w}_{pN}\}$ where N is the size of the library. Notice we learn a separate weight vector $\mathbf{w}_{p\bullet}$ for each patch in the input image. This is done by maximizing the likelihood of the observed patches in the training images given their poses with respect to the parameters $\mathbf{w}_{p\bullet}$. This procedure is similar to multi-class logistic regression (See [18] for more details). The likelihood of a single training patch \mathbf{x}_{ip} from the i^{th} training image is defined as:

$$z_{ipl}(\boldsymbol{\phi}) = Pr(\mathbf{x}_p, \hat{l}_i | \beta, \mathbf{w}_{p\bullet}) = \frac{\exp(\mathbf{w}_{pl}^T \boldsymbol{\phi}_i)}{\sum_{l=1}^N \exp(\mathbf{w}_{pl}^T \boldsymbol{\phi}_i)} \quad (4.5)$$

where \hat{l}_i is the library site that most closely matches the training patch \mathbf{x}_{ip} and is defined as

$$\hat{l}_i = \arg \max_l \mathcal{G}_{\mathbf{x}_{ip}}[\mathcal{L}_l; \sigma^2 \mathbf{I}] \quad (4.6)$$

Now we will use 1-of- N coding scheme to represent the selected library site \hat{l}_i for each patch of each training image and store it in a target vector \mathbf{t}_{ip} . Thus, \mathbf{t}_{ip} is the target vector for the p^{th} patch of the i^{th} training example with a feature vector $\boldsymbol{\phi}_i$, and is a binary vector with all elements zero except for \hat{l}_i which equals one. Consider the entire training data, we can now rewrite the likelihood term in Equation 4.5 as follows:

$$\begin{aligned} Pr(T_{\bullet p} | \mathbf{w}_{p1}, \dots, \mathbf{w}_{pN}) &= \prod_{i=1}^I \prod_{l=1}^N Pr(\mathbf{x}_{ip}, \hat{l}_i | \beta, \mathbf{w}_{p\bullet})^{t_{ipl}} \\ &= \prod_{i=1}^I \prod_{l=1}^N z_{ipl}^{t_{ipl}} \end{aligned} \quad (4.7)$$

where $z_{ipl} = z_{pl}(\boldsymbol{\phi}_i)$ and $T_{\bullet p}$ is a $I \times N$ matrix of target vectors (closest library sites) for p^{th} patch and all of the training images with elements t_{ipl} (see figure 4.3c). To find the parameters \mathbf{w}_{pl} we need to maximize Equation 4.7 with respect to these parameters. This is equivalent to minimizing the negative logarithm of Equation 4.7:

$$E(\mathbf{w}_{p1} \dots \mathbf{w}_{pN}) = -\ln Pr(T_{\bullet p} | \mathbf{w}_{p1}, \dots, \mathbf{w}_{pN}) = -\sum_{i=1}^I \sum_{l=1}^N t_{ipl} \ln z_{ipl} \quad (4.8)$$

We now minimize the negative log-multinomial likelihood (Equation 4.8) to determine the parameters \mathbf{w}_{pl} . This is done by taking the gradient of Equation 4.8 with respect to one of the parameter vectors \mathbf{w}_{pl} (for details on the derivatives of the softmax function see [18]). After this procedure we obtain:

$$\nabla_{\mathbf{w}_{pl}} E(\mathbf{w}_{p1} \dots \mathbf{w}_{pN}) = \sum_{i=1}^I (z_{ipl} - t_{ipl}) \boldsymbol{\phi}_i \quad (4.9)$$

where we made use of $\sum_{l=1}^N t_{ipl} = 1$. Because Equation 4.8 depends on all of the parameters $\mathbf{w}_{p1} \dots \mathbf{w}_{pN}$ we concatenate all $\nabla_{\mathbf{w}_{pl}}$ to form the derivative and repeat for every patch. We perform a non-linear optimization to estimate the parameters \mathbf{W} using the BFGS Quasi-Newton method.

4.4 Databases

To verify the performance of our method in both controlled and uncontrolled environments, we test our algorithms on two databases: (i) the CUbiC FacePix(30) database [113] which contains images in controlled environments, and (ii) the UCL database, which is a large database of face images captured in uncontrolled environments, that we collected from internet. We will provide details of these databases in the following sections.

4.4.1 The CUbiC FacePix(30) Database: Images Captured in Controlled Environments

FacePix(30) [113] is a face image database created at the Center for Cognitive Ubiquitous Computing (CUbiC) at Arizona State University, and made available free of charge to the research community. It contains face images of 30 people. We use the pose subset of this database. This set contains 181 color face images that collectively represent a spectrum of pose angles (each image corresponds to a rotational interval of 1 degree). The face images in this set are captured against a solid background and contain very little shadowing. Pose angles vary across a range from +90 degrees to -90 degrees, where +90 degrees represents a left profile view, 0 degrees represents a frontal view, and -90 degrees represents a right profile view. Some examples are shown in figure 4.1a.

We resized these images to a 60x60 template, band-pass filtered them using lower and upper cutoff frequencies of 2.5 and 25 cycles per image respectively. Then we weighted the pixels using a Gaussian function with $\sigma = 0.5$ centered on the image. Each image was normalized to have zero mean and unit standard deviation.

4.4.2 The UCL Database: Images Captured in Uncontrolled Environments

We harvested a large database of images of men and women from the web. These were captured in uncontrolled environments and exhibit a wide variation in illumination, scale, expression and pose as well as partial occlusion and background clutter (see figure 4.1b). Faces were automatically detected using a commercial frontal face detector [94]. The images were subsequently transformed to a 60x60 template using a Euclidean warp. We band-pass filtered the images using lower and upper cutoff frequencies of 2.5 and 25 cycles per image respectively. Then we weighted the pixels using a Gaussian function with $\sigma = 0.5$ centered on the image. Each

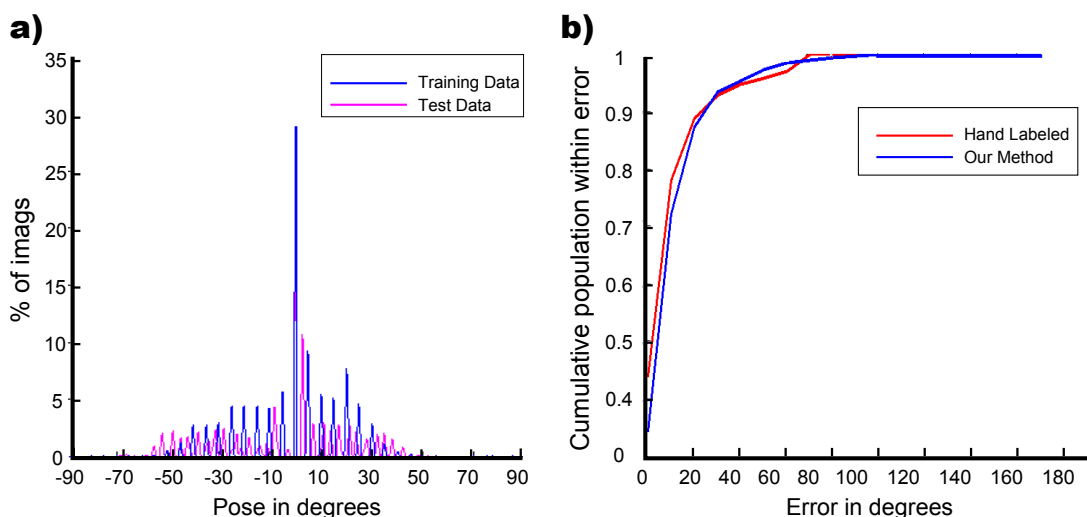


Figure 4.5: Human vs. computer pose estimation error: (a) The distribution of poses in training and test datasets. Note that we do not observe a uniform distribution across pose: we have very few images with large poses. (b) The cumulative population of the data as a function of the pose error in degrees for hand labeled poses (red) and our method (blue). We observe that with our method around 87% of the test data have an error of 20 degrees or less which is equivalent to the error between two subjects in hand labeled poses.

image was normalized to have zero mean and unit standard deviation. Note that since we used a commercial frontal face detector to locate the faces, some of the faces with large pose angles might have been missed hence we do not have an even distribution of poses in our database. Moreover, there is also some bias towards frontal faces in our dataset as most people are looking at the camera. The pose distribution for training and test sets are shown in figure 4.5a. To obtain a human estimate of pose for the above database, four subjects were asked to label the pose in face images ranging from -90° to 90° in 10° steps. The subjects were shown relevant images from the CUbiC FacePix database [113] as a reference and asked to label the images according to their similarity to the pose in the reference image. Due to the large number of images, the database was divided into two subsets. Two different subjects were asked to label each subset of the data separately. The labeled poses of the two subjects were averaged to obtain a continuous estimate of pose for that subset. Some example images with their average labeled pose are shown in figure 4.6. The x-axis represents the average labeled pose which is used as average human estimate. However, the obtained human estimate is still considered noisy since the poses labeled by the two subjects only have a correlation coefficient of 0.76 on the training set and 0.73 on the test set. To measure the noise level in the human pose estimate, we plot the cumulative population of the test data as a function of the mean absolute error (MAE : the absolute error averaged across all test images) between the two subjects. This is shown in figure

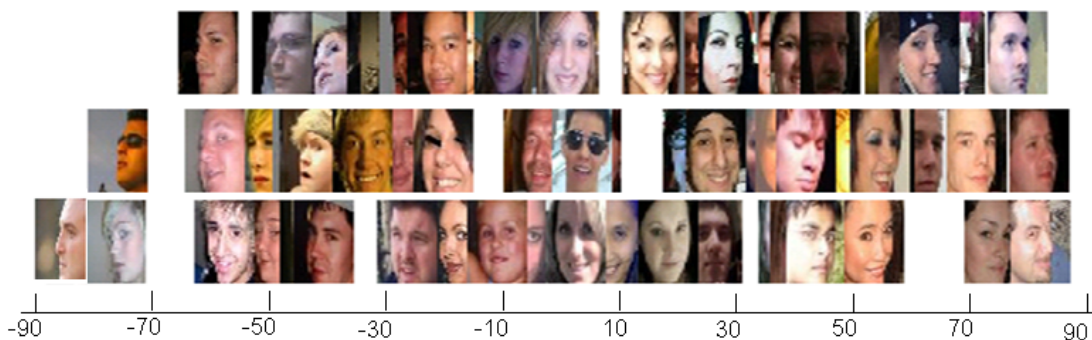


Figure 4.6: Human estimate examples-Some example images along with the average human estimate of pose. The x-axis represents the average labeled pose which is used as the ground-truth.

4.5b, and results in a MAE of 10.89 degrees.

4.5 Experiments on the Constrained Database

We now test our algorithm on automatic pose estimation on the constrained face images of the FacePix [113] database. We divide the pose subset of this database into three non-overlapping sets for training, library and testing such that the same individual will not be used for both training and testing. The training set contains images of 20 individuals which make a total of 3,620 images. The library contains 61 images uniformly sampled from five individuals in 3 degree steps. Finally we test our algorithm on 905 images that come from 5 individuals.


We investigate the effects of two parameters on the performance of our algorithm: (i) the patch grid resolution and (ii) the standard deviation σ of the radial basis functions. We measure the performance using the following metrics: (i) Pearson correlation coefficient (PCC) between the true pose and the estimated pose and (ii) mean absolute error (MAE): which is the absolute error averaged across all test images.

4.5.1 Experiment 1: Varying the Patch-Grid Resolution

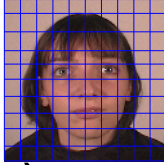
In this experiment we vary the grid resolution from 5×5 to 20×20 by keeping the standard deviation fixed at $\sigma = 11.25$. The results are summarized in Table 4.7a. For illustration, we also plot the patch grid on a sample face in figure 4.7(b-e) for 5×5 , 10×10 , 15×15 and 20×20 grid sizes respectively. The results noticeably improve as we change the patch grid from 5×5 to 10×10 both in PCC and MAE. Then there is a gradual drop in performance when the grid resolution is increased. This might be due to the fact that patches become very small. The performance peaks with a 10×10 grid (6×6 pixel patches). We achieve a correlation coefficient of 0.99 between the true and estimated pose on the test set with a MAE of 5.24° .

Grid Size	PCC	MAE
5 x 5	0.97	7.61
10 x 10	0.99	5.24
15 x 15	0.99	5.77
20 x 20	0.99	5.89

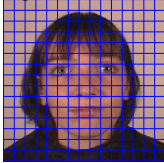
b)



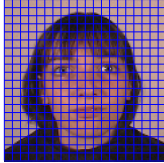
c)



d)



e)



σ	PCC	MAE
5.5	0.98	8.05
11.25	0.99	5.24
22.5	0.99	6.08
45	0.98	8.04
90	0.97	19.22

Figure 4.7: Parameter selection (a) Comparison of performance on pose estimation in various grid resolutions with fixed $\sigma = 11.25$, measured by Pearson correlation coefficient (PCC) and mean absolute error in degrees (MAE) between the true pose and the estimated pose. Visualization of (b) 5×5 (c) 10×10 (d) 15×15 and (e) 20×20 grid sizes. (f) Comparison of performance on pose estimation on various values of σ with patch grid resolution fixed at 10×10 .

4.5.2 Experiment 2: Varying the parameters of the Radial Basis Functions (RBFs)

In this experiment we vary the overlap between the 9D radial basis functions (i.e. the term σ in Equation 4.4), by keeping the patch grid resolution fixed at 10×10 . We test our algorithm with five different values for σ : 5.5, 11.25, 22.5, 45 and 90 degrees. The results of this experiment are summarized in figure 4.7f. We observe a change as the overlap between the radial basis function is varied, reaching a peak with $\sigma = 11.25$ where we achieve a correlation coefficient (PCC) of 0.99 and a MAE of 5.24° . The scatter plot of the results are shown in figure 4.8a. We also plot the average MAE for each degree angle in figure 4.8b. It is interesting to notice the shape of the error histogram in figure 4.8b: two peaks around the large pose degrees ($+/- 30^\circ$ to 70°) and a more flat region around the center. This suggests that, our algorithm performed better at semi frontal images and had larger average error on images that are close to profile. The posterior probability over pose is shown in figure 4.9 for two individual examples. Some example test images are shown in figure 4.10 along with their true (human) and estimated (our algorithm) poses. Despite the fact that our algorithm is designed for very large training data sets, our performance is reasonable compared to the previous results reported on the FacePix database. These include a MAE of 10.41° with Isomap [10], 5.02° with biased Isomap [10], 3.27° with Locally Linear Embedding (LLE) [10], 2.11° with Biased LLE [10], 3.93° with Laplacian Eigenmap (LE) [10] and 1.44° with Biased LE [10]. The reported results use a manifold learning method where the dimension of embedding was 100. Unfortunately the correlation coefficients are not available for the above methods. Note, the protocol in which the

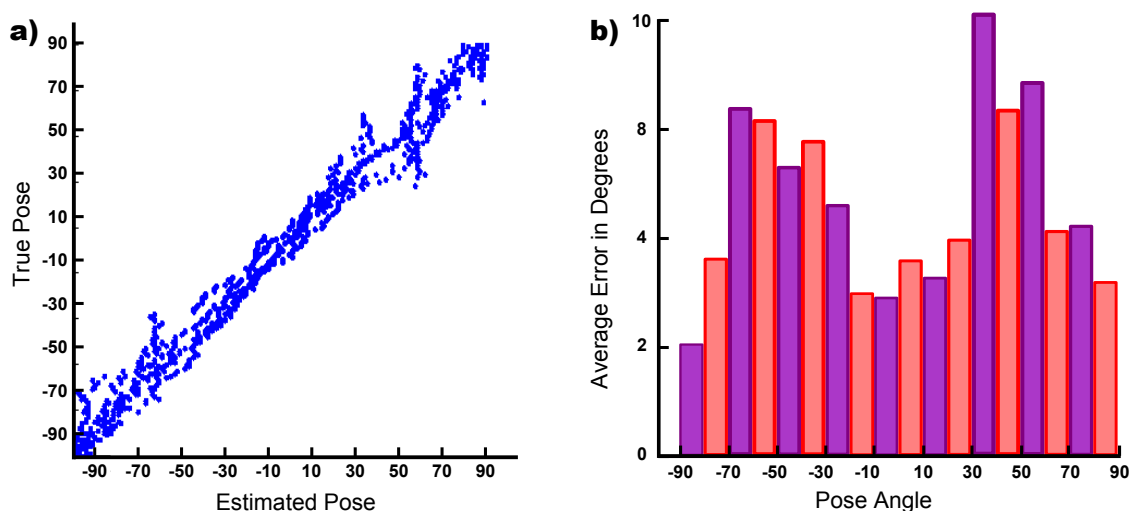


Figure 4.8: Pose estimation results (a) Scatter plot of the results on 10×10 grid with 9D RBF and $\sigma = 11.25$ for the FacePix database. We achieve a PCC of 0.99 and a MAE of 5.24 degrees. (b) Displaying the mean absolute error for each degree pose.

experiments were done are different from ours, in that they used images with increments of 2° , in contrast to our images which are 1° apart.

4.6 Experiments on the Unconstrained Database

We now consider automatic pose estimation of unconstrained face images. We use 10,900 training and 1000 test images. The library is made up 240 images. The training, testing and library sets are randomly selected, are non-overlapping and do not contains the same individuals. The library contains poses varying from -90° to 90° .

As before we investigate the effects of the patch grid resolution and the standard deviation σ on the performance of our algorithm. Similarly, we report the results in terms of the Pearson correlation coefficient (PCC) between the true pose and the estimated pose and the mean absolute error (MAE): which is the absolute error averaged across all test images.

4.6.1 Experiment 3: Varying the Patch-Grid Resolution

In this experiment we vary the grid resolution from 5×5 to 20×20 by keeping the standard deviation fixed at $\sigma = 22.5$. The results are summarized in Table 4.11a. The performance peaks at 10×10 grid (6×6 pixel patches). We achieve a correlation coefficient of 0.74 between the true and estimated pose on the test set with a MAE of 13.75. For visualization purposes we also plot the patch grid on a sample face in figure 4.11(b-e) for 5×5 , 10×10 , 15×15 and 20×20 grid sizes respectively. It can be seen that as the grid resolution becomes higher the patches themselves become very small and perhaps not very informative for the task.

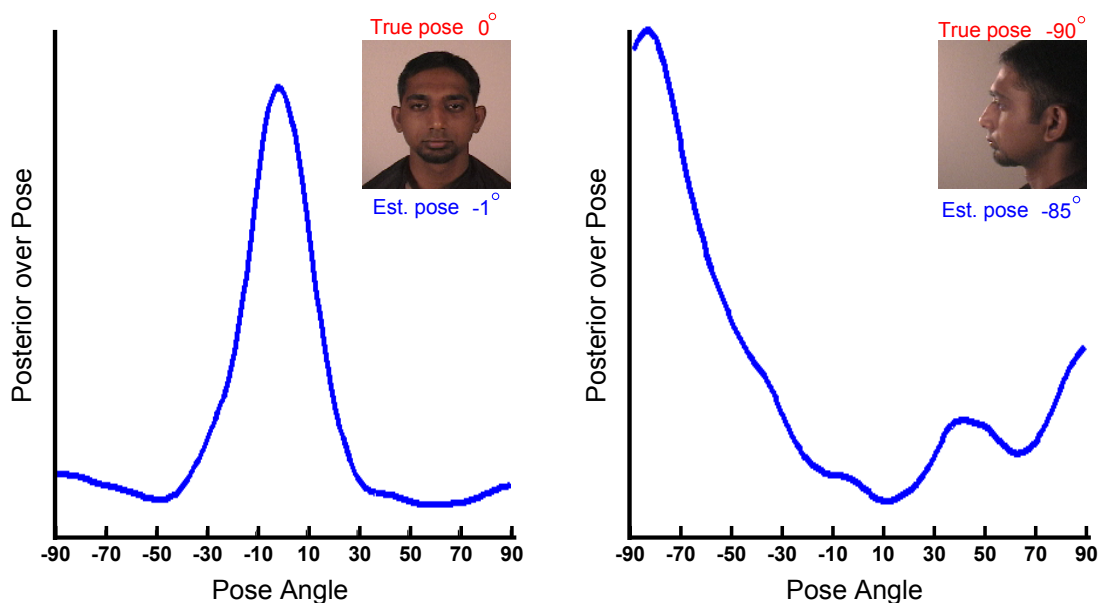


Figure 4.9: *Example posterior over pose: Individual examples with posterior over pose for the images given top right. The true pose and the estimated pose are displayed above and below the image respectively.*

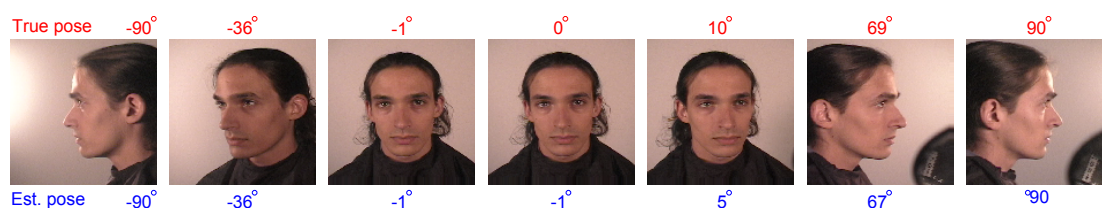


Figure 4.10: *Example Results: Some example images with the correct pose (above) and estimated pose (below).*

4.6.2 Are 6×6 pixel patches sufficient for face pose estimation?

From the previous experiment we conclude that the performance peaks with a 10×10 grid resolution when using 60×60 pixel face images. To further verify that 6×6 pixel patches in the 10×10 grid are sufficient we reconstruct the original images using the closest patches l^* from the library (see figure 4.12). To find the closest library patch l^* in a computationally efficient way, in practice we restrict the possible indices l to a subset corresponding to a 6×6 pixel window around the current test patch position in each library image so patches containing eyes are only approximated by other patches containing eyes etc. Figure 4.12 shows that despite the loss of resolution when using 6×6 patches compared to single pixels, the level of difficulty for a human observer to predict pose from approximated faces is comparable to that of predicting pose from the original images. Therefore, we conclude that 6×6 patches preserve the structure of the original face images which are used to predict pose. We also achieve a very compact

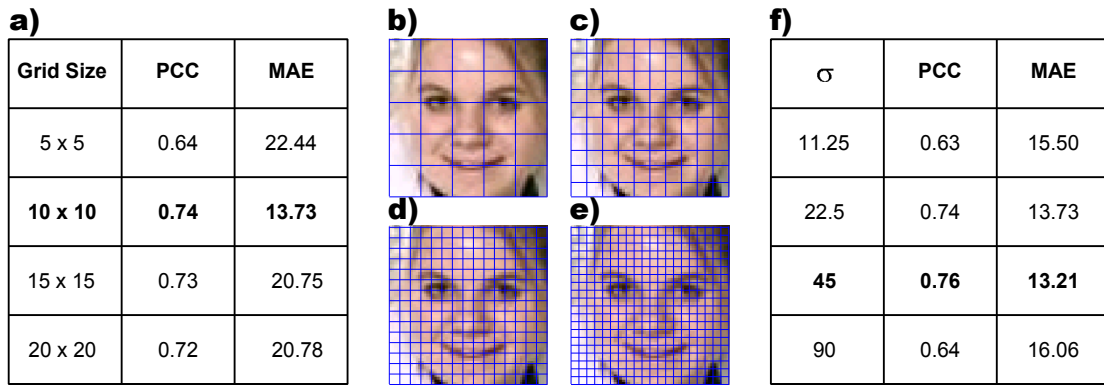


Figure 4.11: Parameter selection (a) Comparison of performance on pose estimation in various grid resolutions with fixed $\sigma = 22.5$, measured by Pearson correlation coefficient (PCC) and mean absolute error in degrees (MAE) between the true pose and the estimated pose. Visualization of (b) 5×5 (c) 10×10 (d) 15×15 and (e) 20×20 grid sizes. (f) Comparison of performance on pose estimation on various values of σ with a 10×10 grid resolution.

representation, by representing the test images as a set of indices to this library. To illustrate this, we compare the amount of memory needed to store the original 60×60 test image and its patch-based representation for a 10×10 grid as a set of indices to a library of 240 images. This experiment shows that we require 10,800 bytes to store the original test image, whereas we only need 400 bytes to store its patch-based representation, resulting in a 27 times more compact representation.

4.6.3 Experiment 4: Varying the parameters of the Radial Basis Functions (RBFs)

In this experiment we vary the overlap between the 9D radial basis function i.e. the term σ in Equation 4.4, by keeping the patch grid resolution fixed at 10×10 . We test our algorithm with four different values for σ : 11.25, 22.5, 45 and 90 degrees. The results of this experiment are summarized in figure 4.11f. The results demonstrate change as the overlap between the radial basis function is varied, reaching a peak with $\sigma = 45$ where we achieve a correlation coefficient of 0.76 on the test data with a MAE of 13.21. The scatter plot of the results on 10×10 grid with $\sigma = 45$ is shown for all of the test data in figure 4.13a. For comparison, we show the scatter plot of the estimated pose for two human subjects on the same test set in figure 4.13b. The PCC and MAE between two human subjects are 0.73 and 10.89 which is comparable to the results our algorithm achieves.

We repeat the pose experiment with 10×10 grid and $\sigma = 45$ for a subset of the test data where the poses are uniformly sampled (i.e. there is the same number of images for each pose). For this subset we achieve a higher PCC of 0.88 and a lower MAE of 11.72 (see figure 4.14a).

a) Original Images**b) Reconstructed Images**

Figure 4.12: Patch-based reconstruction We verify that 6×6 pixel patches are sufficient by reconstructing the original images using the closest patches l^* from the library. It is still easy to identify the pose of the images using the approximated versions. Note: for illustration purposes the RGB form of the test and library images were used in this case to produce the reconstructed images above.

We show the posterior over pose for two individual examples in figure 4.14b.

In figure 4.15a we plot the average error for each pose angle. It is interesting to notice that unlike our results on the constrained database, the shape of the error histogram is mostly flat, meaning the average error was similar for most poses (with the exception of poses between $-67.5 : -90$). We also plotted the proportion of the test data that lies within an error range in degrees (figure 4.15b). We observe that around 87% of the test data has an error of 20 degrees or less. Some example results are shown in figure 4.16, where the correct pose is displayed above the image, and the estimated pose is shown below the image. We also experimented with choosing the expected pose instead of the MAP pose, however the results were inferior.

For comparison, we developed a Gaussian process regression model with an RBF kernel for face pose estimation where the parameters of the RBF kernel were fitted in a maximum likelihood manner. Unfortunately this model cannot handle large number of training images as it involves matrix inversion. Consequently, the model was trained using a subset of the training data used in experiments 3 and 4 and contained 500 images with poses varying from -90 to 90 degrees. This model was tested on the uniformly sampled subset of the test images and achieved a PCC of 0.77 and a MAE of 17.63 degrees. These results are inferior compared to those of our patch-based model on the same test set which scored a PCC of 0.88 an MAE of 11.72 degrees.

4.7 Summary and Discussions

In this chapter, we proposed a probabilistic model for regression and applied it to automatic estimation of head pose. We use a generic patch-based representation that does not rely on

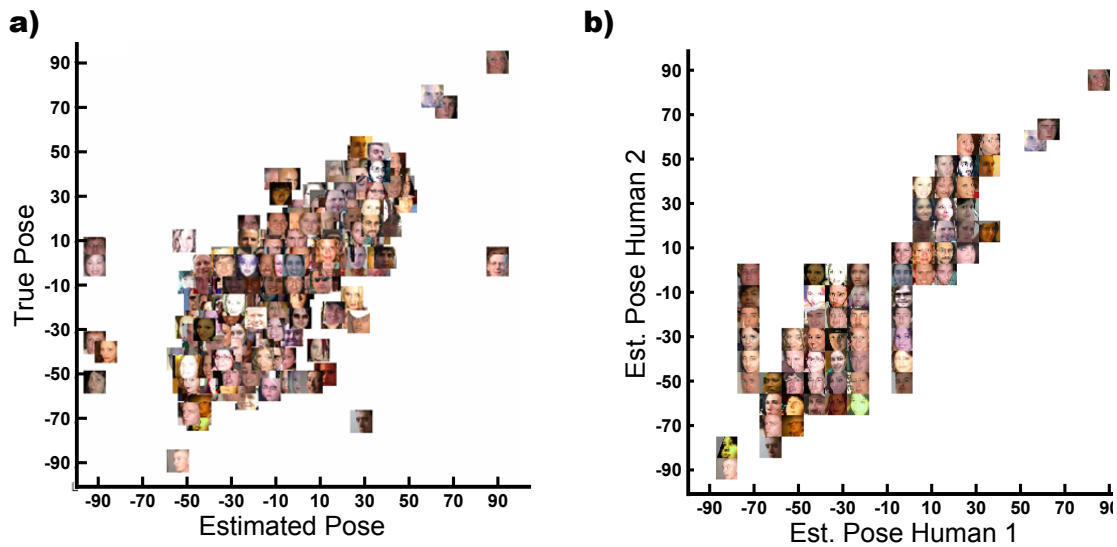


Figure 4.13: Pose estimation results: our algorithm vs human subjects (a) Scatter plot of the results on 10×10 grid with 9D RBF and $\sigma = 45$ for all of the test data. We achieve a PCC of 0.76 and a MAE of 13.21 degrees. (b) Displaying the scatter plot of labeled pose between two human subjects, they achieve a PCC of 0.73 and a MAE of 10.89 degrees.

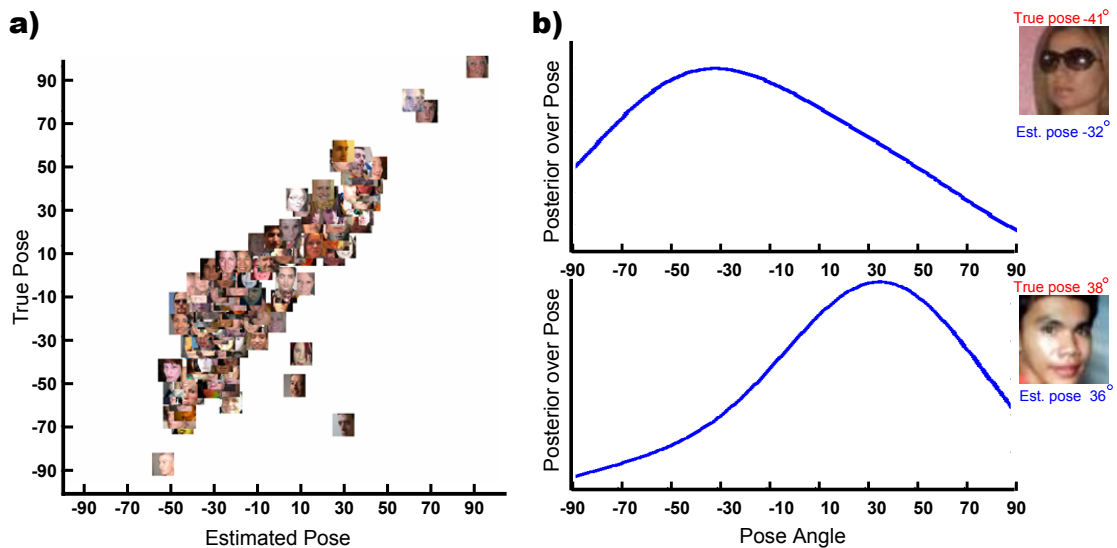


Figure 4.14: Results on unconstrained database by varying σ : (a) Scatter plot of the results on 10×10 grid with 9D RBF and $\sigma = 45$ for a subset of the test data uniformly sampled at each true pose. We achieve a PCC of 0.88 and a MAE of 11.72 degrees. The x-axis and y-axis represent the estimated pose and the true pose respectively. (b) Individual examples with posterior over pose for the images given top right. The true pose and the estimated pose are displayed above and below the image respectively.

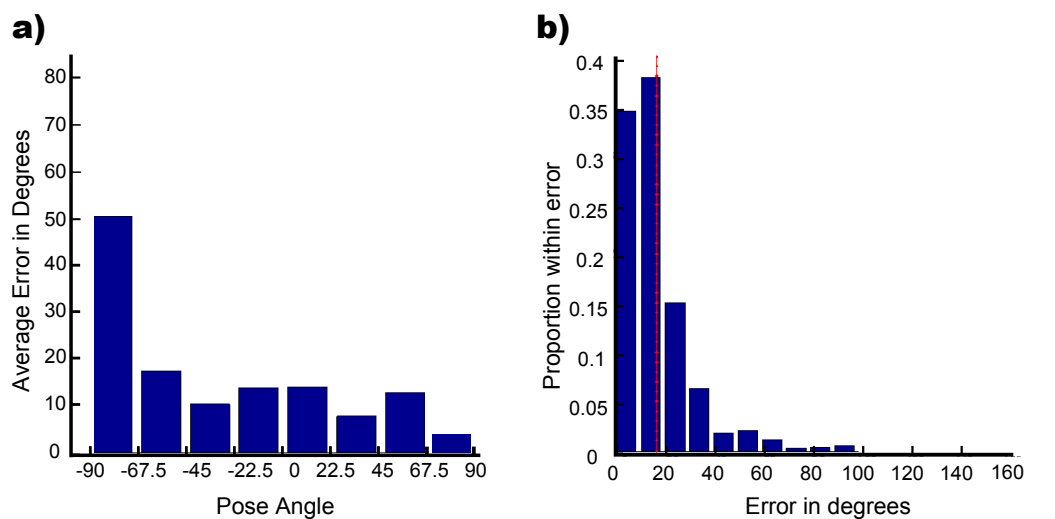


Figure 4.15: Average Error for Each Pose Angle: (a) The average error per pose angle. (b) We plot the proportion of the test data that lies within an error range in degrees. Notice that 78% of the data has an error of 20 degrees or less, shown by the red dotted line.

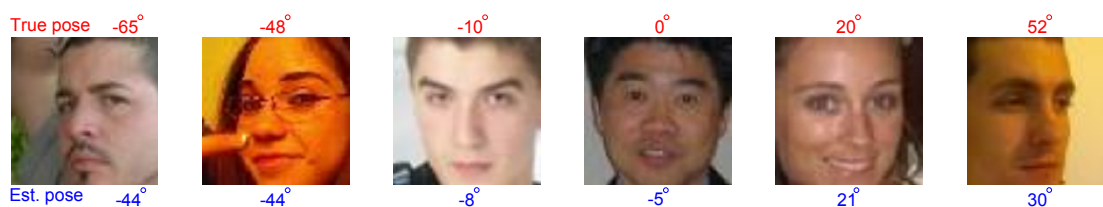


Figure 4.16: Individual image results Example images with true pose (above) and the estimated pose (below).

object-specific landmarks. This allows our model to be used for regression problems on other object classes without major alterations.

The contributions of this chapter are threefold:

1. We developed a novel method for regression type problems. We have shown that a patch-based model can be sufficiently used to represent objects, to allow making inferences about continuous parameters. As a result, we have marked another successful application area for patch-based representation, which helps progress towards our main research objective.
2. We have used the new patch-based model to represent faces taken outdoors and in uncontrolled environments. Our model achieved promising results on face pose estimation, despite variations in scale, illumination, expression, occlusion and background clutter. This suggest that patch-based representation is a robust representation.
3. We have provided a very large database of face images (11,000) in uncontrolled environments and provided human estimates of face pose as ground truth. As such, we have contributed towards testing the ability of patch-based representation in handling large databases.

We have applied our model on automatic face pose estimation, on both constrained and unconstrained databases. We achieve a 0.99 correlation (PCC) between the true pose and the estimated pose on the constrained database with a mean absolute error (MAE) of 5.24 degrees. This is comparable to current best methods on the same database, which are based on manifold learning, and achieve a mean absolute error (MAE) of around 2 to 12 degrees. Note that despite the high performance of these methods on such constrained databases, it is not clear how these methods will generalize on unconstrained databases such as the UCL database. Conversely, our pose estimation algorithm achieves a PCC of 0.88 and MAE of 11.72 on this challenging database which contains images taken in uncontrolled environments. Remarkably, the correlation coefficient of our algorithm to average human performance is about the same as the correlation coefficient between two human subjects.

The model developed in this chapter is closely related to the multi-class logistic regression model. In fact our model takes the form of an inverse multi-class logistic regression where we use the standard logistic regression (a discriminative model) in the reverse order making it a generative model that predicts a continuous parameter here the continuous pose vector, from a set of discrete classes which are the library sites.

Our regression algorithm has a close relationship with nonparametric synthesis algorithms such as image quilting [46] where patches from one image are used to model others. Our algorithm works on similar principles - all the knowledge about the object class is embedded in the library images. This accounts for why the algorithm works so well in different circumstances. If we have enough library images they naturally provide enough information to discriminate the classes.

Our model also has connections to ‘bag-of-words’ type models. The patches in the library can be thought of as a structured set of visual words. There is a major difference however: unlike the visual words in the bag-of-words models, our library patches implicitly encode spatial information about the position of each patch. The success of our algorithm, suggests that the bag-of-words models could potentially be used for pose estimation.

Our algorithm is designed to handle large amounts of data. In terms of scalability it is linear with respect to the size of the library and the dimension of the RBF function. For a library of size m and a n dimensional RBF function, for each patch the processing time scales as $O(mn)$ in training and $O(n)$ in testing. This is considerably faster than manifold learning algorithms such as ISOMAP and LLE which scale as $O(m^2n)$ where m is the number of training samples and n is the dimension of each data point. We also showed that the patch-based representation used in this chapter, where a test image is represented as a list of indices to a library, results in a 27 times more compact representation than storing the original 60×60 test images.

Unfortunately, our current database has limited number of images in large poses (e.g. over 65°) and contains very few close to profile images. One way to overcome this, would be to extract face images from movies. Moreover, currently we use average human estimate as the ground-truth for the unconstrained database. This suggests that our performance is limited by the fidelity of the original labelling. The accuracy of the ground-truth could be increased by using landmarks along with other regression mechanisms such as support vector regression similar to [119] to predict the true pose.

The proposed algorithm is suited to general regression type problems. However, its potential has only been tested on face pose estimation. One interesting application would be estimation of human age. Unfortunately, this was not possible at the time as we did not have access to a large database with age ground-truth. In the future, we would also like to investigate other regression problems such as human body pose estimation using the HumanEva database [1], and predicting GPS coordinates from images using the IM2GPS database [74].

Chapter 5

Patch-based Semantic Segmentation

5.1 Introduction

The success of our proposed patch-based representation models in classification and regression tasks encourages us to investigate this representation for a more challenging vision task: semantic segmentation. Successful completion of this task will contribute towards our main research objective of finding a general representation model that can be used across a variety of object-related tasks. In this chapter we will present a framework where the patch-based representation is used to segment an object into its constituent parts.

Semantic segmentation is the task of labelling each pixel in the image as one of several predefined classes. For example, given an outdoor scene we would like to label the pixels as either sky, road or building etc. One can also consider a more detailed semantic segmentation where we are interested in segmenting parts of an object. For example, given an image of a face we would like to label each pixel as either nose, mouth or eyes etc (see figure 5.1).

In this chapter we will develop a new prior which we call the “shiftmap prior”, and will apply it to semantic segmentation of objects. The proposed prior will exploit the structure of the objects to provide contextual information. We take a generative approach to this problem. Our method takes an intensity image as input and returns a label image (or *labelmap*), where each pixel contains the estimated class label for the corresponding pixel in the input image.

The core idea behind our model is that the final *estimated* labelmap should be similar to a *library* labelmap which is an example labelmap that we have observed before. This is motivated by the fact that there is only a limited variability in the structure of objects like faces (see figure 5.2a). We encourage the estimated labelmap to look like the library labelmap (figure 5.2b). However, there may not be a library labelmap that is sufficiently similar. Therefore, we allow small perturbations to the pixels in the library labelmap to make it agree with the observed image. These perturbations are learned in the form of a “shiftmap” which are a set of relative

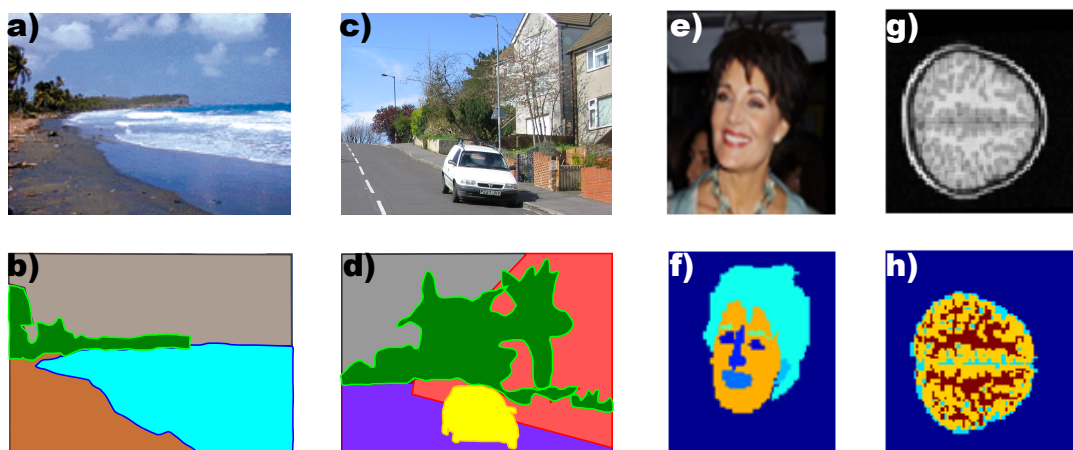


Figure 5.1: Semantic segmentation *Semantic segmentation is the task of labelling each pixel in the given image (top row) as one of several predefined classes and producing a corresponding labelmap (bottom row). (a) For example, given a beach scene, (b) label the pixels as sea, sky, sand and tree. (c) Given an outdoor scene, (d) the goal is to label the pixels as either sky, road, car or building etc. One can also consider semantic segmentation of parts of an object. (e) For example, given an image of a face, (f) the goal is to label each pixel as either nose, mouth or eyes etc. (g) Similarly given an image of a human brain, (h) one would like to label each pixel as white matter, gray matter or CSF.*

shifts that warp the existing labelmap (figure 5.2c). The *warped* labelmap then influences the final estimated labelmap.

This chapter is structured as follows: we will discuss existing methods in section 5.2. We will introduce the *shiftmap prior* for semantic segmentation in section 5.3. In section 5.4, we will describe how to use the shiftmap prior for semantic segmentation of objects. We will show how to optimize the class posterior probabilities in section 5.5. Databases, experiments and results will follow in sections 5.6-5.8. Finally we will summarize in section 5.9.

5.2 Motivation

Semantic segmentation of images remains a difficult problem due to the need to combine local and contextual information when labelling the image. Semantic segmentation of a scene in particular is very difficult because of its relatively unstructured nature. Semantic segmentation of objects on the other hand is more practical since one can exploit the structure of an object class to learn contextual information. For example, when segmenting face images one can make use of the fact that eyes always appear above the nose, or when segmenting images of houses one can assume the roof always appears to be above the walls. Semantic segmentation of objects has widespread applications: segmenting parts of the face can be used as a preprocessing step for

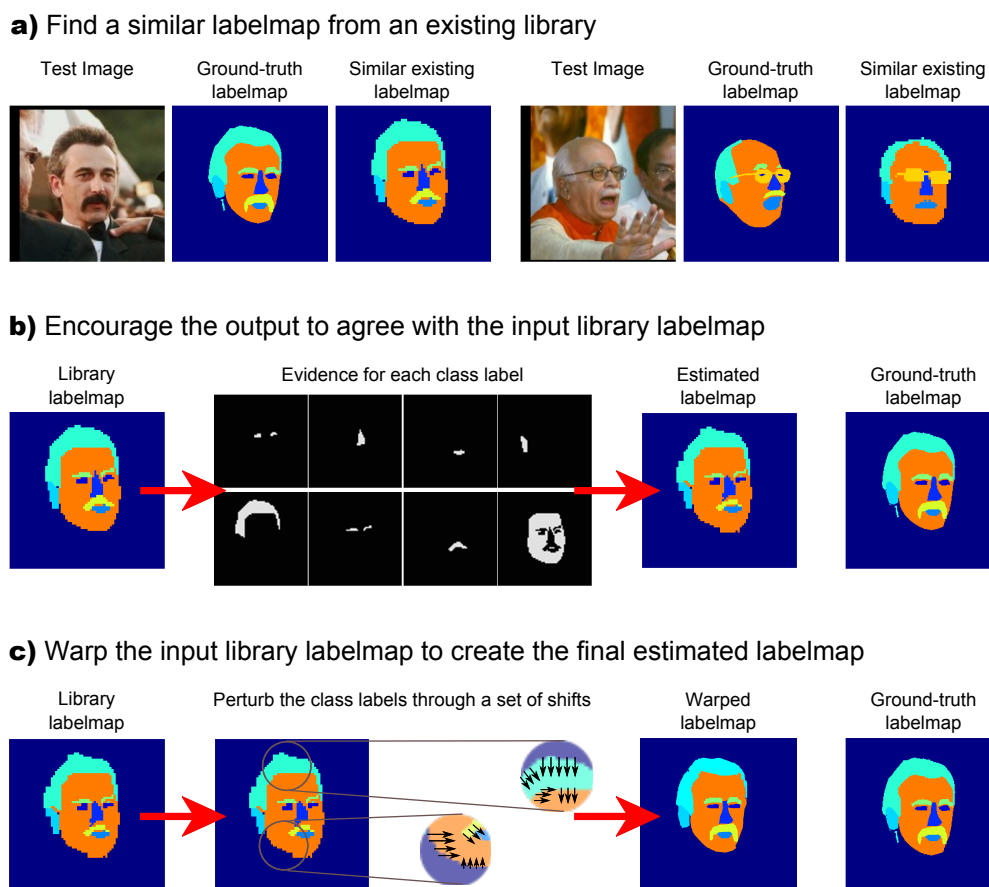


Figure 5.2: Shiftmap prior overview (a) We assume that the estimated labelmap should be similar to an existing library labelmap. Here are two such examples. (b) We encourage the estimated labelmap to look like the library labelmap by assigning high probabilities at each pixel to those classes that agree with the library labelmap. (c) We allow small shifts over the class labels in the library labelmap in the form of a “shiftmap” and show that this improves the segmentation.

face recognition and semantic segmentation of facades and building is beneficial to architects and real estate agents.

A typical approach to semantic segmentation of images is to classify regions of consistent labels using a unary classifier. The term region may denote an image pixel, a block of a regular grid, an irregular patch in the image or an object itself [99]. This is then combined with a prior over label configurations using Markov random fields (MRFs) [56] or conditional random fields (CRFs) framework [99, 157]. Unfortunately these models only capture the object interactions locally and are not capable of representing long range interactions. Hence, they cannot exploit global context.

These undirected models have been extended to capture label relationships at different scales (e.g. Hierarchical MRFs [101]). However, they have many parameters to estimate. Other

extensions include Multiscale Conditional Random fields [77] which incorporate potentials over several different clique sizes using global and local label features. However, this model requires sampling both for parameter learning and label inference and is therefore limited in the size of dataset and number of classes which can be handled efficiently.

Other approaches use directed models such as the Markov structured prior of Domke et al. [41] or the tree structured prior of Feng et al. [56] which have computational advantages over the undirected models. Unfortunately to be able to capture non-stationarity (which is desired for capturing object structure) these models also require a drastic increase in parameters. Note, these models remain essentially unexplored for semantic segmentation of objects such as faces and human brains. To capture non-stationarity, Warrell and Prince [168] proposed a set of priors based on ‘epitomes’ and showed that the performance improves compared to the alternative non-stationary priors.

Another class of segmentation methods is part based models. They use explicit part models to assist object segmentation. For example, Berg et al. [16] learn multinomial distributions over the labels: door, window, roof, etc. in a conditional random field model to segment images of architectural scenes. Despite exploiting the object structure, one drawback of this method is that it needs to have seen all possible combination of features (roof, window etc.) at all possible locations and scales in order to be able to perform detailed semantic segmentation of a building. Unfortunately, this requires tens of thousands of detectors to be trained (one for each possible combination). In addition, there is the laborious task of labelling all of these combinations.

Recently graphcut based energy minimization methods have been very successful in many aspects of computer vision and graphics such as object segmentation [172], image inpainting [75], retargeting [8] and image synthesis [100]. In particular ‘Shiftmap image editing’ [142] uses a new representation for image editing as a graph labelling problem. A shiftmap is the optimal relative shifts for every image pixel required to perform a particular image editing task.

In this chapter we will extend the shiftmap representation for image patches and use it as a prior to perform semantic segmentation of objects. Similar to [142], where they create a new image by re-arranging pieces of an existing image, our method encourages the final segmentation to look like a rearrangement of existing ground truth segmentations. The proposed “shiftmap prior” is locally smooth and globally coherent and can be used for patch-based semantic segmentation of objects in a general probabilistic framework. We formulate the model as an image labelling problem and take advantage of existing efficient training and inference algorithms.

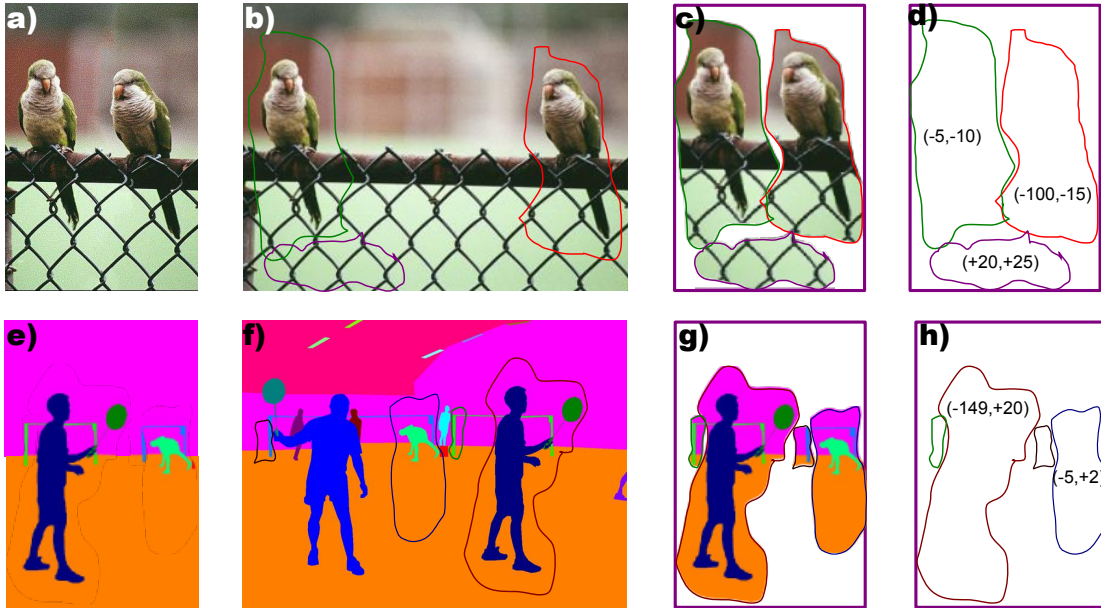


Figure 5.3: Shiftmaps for image retargeting to reduce width. (a) The output image \mathbf{R} is created (b) by copying parts of the original image \mathbf{L} . (c) These parts are carefully chosen to produce a seamless result. (d) The underlying representation is a shiftmap \mathbf{S} which contains a relative shift at each pixel of the new image that specifies the 2d offset to the position in the original image that will be copied from. Figures (e-h) illustrate that shiftmap can be used in a similar manner to edit labelmaps. The input and the output are now labelmaps and the shiftmap defines relative shifts over pixels containing class labels.

5.3 The shiftmap prior: a prior over permutations of class labels

Before presenting our model, we will first describe the shiftmap representation of [142].

The shiftmap proposed by Pritch et al. [142] is a representation used for geometric rearrangements of images which has been used for tasks such as image retargeting and inpainting etc. The shiftmap is defined as *the relative shift for every pixel in the output image from its source in an input image* [142] (see Fig. 5.3).

The relationship between an input image $\mathbf{L}(u, v)$ and an output image $\mathbf{R}(u, v)$ is defined as a shiftmap $\mathbf{S}(u, v) = (t_u, t_v)$. The output pixel is derived from the input pixel as follows:

$$\mathbf{R}(u, v) = \mathbf{L}(u + t_u, v + t_v) \quad (5.1)$$

where the terms t_u and t_v represent the relative shifts in the vertical and horizontal directions respectively, for the pixel in the u^{th} row and the v^{th} column of the output image. We will now present our model, which extends the shiftmap to represent shifts over pixels in a labelmap, and describe how we use it as a prior for semantic segmentation.

In the context of image labelling, we assume that for any given image there exists a similar library labelmap that we have observed before. We can then rearrange the pixels in this library

labelmap to agree with the observed image more closely. These rearrangements are represented as a set of relative shifts i.e. a shiftmap $S(u, v)$, which warps the library labelmap. The *library* and the *warped* labelmaps are equivalent to the input $\mathbf{L}(u, v)$ and the output $\mathbf{R}(u, v)$ in Equation 5.20 respectively. The warped labelmap then influences the final *estimated* labelmap for the given image.

Different rearrangements of input pixels will result in different output labelmaps. In other words different shiftmaps will generate different labelmaps. We define a probability for each set of shifts over the pixels in the input labelmap and call this probability distribution a *shiftmap prior* denoted by $Pr(\mathbf{S})$. We define the *shiftmap prior* as an undirected model which takes the form of a pairwise Markov Random field (MRF):

$$Pr(\mathbf{S}) = \frac{1}{Z} \prod_{uv} \prod_{ij \in \mathcal{N}(uv)} \phi(S(u, v), S(i, j)) \quad (5.2)$$

where (u, v) denotes the pixel in the u^{th} row and the v^{th} column of the image and the term $S(u, v)$ represents the value of the shiftmap for this pixel and the term $\mathcal{N}(uv)$ denotes the neighbouring pixels of this pixel. Our model obeys the Markov property that states that $S(u, v)$ is conditionally independent of all of the other variables given the values of the variables in the neighbourhood $\mathcal{N}(uv)$.

The term $\phi(S(u, v), S(i, j))$ is a potential function and Z is the partition function and normalizes the product of these positive functions so that the total probability $Pr(\mathbf{S})$ is one:

$$Z = \sum_{t_u t_v} \prod_{uv} \prod_{ij \in \mathcal{N}(uv)} \phi(S(u, v), S(i, j)) \quad (5.3)$$

where t_u and t_v represent all possible horizontal and vertical shifts the shiftmap \mathbf{S} can take. We can alternatively write Equation 5.2 as:

$$Pr(\mathbf{S}) = \frac{1}{Z} \prod_{uv} \exp \left[- \sum_{ij \in \mathcal{N}(uv)} \psi(S(u, v), S(i, j)) \right] \quad (5.4)$$

where

$$\psi(S(u, v), S(i, j)) = - \log \phi(S(u, v), S(i, j)). \quad (5.5)$$

The term $\psi(S(u, v), S(i, j))$ is the smoothness term for the relative shifts defined by $S(u, v)$ and $S(i, j)$. This is equivalent to a pairwise cost for neighbouring pixels. As this cost increases, the probability decreases. We define this smoothness term as follows:

$$\psi(S(u, v), S(i, j)) = \begin{cases} 0 & \text{if } S(u, v) = S(i, j) \\ \rho & \text{otherwise} \end{cases} \quad (5.6)$$

where the term ρ is a fixed cost associated with two neighbouring pixels having different shifts. Equation 5.6 implies that there is a zero cost if the pixel on the u^{th} row and the v^{th} column and its neighbouring pixels all have the same relative shift and a fixed cost otherwise. This encourages the pixels to be copied verbatim from the library labelmap.

5.4 Semantic segmentation using the shiftmap prior

Consider the task of assigning a class label $\mathbf{C} = \{C_1 \dots C_I\}$ to a test image where C_i is the class label for the i^{th} pixel in the image and there are K possible classes so $C_i \in \{1 \dots K\}$. The output of our algorithm is a posterior distribution over the class labels \mathbf{C} . According to Bayes rule:

$$Pr(\mathbf{C}|\mathbf{X}) = \frac{\prod_{i=1}^I Pr(x_i|C_i)Pr(\mathbf{C})}{Pr(\mathbf{X})} \quad (5.7)$$

$$\mathbf{C}^{MAP} \propto \arg \max_{\mathbf{C}} \sum_{i=1}^I \log Pr(x_i|C_i) + \log Pr(\mathbf{C}) \quad (5.8)$$

where we have assumed that the observed pixels x_i are independent. The term $Pr(x_i|C_i)$ is the likelihood of the observed image at the i^{th} pixel and the term $Pr(\mathbf{C})$ is the prior over the class labels \mathbf{C} . We perform maximum a posterior (MAP) inference to find the most likely labels for a given image in Equation 5.8.

We are now going to incorporate the idea of the shiftmap as described in section 5.3 into the prior over the class labels $Pr(\mathbf{C})$. This idea is built upon the concept that for any given test image we can re-arrange the pixels in a library labelmap, such that this warped (re-arranged) labelmap influences the final estimated labelmap. For now, the reader should take it for granted that we have a ground truth labelmap that is close to the test image. We will return to the issue of using multiple library labelmaps in section 5.5.3. To find the best possible re-arrangement, we define the prior over the class labels $Pr(\mathbf{C})$ in Equation 5.7 as follows:

$$Pr(\mathbf{C}) = \int_{\mathbf{S}} Pr(\mathbf{C}, \mathbf{S}) \quad (5.9)$$

$$= \int_{\mathbf{S}} \prod_i Pr(C_i|S_i)Pr(\mathbf{S}) \quad (5.10)$$

As Equation 5.10 shows, our prior over the class labels depends on the shiftmap parameters S_i , and that $Pr(\mathbf{S})$ is achieved by marginalizing the joint probability of the class labels and the shiftmap S_i . The term S_i is the ‘‘shiftmap’’ defined in section 5.3 where we have replaced the 2D representation of the pixel with indices (u, v) to a single index i for notation simplicity (we will describe how to find the optimal shiftmap in section 5.5).

The first term $Pr(C_i|S_i)$ is a measure of evidence for label C_i given the value of the

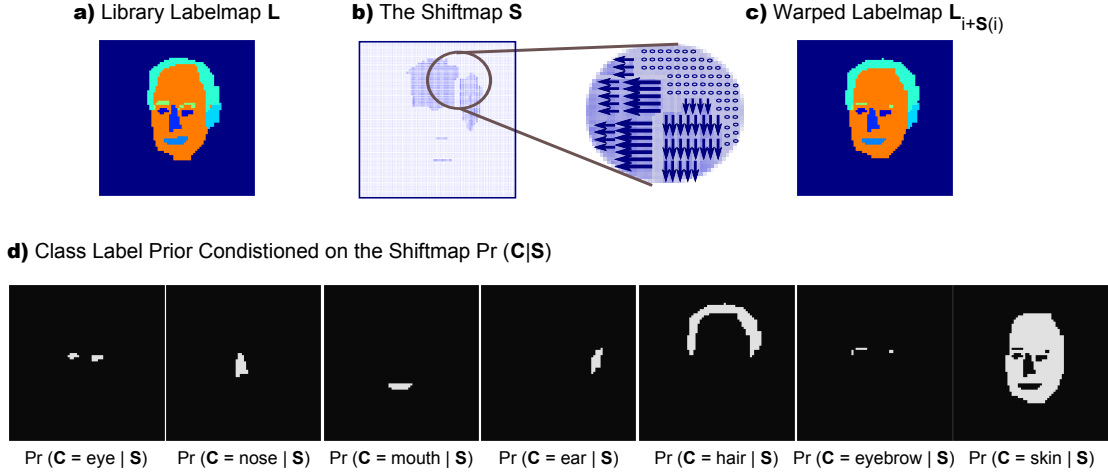


Figure 5.4: Class label priors conditioned on the shiftmap: (a) A library labelmap \mathbf{L} which we believe is similar to the test image. (b,c) A shiftmap \mathbf{S} is used to warp the library labelmap \mathbf{L} such that the warped labelmap is $\mathbf{L}_{i+\mathbf{S}(i)}$, where i indexes the i^{th} pixel. (d) We then use the warped labelmap to determine prior probabilities $\Pr(\mathbf{C}|\mathbf{S})$ for all of the classes conditioned on the shiftmap \mathbf{S} . For example, the term $\Pr(\mathbf{C} = \text{eye}|\mathbf{S})$ takes a high probability at the pixels where the warped labelmap has the class label ‘eye’. Similarly $\Pr(\mathbf{C} = \text{nose}|\mathbf{S})$ takes a high probability at the pixels where the warped labelmap has the class label ‘nose’ etc.

current shiftmap S_i at pixel i and is defined as a multinomial distribution:

$$\Pr(C_i|S_i) = \text{Mult}_{C_i}[\theta[S_i, \mathbf{L}]] \quad (5.11)$$

where \mathbf{L} is the library labelmap (figure 5.4a) and θ represents the tendency for the class label C_i to be picked at the i^{th} pixel when considering the shiftmap S_i (figure 5.4b). The term $\Pr(\mathbf{S})$ is the shiftmap prior defined in Equation 5.2. In principle, one can learn the multinomial distribution in Equation 5.11 from training data. In practice, we define $\Pr(C_i|S_i)$ as follows:

$$\Pr(C_i|S_i) = \begin{cases} \theta_{\text{correct}} & \text{if } \mathbf{L}_{i+\mathbf{S}(i)} = C_i \\ \frac{1-\theta_{\text{correct}}}{K-1} & \text{otherwise} \end{cases} \quad (5.12)$$

where \mathbf{L} is the chosen library labelmap and K is the number of possible class labels. The term θ_{correct} is set to be close to 1 (see sections 5.7.4 and 5.8.4). This implies that probability $\Pr(C_i|S_i)$ becomes high when the pixel in the warped labelmap resulting from the shift S_i has the same value as the current class label C_i . This probability is low otherwise (figure 5.4c,d).

Note that the integration in Equation 5.10 is intractable in practice hence we perform a point estimate of $\Pr(C_i)$ by maximizing the joint probability instead. Therefore

$$\Pr(C) \approx \max_{\mathbf{S}} \prod_i \Pr(C_i|S_i)\Pr(\mathbf{S}) \quad (5.13)$$

5.5 Optimizing the full posterior probability

In this section we show how to perform semantic segmentation of an object by alternately maximizing over the class labels \mathbf{C} and the shiftmap parameters \mathbf{S} . In fact, we consider finding the best class labels \mathbf{C}^* in terms of energy maximization and take advantage of existing efficient algorithms [87]. For this purpose we turn the posterior probability in Equation 5.8 to an energy term E and find the best labels \mathbf{C}^* by maximizing this energy:

$$E = \sum_i \log Pr(x_i|C_i) + \lambda_1 \sum_i \log Pr(C_i|S_i) + \lambda_2 \log Pr(\mathbf{S}) \quad (5.14)$$

$$\mathbf{C}^* = \arg \max_{\mathbf{C}} \left[\max_{\mathbf{S}} \sum_i \log Pr(x_i|C_i) + \lambda_1 \sum_i \log Pr(C_i|S_i) + \lambda_2 \log Pr(\mathbf{S}) \right] \quad (5.15)$$

where the first two terms encourage the estimated labelmap to look like the test data and the shiftmap prior respectively, and the third term is equivalent to a smoothness term which encourages the neighbouring pixels to have the same shift. The terms λ_1 and λ_2 are the weighting parameters for the data term and the shiftmap prior term respectively. These two terms are not part of our original model. However, they improve the performance in practice. One possible reason for this improvement is that because we have assumed independence among the observed pixels (which is not true in general), our solution is scaled. These weighting parameters resolve this issue and therefore improve the final solution. We will empirically set λ_1 and λ_2 in the experimental section.

In our iterative maximization algorithm, we maximize the energy term in Equation 5.15 alternately with respect to two sets of parameters. In step 1, we update the shiftmap \mathbf{S} , which effects the last two terms i.e. the prior probabilities of the classes conditioned on the shiftmap $\log Pr(C_i|S_i)$ and the prior probability of the shiftmap $\log Pr(\mathbf{S})$. Then in step 2, we update the class labels \mathbf{C} , which effects the first two terms i.e. the data likelihood $\log Pr(x_i|C_i)$ and the prior probabilities $\log Pr(C_i|S_i)$. We do this in an iterative manner and show that the energy in Equation 5.14 increases at each iteration. We will now describe this iterative maximization process.

We start with an initial estimate of the class labels \mathbf{C} . This is achieved in a maximum log likelihood manner using the likelihood of the observed data given the class labels i.e. $\sum_i \log Pr(x_i|C_i)$. We call this $\mathbf{C}^{[t]}$, where $[t]$ indicates current iteration. We keep the $\mathbf{C}^{[t]}$ fixed and use it to maximize over the shiftmap \mathbf{S} :

$$\mathbf{S}^* = \arg \max_{\mathbf{S}} \sum_i \log Pr(C_i^{[t]}|S_i) + \log Pr(\mathbf{S}) \quad (5.16)$$

Having found the best shiftmap \mathbf{S}^* , we then keep it fixed as $\mathbf{S}^{[t]}$ and maximize over the class

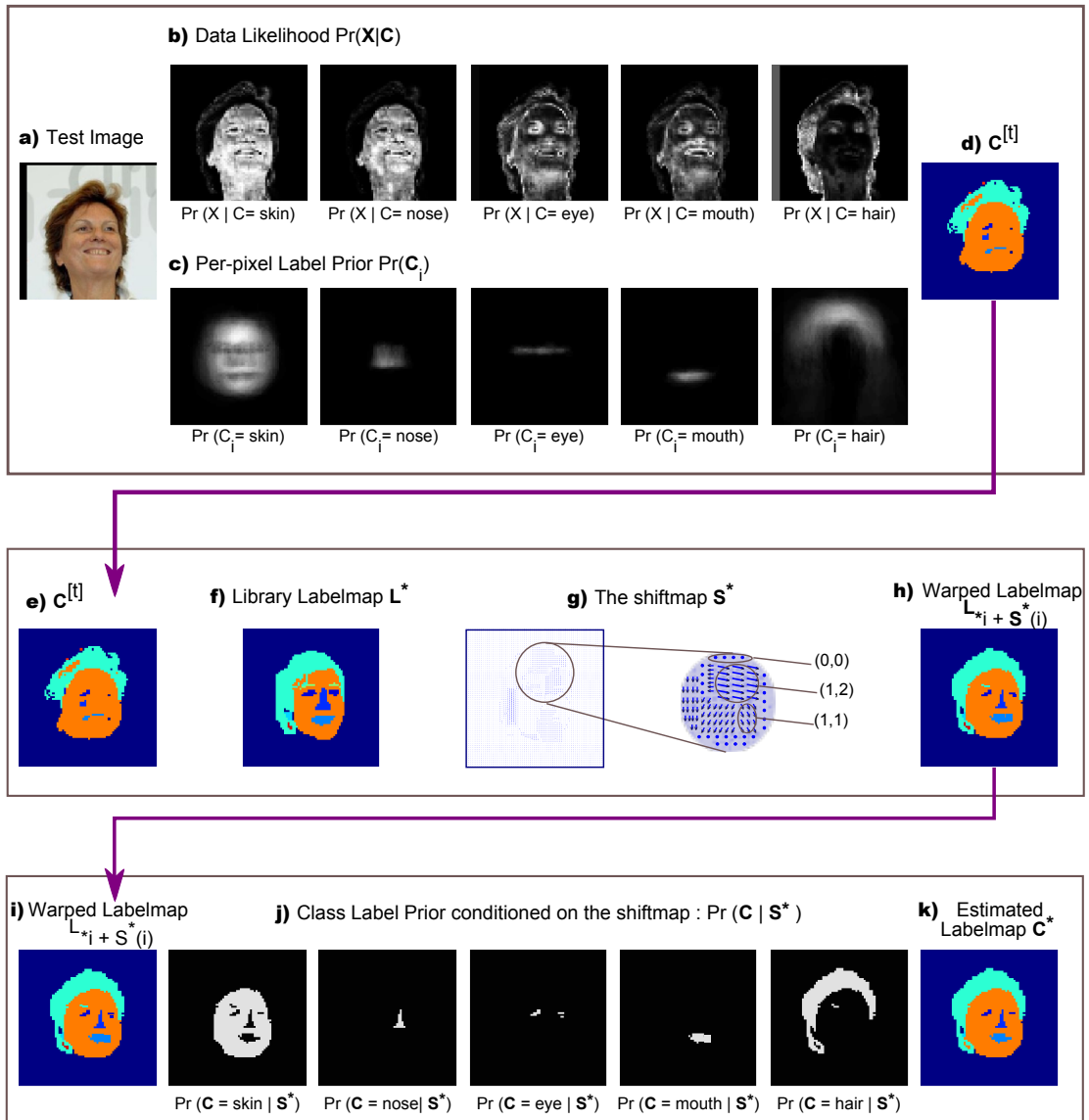


Figure 5.5: The complete semantic segmentation algorithm: (a) Given a test image, (b) find the data likelihood for each of the class labels $\Pr(\mathbf{X}|\mathbf{C})$. (c) Combine this with a set of learned per-pixel class label prior $\Pr(\mathbf{C}_i)$ (d) and find the maximum a posteriori segmentation of class labels $\mathbf{C}^{[t]}$. (e) Given current $\mathbf{C}^{[t]}$ (f) find a similar library labelmap \mathbf{L}^* . (g) Given $\mathbf{C}^{[t]}$ and \mathbf{L}^* , find the best shiftmap \mathbf{S}^* (figure 5.5g) and (h) warp the library labelmap to make it as similar to $\mathbf{C}^{[t]}$ as possible. (i) Use the warped labelmap (j) to update the prior probabilities $\Pr(\mathbf{C}|\mathbf{S}^*)$. (k) Finally, use the data likelihood $\Pr(\mathbf{X}|\mathbf{C})$, the prior probabilities $\Pr(\mathbf{C}|\mathbf{S}^*)$ and the shiftmap prior term $\Pr(\mathbf{S})$ in the energy term defined in Equation 5.14 and maximize it to obtain the final estimate of class labels \mathbf{C}^* .

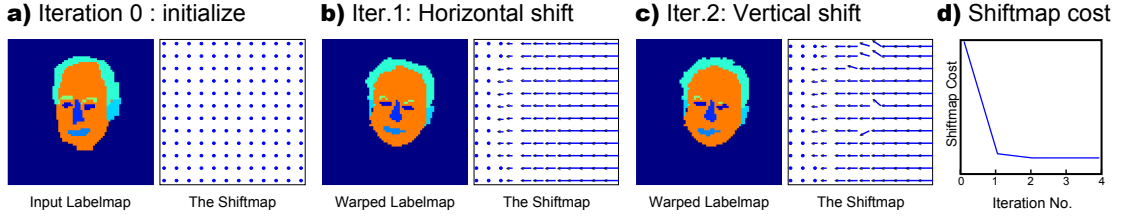


Figure 5.6: We alternately find globally optimal solutions to the vertical and horizontal shifts: (a) Iteration 0: where we initialize the shiftmap with zero shifts. We then alternately (b) Find the best horizontal shifts, keeping the vertical shifts fixed. The labelmap on the left shows the result of this horizontal warp. (c) We keep the horizontal shifts fixed and we find the optimal vertical shifts. The labelmap on the left shows the result of this warp. (d) This graphs shows that the shiftmap cost (energy) decreases at each iteration. We repeat this until the energy no longer decreases.

labels \mathbf{C} :

$$\mathbf{C}^* = \arg \max_{\mathbf{C}} \sum_i \log Pr(x_i | C_i) + \sum_i \log Pr(C_i | S_i^{[t]}) + \log Pr(\mathbf{S}^{[t]}) \quad (5.17)$$

We then go back to Equation 5.16 and use \mathbf{C}^* as $\mathbf{C}^{[t]}$ and repeat the process until convergence. A summary of the complete algorithm is shown in figure 5.5.

5.5.1 Graph Construction

In this section we will describe how to use graphcuts to find the MAP solution for the shiftmap prior $Pr(\mathbf{S})$ which is formulated as an undirected MRF model. For simplicity we will first explain how to build the graph using single pixels. We will then show how to construct the graph for patches in section 5.5.2. We use the graph construction proposed by Moore et al. [128] which is an efficient implementation of a special case of the more general graph construction used in [151, 87]. We will now explain this in detail.

Given a library labelmap, our goal is to find the best set of shifts \mathbf{S} to transform (warp) this library labelmap such that it becomes closer to the ground truth for a given test image. We associate two shifts with each pixel, which represent the horizontal and the vertical shifts corresponding to that pixel. We define the problem as a Markov random field (MRF) model. We then use graphcuts to alternately find globally optimal solutions to the vertical and horizontal shifts (see Figure 5.6). For this purpose we write the model in terms of an energy minimization problem:

$$E = \sum_p D_p(u_p) + \sum_{p,q \in \mathcal{N}} V_{pq}(u_p, u_q) \quad (5.18)$$

where u_p is a shift assigned to pixel p from a finite shift set \mathcal{U} . The first term D_p represents the penalty for pixel p having a shift of u_p . This is equivalent to the prior probability term

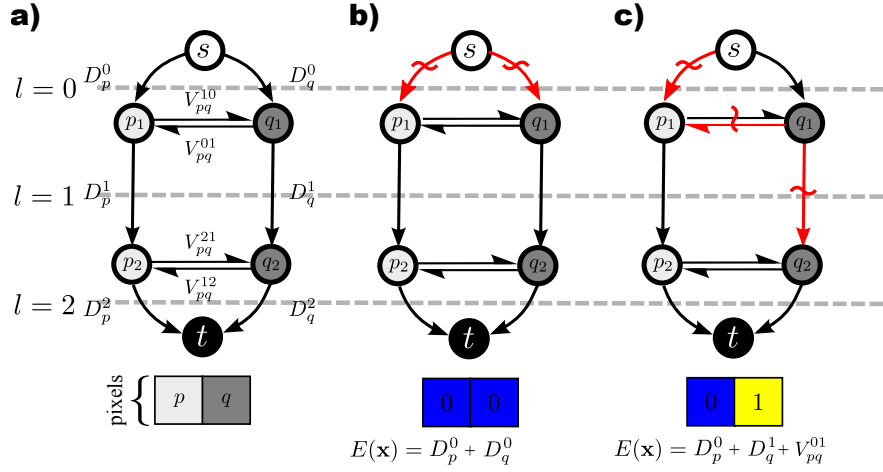


Figure 5.7: Graph construction for shiftmap prior: (a) Example graph construction with two pixels p and q , each of which can take a shift $l \in \{0, 1, 2\}$. Unary costs are associated with vertical links. D_p^k is the unary cost for assigning the k^{th} shift to pixel p . Pairwise costs are represented on horizontal links. V_{pq}^{mn} is the pairwise cost for assigning the m^{th} shift to pixel p and the n^{th} shift to pixel q . The assignment of the shift is determined by which vertical links are cut. (b) Topmost vertical links are cut and so both pixels are assigned shifts of zero. (c) Cuts where the shifts change sequentially also incur a pairwise cost. (images adapted from [128])

$\log Pr(C_i | S_i)$ in Equation 5.14. In particular, we use the maximum likelihood segmented image $\mathbf{C}^{[t]}$, and for each pixel we penalize the shifts that result in an estimated labelmap \mathbf{C}^* where $\mathbf{C}_i^{[t]} \neq \mathbf{C}_i^*$.

The term V_{pq} in Equation 5.18 is the pairwise term which encourages the neighbouring pixels to have the same shift. This is equivalent to the shiftmap prior term $\log Pr(\mathbf{S})$ in Equation 5.14, where there is a cost for the 4-connected neighbours to have different shifts.

We will now describe how to build the graph corresponding to Equation 5.18. For an image with $M \times N$ pixels and $|\mathcal{U}|$ possible shifts, we build a graph with $|\mathcal{U} - 1|$ layers of $M \times N$ nodes (figure 5.7). The topmost layer connects to the source and the bottom most layer connects to the sink. For example, for a $M \times N$ image the possible horizontal and vertical shifts are $\mathcal{U} = \{-(M-1) : +(M-1)\}$ and $\mathcal{U} = \{-(N-1) : +(N-1)\}$ respectively. In practice however, we only use a subset of these shifts by limiting the possible shifts to a subset $\mathcal{K} \subset \mathcal{U}$. We discuss how to set the parameter \mathcal{K} in sections 5.7 and 5.8.

Links between layers in the graph (and the source and sink) have capacities given by the unary terms D_p^k which represent the cost at pixel p for assigning the shift k . Links between neighbouring pixels p and q within layer l have capacities given by pairwise terms $V_{pq}^{l, l-1}$ which denotes the pairwise cost for assigning pixel p to a shift l and pixel q to a shift $l-1$. An example

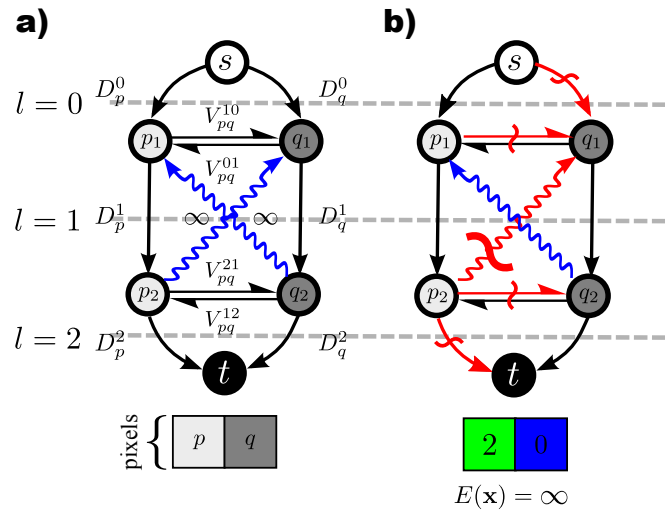


Figure 5.8: Imposing sequential constraints: We impose sequential constraints on the labels. This means that the shifts of neighbouring pixels are not allowed to differ by more than one. This constraint is aimed at avoiding large shifts. (a) Diagonal constraint links with infinite cost prevent non-sequential labels. (b) For pixel p to have a shift of 2 and pixel q to have a shift of zero, we must now also cut one of these diagonal links: the cost for this solution becomes infinite and it will never be chosen (images adapted from [128]).

graph construction with two pixels p and q , is shown in figure 5.7, where each pixel can take a shift $u \in \{0, 1, 2\}$.

When constructing the graph we impose sequential constraints on the labels. This means that the labels of neighbouring pixels are not allowed to differ by more than one. This constraint is aimed at avoiding large shifts. Thus, it preserves the structure of the object in terms of warping. Figure 5.6b-c, show two example shiftmaps where this constraint was applied. The arrows denote the shifts and the length of each arrow represents the length of the shift. Note, that the length of the neighbouring arrows do not differ by more than one, and so they represent a smooth transformation. To force this constraint, Moore et al. [128] introduce a set of diagonal links (see figure 5.8) with infinite capacity between the node at $\{x, y, l\}$ and $\{x, y - 1, l - 1\}$, $\{x, y + 1, l - 1\}$, $\{x - 1, y, l - 1\}$ and $\{x + 1, y, l - 1\}$ to prevent non-sequential labels at neighbouring pixels.

These diagonal constraint links in the current graph construction, effectively allow the shifts to have a difference of one per pixel. For example, for pixel p to have a shift of 2 and pixel q to have a shift of zero, we must now also cut one of these diagonal links: the cost for this solution becomes infinite and it will never be chosen. To find the final solution, we add the

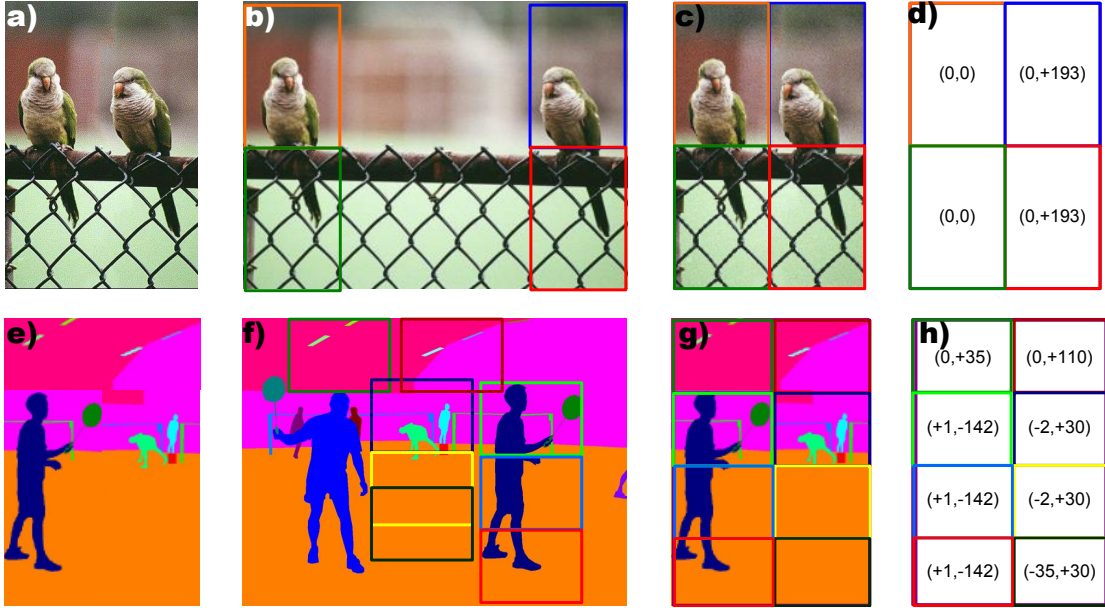


Figure 5.9: Patch-based shiftmaps for image retargeting to reduce width. (a) The output image \mathbf{R} is created (b) by copying patches of the original image \mathbf{L} . Note these patches are of fixed size but can come from anywhere in the input image (i.e. they can be overlapping). (c) The output image is composed of a grid of non-overlapping patches. (d) The underlying representation is a shiftmap \mathbf{S} which contains a relative shift at each patch position in the output image that specifies the 2d offset to the position in the original image that will be copied from. For example $\mathbf{S}(1, 2) = (0, 193)$ implies that the patch in the row 1 and column 2 of the grid comes from the patch at position $((1-1)*\nu+1+0, (2-1)*\nu+1+193)$ in the input image, where ν denotes the patch size. Figures (e-h) illustrate that shiftmap can be used in a similar manner to edit labelmaps. The input and the output are now labelmaps and the shiftmap defines relative shifts over patches containing class labels.

diagonal constraint term to the cost function in Equation 5.18 as follows:

$$E(\mathbf{x}) = \sum_{p \in \mathcal{P}} D_p(u_p) + \sum_{p, q \in \mathcal{N}} V_{pq}(u_p, u_q) + \sum_{p, q \in \mathcal{M}} W_{pq}(u_p, u_q) \quad (5.19)$$

where W represents the diagonal interaction terms that restricts the set of possible solutions on the graph over a set of labels \mathcal{U} with neighbourhood system \mathcal{M} . This cost function is then minimized exactly in polynomial time by finding the minimum cut in the constructed graph.

5.5.2 Graph construction for patches

We will now describe the graph construction for patches. The main difference is that, previously there were as many nodes in the graph as there were pixels in the given image. When using patches however, we only need as many nodes as there are patches in the image. The diagonal links are also adapted to account for different patch sizes. We will first recall the shiftmap concept for patches and then describe how to build the corresponding graph.

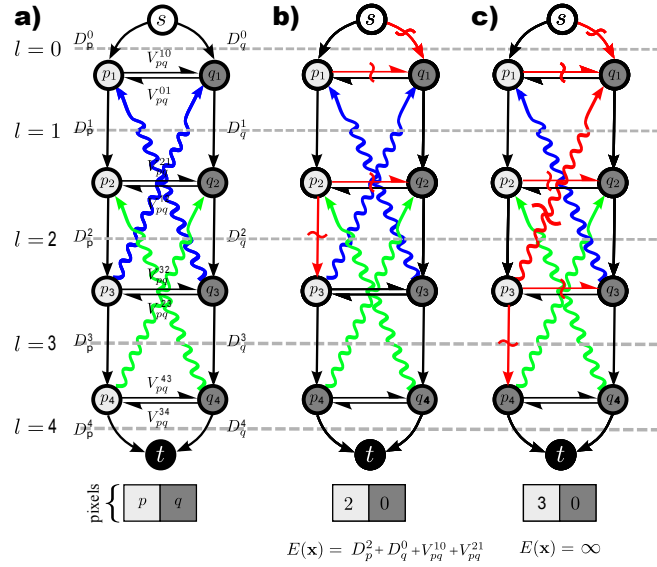


Figure 5.10: Graph construction for the patch-based model (a) To adapt the sequential constraint for the patch-based model we introduce new diagonal constraint links to all nodes in the layer that is n nodes above, where n is set to be the size of the patch. (b) An example graph for 2×2 patches, where the diagonal links are added to all nodes in the layer that is two nodes above. Thus we allow a maximum difference of two in the shifts for neighbouring pixels. (c) As such, when we try to have a shift difference of greater than 2, e.g. for pixel p to have a shift of 3 and pixel q to have a shift of zero, we must now cut one of these diagonal links: the cost for this solution becomes infinite and it will never be chosen (this is similar to the case in figure 5.8b). The graph can be similarly constructed for larger patch sizes.

The shiftmap concept as described in section 5.3 is easily extendable to define relative shifts over patches of an image/labelmap, instead of pixels. The output labelmap is created by a grid of non-overlapping patches. Note, a patch could contain more than one class label. A shiftmap \mathbf{S} is then defined such that, the relative shift $S(u, v)$, now represents a shift for the patch at the u^{th} row and the v^{th} column of a given patch grid. The output patch is derived from the input image as follows:

$$\mathbf{R}(u, v) = \mathbf{L}((u - 1) * \nu + 1 + t_u, (v - 1) * \nu + 1 + t_v) \quad (5.20)$$

where ν denotes the patch size. For example, consider the output labelmap being composed of a 2×2 grid of patches, where patches are 3×3 pixels. A shiftmap of $S(\mathbf{1}, \mathbf{2}) = (0, 193)$ means that the patch in the first row and the second column would come from a patch at position $((1 - 1) * 3 + 1 + 0, (2 - 1) * 3 + 1 + 193)$ i.e. the position (1, 197) in the input image. Figure 5.9 illustrates how we use shiftmap with patches. We will make use of this when dealing with patches in our experimental section.

We will now describe how to build the graph corresponding to Equation 5.18 for patches.

Consider an image composed of an $P \times Q$ grid of patches and $|\mathcal{U}|$ possible labels, we build a graph with $|\mathcal{U} - 1|$ layers of $P \times Q$ nodes. As before, the topmost layer connects to the source and the bottom most layer connects to the sink. Since the labels represent shifts in our case, all possible labels correspond to all possible shifts. For example for a $M \times N$ image the possible horizontal and vertical shifts are $\mathcal{U} = \{-(M - \nu) : +(M - \nu)\}$ and $\mathcal{U} = \{-(N - \nu) : +(N - \nu)\}$ respectively, where ν represents the patch size. In practice however, we only use a subset of these shifts by limiting the possible labels to a subset $\mathcal{K} \subset \mathcal{U}$. We discuss how to set the parameter \mathcal{K} in sections 5.7 and 5.8. The unary and data terms remain as before, we only change the diagonal constraints which we will now describe.

The diagonal constraint links described in section 5.5.1 effectively allowed the shifts to differ by one per pixel. For example, for pixel p to have a shift of 2 and pixel q to have a shift of zero, we had to cut one of these diagonal links: the cost for the solution would then become infinite and it would never be chosen. This is no longer sufficient when we use a patch-based model. This is because the patches are of size $\nu \times \nu$ where $\nu \geq 2$. This means that the current model allows one shift difference per every two or more pixels. To resolve this, we introduce new diagonal constraint links to all nodes in the n layers above, where n is set to be the size of the patch. Figure 5.10 shows the graph construction corresponding to 2×2 pixel patches, where we allow a maximum difference of 2 labels for each neighbouring nodes. Therefore, when we try to have a shift difference of greater than 2, e.g. for pixel p to have a shift of 3 and pixel q to have a shift of zero, we must now cut one of these diagonal links: the cost for this solution becomes infinite and will never be chosen. These diagonal links allow a more smooth warping of a library labelmap using patches. The graph can be similarly constructed for larger patch sizes.

5.5.3 Using multiple library labelmaps

So far, we have assumed that there is a single library labelmap that is similar to the ground-truth test labelmap. We will now drop this assumption and describe how to use multiple library labelmaps.

Previously, we chose a single library labelmap which we used to find the best shiftmap \mathbf{S} . Instead, we are now going to find a shiftmap \mathbf{S} for each of the labelmaps in the library \mathcal{L} . We will then choose the library labelmap \mathbf{L}^* that results in a shiftmap with the minimum cost.

If the size of the library is very large, the above process may not be efficient. Therefore, in practice we first select a subset of the library that are most similar to the current estimate of the class labels $\mathbf{C}^{[t]}$. To choose this subset, we use the log data likelihood $\log Pr(\mathbf{X}|\mathbf{C}_i)$ summed across all classes to rank the library labelmaps and choose the top k . Once we have

this subset, we then find the best labelmap \mathbf{L}^* as we described above. We empirically choose k in the experimental sections 5.7.3 and 5.8.3.

A summary of the shiftmap prior algorithm is presented in Algorithm 1 (on the following page) in the form of a pseudocode.

Algorithm 1 Semantic segmentation using the Shiftmap prior

Input: \mathbf{X}, \mathcal{L}

```

find  $Pr(\mathbf{X}|\mathbf{C})$  {Find data likelihood for each class}
load  $Pr(C_i)$  {Retrieve the per pixel prior for each class}
 $\mathbf{C}^{[t]} \leftarrow \frac{\prod_i Pr(\mathbf{x}_i|\mathbf{C}_i)Pr(C_i)}{Pr(\mathbf{X})}$  {Find the MAP segmentation of class labels}
 $\mathcal{L}_{sub} \leftarrow$  select  $k$  library labelmaps from  $\mathcal{L}$  most similar to  $\mathbf{C}^{[t]}$ 
for  $cIter = 1$  to  $maxIter$  do {Loop over iterations}
  if  $cIter = 1$  then {If this is the first iteration}
    for  $cLib$  to  $k$  do
      [shiftmapCost(cLib),  $\mathbf{S}(:, :, cLib)$ ]  $\leftarrow$  findBestShiftmap[ $\mathcal{L}_{sub}(:, :, cLib)$ ] {Find the
      best shiftmap given current library labelmap}
    end for
    minIdx  $\leftarrow$  find[shiftmapCost= min(shiftmapCost)]
     $\mathbf{L}^* \leftarrow \mathcal{L}_{sub}(:, :, minIdx)$  {Choose the library labelmap with the minimum shiftmap
    cost}
     $\mathbf{S}^{[t]} \leftarrow \mathbf{S}(:, :, minIdx)$  {Store the shiftmap for the chosen labelmap}
  else
    [shiftmapCost,  $\mathbf{S}^{[t]}$ ] = findBestShiftmap[ $\mathbf{L}^*, \mathbf{S}^{[t]}$ ] {Find the best shiftmap given  $\mathbf{L}^*$ }
  end if
  Update  $Pr(\mathbf{C}|\mathbf{S})$ 
  Update  $\mathbf{C}^{[t]}$ 
end for
 $\mathbf{C}^* \leftarrow \mathbf{C}^{[t]}$ 
 $\mathbf{S}^* \leftarrow \mathbf{S}^{[t]}$ 

```

Output: \mathbf{C}^*

5.6 Databases

We test our algorithm on two datasets: faces, and human brains which are described in the following sections.

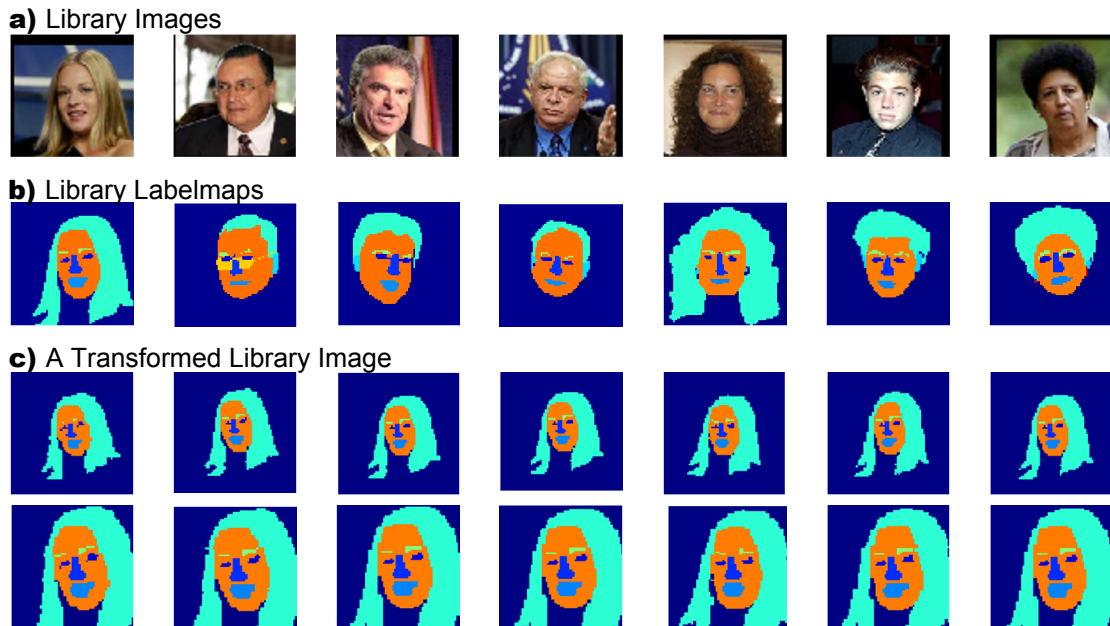


Figure 5.11: Face library images and labelmaps Example library images (a) and their corresponding labelmaps (b). To increase the size of the library, we randomly perturb the library labelmaps using a set of transformations including a rotation of $[-5 : 5]$ degrees and a scaling of $[0.75, 1.25]$. Some of these transformed labelmaps are shown in (c) for an example library image.

5.6.1 Faces

We test our model using images from “Labeled Faces in the Wild” [84]. These are media images collected from internet data using a Viola-Jones face detector, thus representing a moderately unconstrained sample. The original images were registered such that the face appears at the center of the image. To add some noise to the position of each class label we randomly jitter the images using horizontal and vertical shifts of between $\{-26, 26\}$ and $\{-10, 10\}$ respectively. We segmented 150 images (250×250 pixels) by hand into 13 classes to provide ground truth. The classes used were *background*, *hair*, *skin*, *eyes*, *nose*, *mouth*, *ears*, *eyebrows*, *moustache*, *glasses*, *hat*, *earrings*, *beard*. The library consisted of the labelmaps of 100 images, and the remaining 50 images were used to test the model. To increase computational efficiency, the images and labelmaps were resized to 72×72 pixels. Some example library images and their corresponding labelmaps are shown in figure 5.11a-b.

5.6.2 Human brains

We also test our model on a subset of the NIH Pediatric Database [49]. This database contains MR images and correlated clinical/behavioural data on over 500 children, newborn to young adults. We selected 150 T1 weighted MR volumes and their corresponding labelmaps. The labelmaps contained the pixel labels for three classes: *white matter*, *gray matter* and *cerebrospinal*

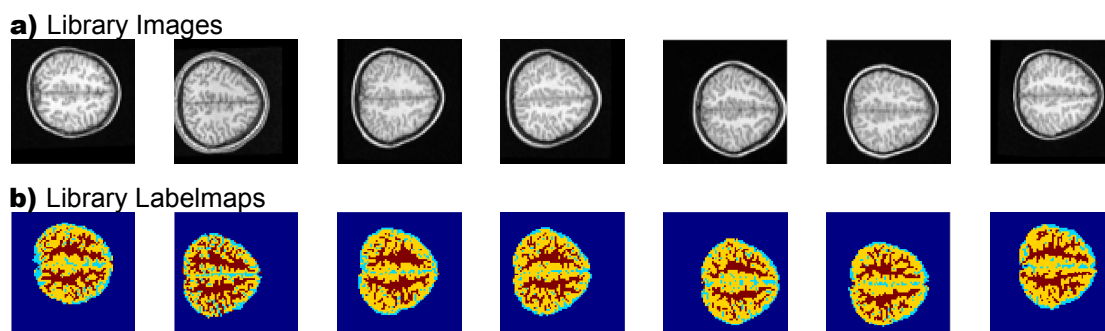


Figure 5.12: Brain library images and labelmaps Example library images (a) and their corresponding labelmaps (b). The labelmaps contain the pixel labels for three classes: white matter (red), gray matter (yellow) and cerebrospinal fluid (cyan). Note, to add some noise to the position of each class label we randomly jitter the images using horizontal and vertical shifts of between $\{-10, 10\}$ and $\{-5, 5\}$ respectively.

fluid (CSF). The volumes contained 189 slices, each with a resolution of 197×233 pixels. In an attempt to have images with all three classes present, we chose the slice 120 for all of the images. The original brain images are registered to a common template. To add some noise to the position of each class label we randomly jitter the images using horizontal and vertical shifts of between $\{-10, 10\}$ and $\{-5, 5\}$ respectively. We subsequently resized the images to 72×72 pixels. As before, we selected 100 images and their labelmaps for the library and 50 for testing. Some example library images and their corresponding labelmaps are shown in figure 5.12.

5.7 Experiments on semantic segmentation of faces

In this section we will test our model on segmenting parts of a face. The goal is to label each pixel as belonging to one of 13 classes: *background, eye, nose, mouth, ear, hair, eyebrows, moustache, glasses, skin, hat, earrings, beard*. Given a test image, we create a labelmap the same size as the test image which contains the estimated class labels for each pixel. The estimated labelmap is made up of a set of non-overlapping patches of a fixed size. The library patches are the same size as the test patches, however they can come from anywhere in the library labelmap.

The test set contains 50 images, and their corresponding labelmaps which are used as ground truth segmentation of parts. The library contains 100 images and their corresponding labelmaps which are not in the test set. The library images are used to learn a Gaussian colour likelihood model for each of the 13 classes. We also learn a per pixel prior for each of the classes using the library labelmaps.

To increase the size of the library, we randomly perturb the library labelmaps using a set

of transformations including a rotation of $[-5 : 5]$ degrees and a scaling of $[0.75, 1.25]$. This results in 23 transformations of each library labelmap making the total size of the library 2300 labelmaps. Figure 5.11c shows some of these transformations for one library image.

To evaluate the performance of our algorithm we use the $F_{0.5}$ measure. This is the harmonic mean of precision and recall which is commonly used to evaluate segmentation models and is defined as:

$$F_{\beta} = (1 + \beta^2) \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}} \quad (5.21)$$

where we set the term β to be 0.5, which means precision is weighted higher than recall. This implies that we encourage the estimated labelmap to contain more of the correct class label in the relevant regions as opposed to finding all of the pixels of a given class in the image. For example, a result where part of the ear is labelled as nose will have a very low score.

We manually fix some of the model parameters such as the maximum shift allowed in the shiftmap prior and the ρ term in Equation 5.6. This is done as follows. We set the maximum shift allowed in the shiftmap prior, to the maximum disparity between two corresponding points across all library images. For example, we measured the distance between the corner of the left eye in one library image and the corner of the left eye in all the other library images, we repeated this for a few random examples and set the largest possible shift to be the maximum shift. For all the face experiments, the maximum shift was set to 15 pixels in each of the horizontal and vertical directions.

We investigate the effects of the remaining model parameters in the next few sections

5.7.1 Experiment 1: Varying the parameter λ_1

In this experiment we investigate the effect of the λ_1 parameter in Equation 5.17 on segmenting parts of a face. We vary this term as $\lambda_1 = \{0.1, 0.2, \dots, 0.9, 1\}$ while we keep the rest of the parameters fixed as follows: we set the maximum number of shifts used in MRF to 15, we set the number of closest library labelmaps to 1 and we set the term λ_2 in Equation 5.17 to 1 and we set the $\theta_{correct}$ term to be 0.88 based on some preliminary experiments.

The term λ_1 weights the probability of each class conditioned on the shiftmap i.e. $Pr(C|S)$. Therefore a low value of λ_1 will encourage the estimated labelmap \mathbf{C}^* to look similar to the segmentation using the data likelihood only i.e. $\mathbf{C}^{[t]}$ and a high value of λ_1 will encourage the \mathbf{C}^* to look similar to the warped library labelmap $\mathbf{R} = \mathbf{L}_{i+S(i)}$.

We plot the results in figure 5.13a for a number of different patch sizes such as 3×3 , 6×6 , 9×9 , 12×12 pixel patches, as well as 1×1 pixel patches i.e. using all of the pixels in the image. The x-axis represents the value used for λ_1 and the y-axis shows the average $F_{0.5}$ for

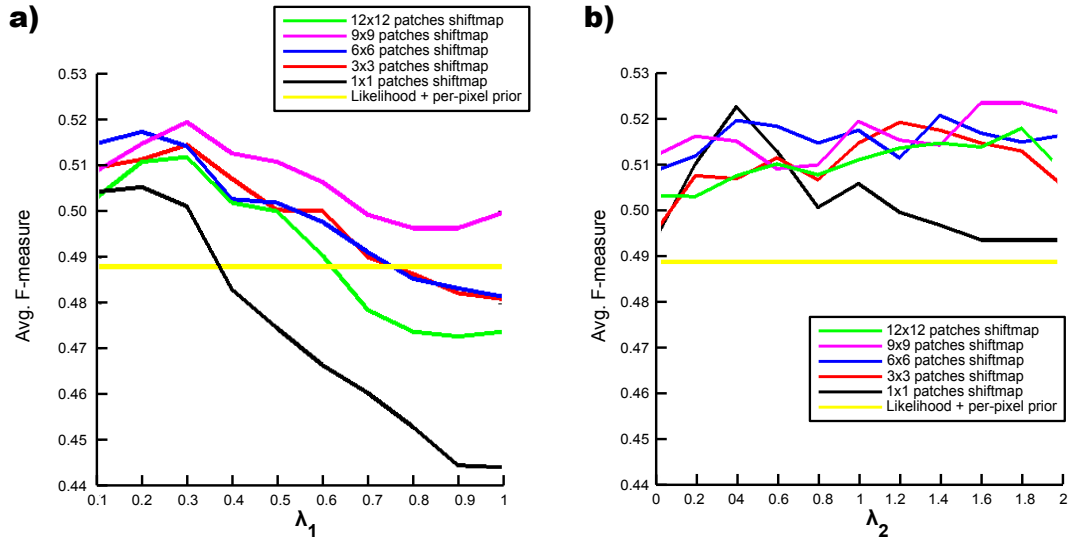


Figure 5.13: *The effect of the parameters λ_1 and λ_2 on semantic segmentation of faces: (a) We vary the term λ_1 while keeping $\lambda_2 = 1$ and holding all the other model parameters fixed. The y-axis shows the average F-measure for all 50 test images. We repeat this experiment for a variety of patch-sizes as well as using pixels. The results show that the term λ_1 has a great influence on the performance and it mostly peaks at $\lambda_1 = 0.3$ or $\lambda_1 = 0.2$. The best performance is achieved by 9×9 patches and $\lambda_1 = 0.3$, with an F-measure of 0.519. (b) We now keep λ_1 fixed at the best value for each patch-size and vary the term λ_2 while holding the rest of the model parameters fixed. Note, the results improve when λ_2 is tuned, however the effect of this term is less compared to λ_1 . Performance peaks with 9×9 patches and $\lambda_2 = 1.4$, with an F-measure of 0.524. For comparison, we also plot the result for using a per-pixel prior. In all of the cases the shiftmap prior improves the per-pixel prior results.*

all 50 library images. Since some of the classes are rare such as hat or earrings, we only use the classes present in each image to calculate the F-measure. The results show that the term λ_1 has a great influence on the performance and this mostly peaks at $\lambda_1 = 0.3$ or 0.2 . The best performance is achieved by 9×9 patches and $\lambda_1 = 0.3$, with an F-measure of 0.519. The term λ_1 weights the probability of each class conditioned on the shiftmap i.e. $Pr(C|S)$. Therefore, a low value of λ_1 means that, a set of shifts that create a labelmap similar to the current maximum likelihood estimate, become more probable.

Figure 5.14 shows the qualitative results of varying λ_1 . As expected, the results suggest that as the value of λ_1 increases the final segmentation C^* look more similar to the warped labelmap.

The likelihood model only performs very weakly for the face dataset with an F-measure of 0.261, since we only have 100 images to learn a colour model. Therefore for comparison, we plot the MAP segmentation results using the likelihood and the learned per-pixel class label

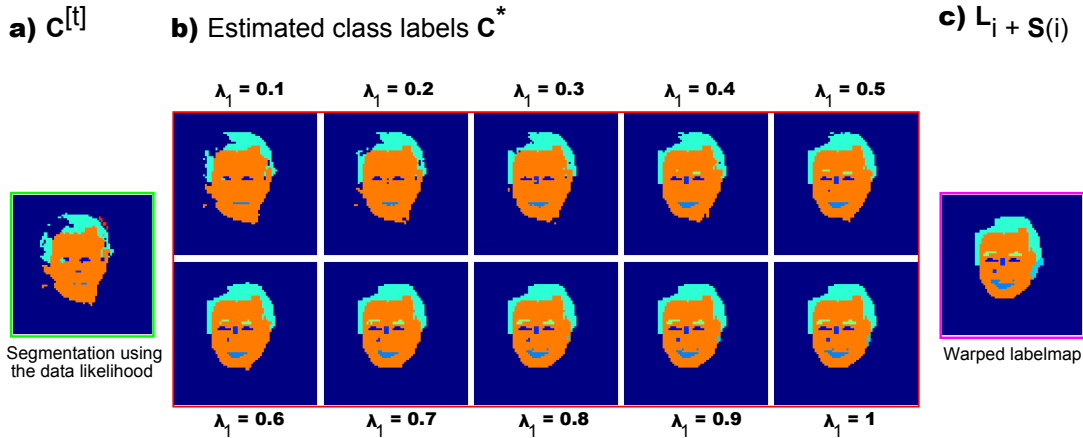


Figure 5.14: Qualitative results of varying the parameters λ_1 on semantic segmentation of faces: The term λ_1 weights the probability of each class conditioned on the shiftmap i.e. $Pr(C|S)$. Therefore a low value of λ_1 will encourage the estimated labelmap C^* to look similar to the segmentation using the data likelihood only i.e. $C^{[t]}$ and a high value of λ_1 will encourage the C^* to look similar to the warped labelmap $L + L$. This figure shows a typical result which demonstrates this effect. Where (a) shows the data likelihood segmentation $C^{[t]}$, (c) shows the warped library image $L + L$. (b) As we would expect as the value of λ_1 increases the final segmentation C^* look more similar to the warped labelmap.

prior instead (the yellow line) which results in a $F_{0.5} = 0.488$. It is clear that the shiftmap algorithm improves the baseline MAP model.

We also plot the qualitative results of changing the patch size in figure 5.15. We observe that when using single pixels 5.15c, the warped labelmap and therefore the final estimate C^* has many holes (e.g. on the outline of the hair and the eyes). This is because each pixel is allowed to change label individually to match the $C^{[t]}$ (we assume no or low pairwise penalty here). However as we increase the patch-size 5.15(d-f), the warped labelmap has a smoother outline and less holes or isolated class labels. This is because it is very unlikely that we observe an isolated patch or a patch with a hole in the library labelmap.

5.7.2 Experiment 2: Varying the parameter λ_2

In this experiment we vary the term λ_2 in Equation 5.17 which is the weighting for the pairwise term in the MRF, while holding all the other parameters fixed as in experiment 1, and fixing the term λ_1 to have the optimal value for each patch size. As before we repeat this experiment for different patch sizes as well as for pixels.

The term λ_2 is the weighting for the pairwise cost in the MRF model. If the value of λ_2 is low, it implies that there is a relatively small penalty for the neighbouring pixels to have different labels. Therefore it allows a large warping of the library labelmap. Alternatively, a high value of λ_2 incurs a large penalty for neighbouring pixels to have different labels. This

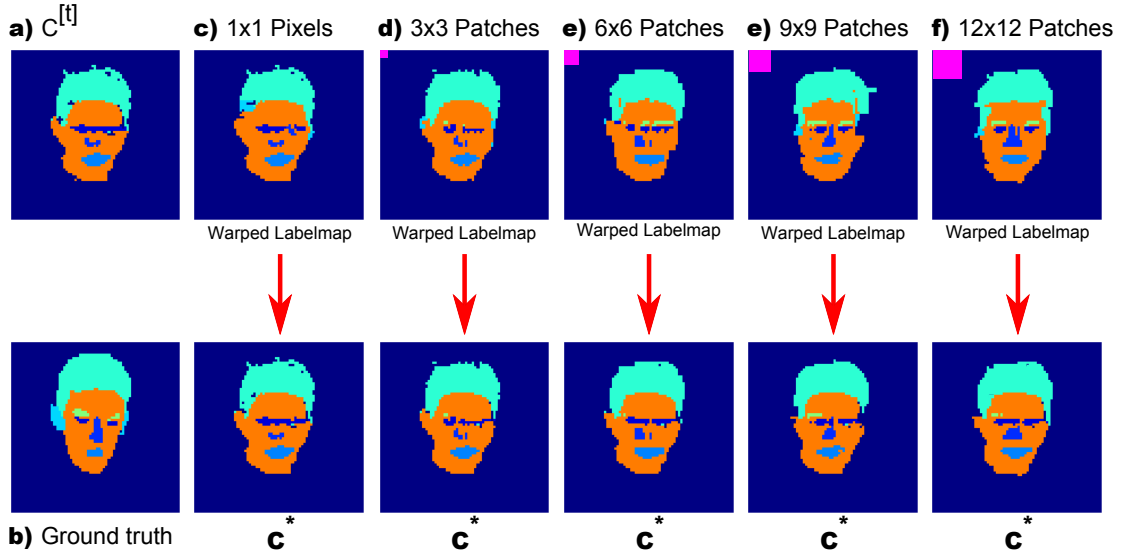


Figure 5.15: *Qualitative results of varying patch sizes on semantic segmentation of faces:* (a) We show (a) the maximum likelihood segmentation of face, (b) the ground-truth segmentation and (c-f) the warped library image (top) and final estimated class labels C^* (bottom) for several patch sizes. The patch size is illustrated by the pink square at the top left corner of each image. We observe that (c) when using single pixels, the warped labelmap and therefore the final estimate C^* has many holes (e.g. on the outline of the hair and the eyes). This is because each pixel is allowed to change label individually to match the $C^{[t]}$ (We assume no or low pairwise penalty here). However as we increase the patch-size (d-f) the warped labelmap has a smoother outline and less holes or isolated class labels. This is because it very unlikely that we observe an isolated patch or a patch with a hole in the library labelmap.

results in a shiftmap with a more smooth transition between different labels. Thus, it allows a limited warping of the library labelmap.

We plot the results for $\lambda_2 = [0, 2]$ in figure 5.13b. The x-axis represents the value used for λ_2 and the y-axis shows the average non-zero $F_{0.5}$ for all 50 library images. Note, the results improve when λ_2 is tuned, however the effect of this term is less, compared to that of λ_1 . The best performance is achieved by 9×9 patches and $\lambda_2 = 1.4$, with an F-measure of 0.524. For comparison, we also plot the result for using a per-pixel prior. In all of the cases the shiftmap prior improves the per-pixel prior results, as well as outperforming the ‘likelihood only’ results of $F = 0.261$. The term λ_2 represents the ratio between unary and pairwise costs in the MRF used in shiftmap prior. These are also known as the data term and the prior term respectively. A high value of λ_2 suggests that the algorithm performs best when there is a high smoothness constraint. In terms of the MRF this means that there is a relatively high penalty for the neighbouring patches to have different shifts. This results in copying patches as verbatim from the library labelmap.

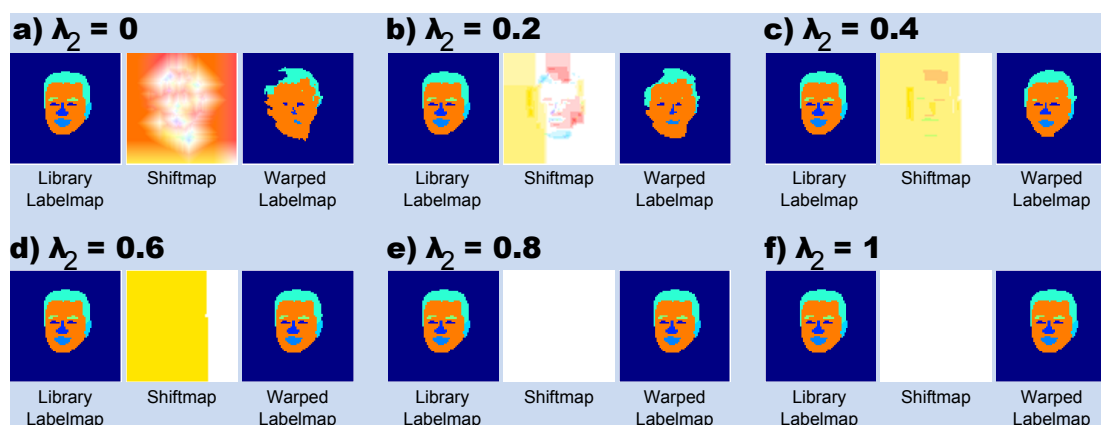


Figure 5.16: Qualitative results of varying the parameters λ_2 on semantic segmentation of faces: The term λ_2 is the weighting for the pairwise cost in the MRF model. Show some examples of the library labelmap (left), the shiftmap (middle), where each color represent a different label, and the warped labelmap (right). We observe that (a) when λ_2 is set to be zero (i.e. no penalty for neighbours having different labels), the resulting shiftmap is very noisy in that it allows all the possible shifts which largely warps the library labelmap. (b-d) As we increase the value of λ_2 the shiftmap becomes smoother as large regions are forced to have a similar label. (e,f) When λ_2 becomes very high, it incurs a very large penalty for the labels to change at all. Therefore, the cheapest solution is when there is a zero shift which results in the library labelmap not being warped at all.

Figure 5.16 shows the qualitative results of varying λ_2 . The results confirm that as we increase the value of λ_2 the shiftmap becomes smoother which results in a smoother warping of the library labelmap.

To verify that the warping of the library image using the shiftmap is improving the overall segmentation we repeat the above experiment for 3×3 patches with the best parameters for this patch size which are $\lambda_1 = 0.3$ and $\lambda_2 = 1.2$, this time using the most likely library labelmap without warping with a shiftmap. This results in a much worse performance with a F-measure of 0.373 compared to $F = 0.519$ with the shiftmap prior.

5.7.3 Experiment 3: Varying the number of top matched library labelmaps

So far we have used a single library labelmap which was the most likely match to the data likelihood. This implies that we have assumed that there is a single labelmap in the library which looks very similar to the true test class labels. This is however not always true as we may have a small and limited library. In this experiment we relax this assumption by choosing more than one library image and letting the model decide which one to use to predict the class labels of the test image. In the first iteration of the model we choose the top l most likely library labelmaps. We perform the shiftmap alignment. We then choose the library labelmap with the

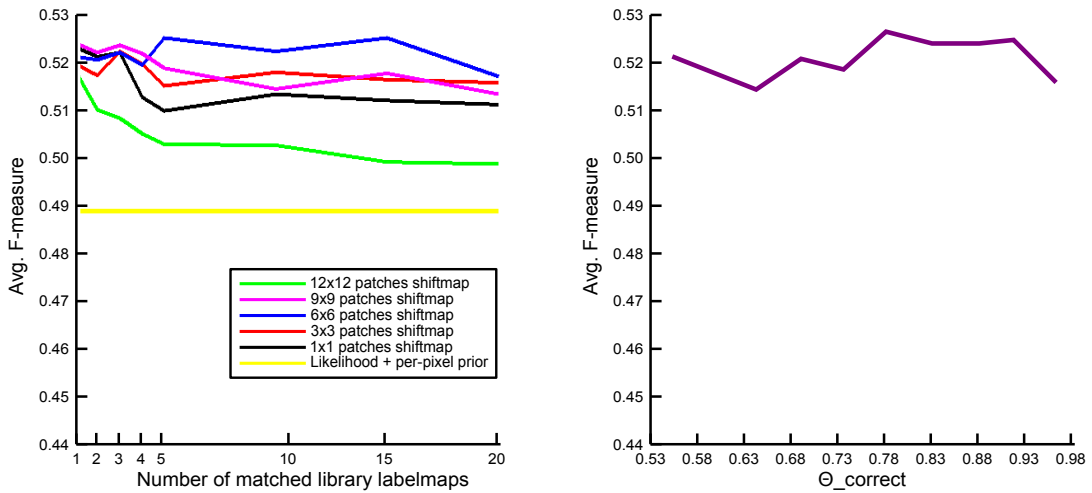


Figure 5.17: *The effect of the number of top matched library images and the term $\theta_{correct}$ on semantic segmentation of faces:* (a) *We relax the assumption of having only one library image that closely matched the test image. Instead we select a number of library images that closely match the test data. We vary this number to $[1, 2, 3, 5, 10, 15, 20]$ and show the effect this has on the results. The performance peaks with 6×6 patches and 5 library labelmaps with an F-measure of 0.525.* (b) *We vary the term $\theta_{correct}$ which is used as a measure of evidence for class labels given the shiftmap. The results peak with $\theta_{correct} = 0.78$ and an F-measure of 0.527.*

minimum cost (i.e. the maximum $\Pr(S)$).

In this experiment we vary the number of top matched library images as $l = \{1, 2, 3, 4, 5, 10, 15, 20\}$ by keeping the optimal parameters λ_1, λ_2 for each patch size, while holding the rest of the parameters the same as in experiment 1. The results are shown in figure 5.17a. The performance peaks with 6×6 patches and 5 library labelmaps with an F-measure of 0.525. This suggests that the top most likely match might not necessarily be the best in predicting the class labels.

5.7.4 Experiment 4: Varying the term $\theta_{correct}$

The term $\theta_{correct}$ is a parameter of the probability of each class label conditioned on the shiftmap $\Pr(C|S)$. If this term is set to be close to 1, it implies that probability $\Pr(C_i|S_i)$ becomes high when the pixel in the warped image resulting from the shift S_i has the same value as the current class label C_i . This probability is low otherwise.

In this experiment we vary the value of $\theta_{correct}$, while keeping the rest of the model parameters fixed at their optimal value. The results are shown in figure 5.17b. The performance peaks with $\theta_{correct} = 0.78$. This suggests that a high value of $\theta_{correct}$ is preferred which means the probabilities $\Pr(C_i|S_i)$ become more certain. However, if they become over-confident i.e.

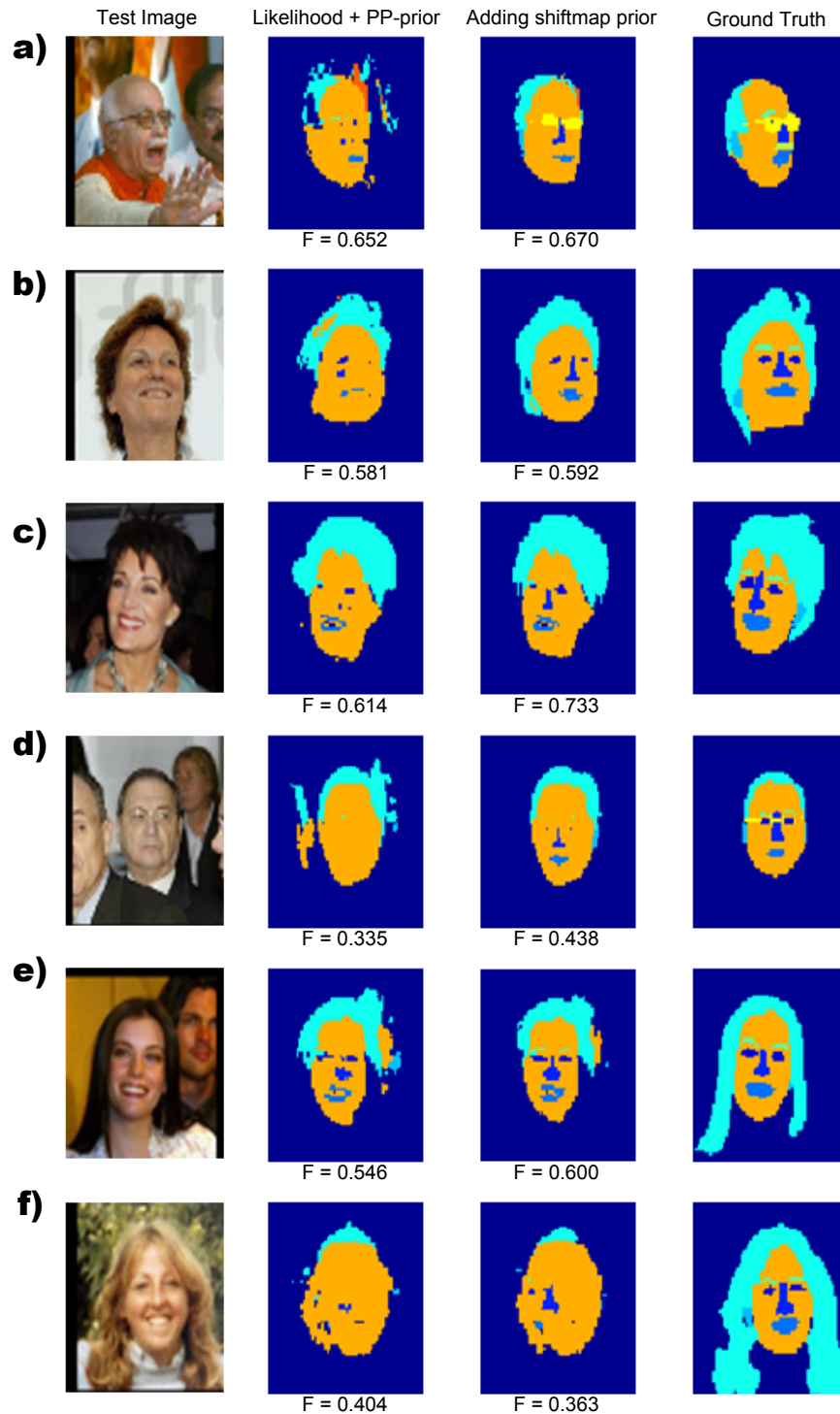


Figure 5.18: Example results on semantic segmentation of faces using the shiftmap prior: Figure 5.18a-e, show successful examples where the shiftmap prior not only improved the F -measure, but it also manages to recover some classes that were initially missed by the per-pixel prior. For example, glasses the nose and the eyebrows are recovered in (a), (b-c), and (e) respectively. In (d) despite a weak likelihood model the shiftmap prior has improved the segmentation result. (f) Unfortunately in the case of an extremely poor likelihood estimate the shiftmap has a slim chance of recovering.

$\theta_{correct} \geq 0.93$ the model will only allow very small contributions from the other components of the model such as the data likelihood, which in this case has resulted in reduced performance.

5.7.5 Sample Results

Some example results are shown in figure 5.18. The first and the last column show the test image and the ground truth class labels respectively. The second column shows the segmentation result using just the per-pixel prior and the third column shows the segmentation results when adding the shiftmap prior. Figure 5.18a-d, show successful examples where the shiftmap prior not only improves the F-measure, but it also manages to recover some classes that were initially missed by the per-pixel prior. For example in figure 5.18a, the shiftmap prior recovers glasses. Similarly in figure 5.18b-c, the nose is recovered. This shows that the prior knows something about the structure of faces, which is achieved through the library labelmaps. In figure 5.18d the shiftmap prior has improved the segmentation result despite a weak likelihood model.

Figure 5.18e shows an example where the likelihood model has failed to predict long hair and has completely missed the eyebrows. The shiftmap prior in this example, has successfully recovered the eyebrows, but has not managed to recover the long hair. This is because allowing a large region in the estimated image to have a label which does not agree with the likelihood model, has a high cost. In spite of this the shiftmap prior has improved the final F-measure. Finally, 5.18f shows an example of extremely weak likelihood segmentation. Unfortunately under such circumstances the shiftmap has a very slim chance of improving the results. The results of the face segmentation experiments are summarized in Table 5.1.

Likelihood	Likelihood + pp-Prior	Shiftmap Prior-Pixels	Shiftmap Prior 6×6 Patches
0.261	0.488	0.522	0.527

Table 5.1: The average $F_{0.5}$ measure for the results on semantic segmentation of faces across 50 test images.

5.8 Experiments on semantic segmentation of brains

In this section we will test our algorithm on semantic segmentation of human brains. Given a slice from an MRI scan of a brain, we will label each pixel as belonging to one the following three classes: *white matter*, *gray matter* and *CSF*. As before, the predicted labelmap is made up of a set of non-overlapping patches of a fixed size which are extracted from a library of labelmaps.

We test our algorithm on 50 MRI brain images randomly selected from the database de-

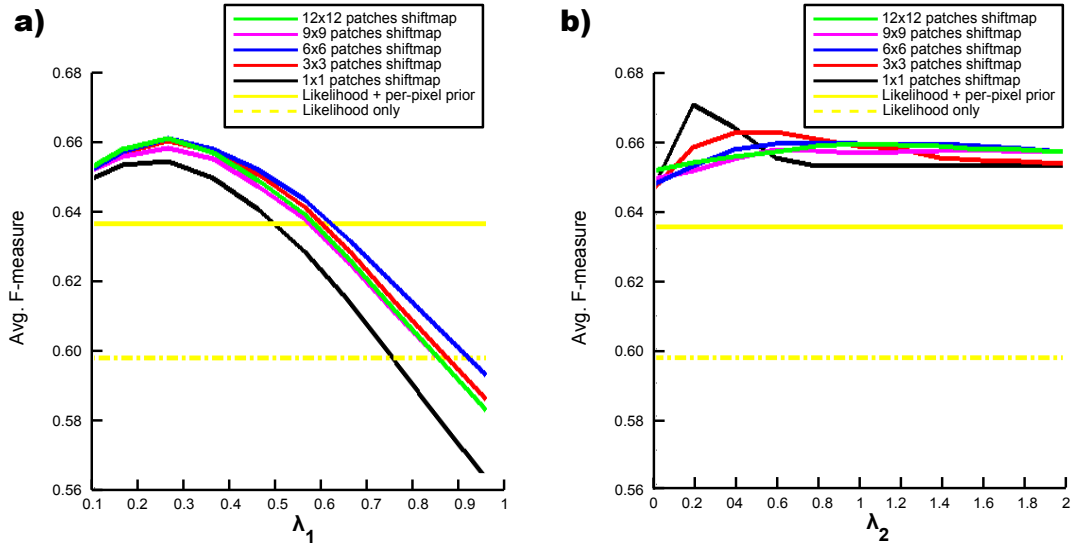


Figure 5.19: *The effect of the parameters λ_1 and λ_2 on semantic segmentation of human brains:* (a) We vary the term λ_1 while keeping $\lambda_2 = 1$ and holding all the other model parameters fixed. The y-axis shows the average F-measure for all 50 test images. We repeat this experiment for a variety of patch-sizes as well as using pixels. The optimal value for all patch sizes is $\lambda_1 = 0.3$. The overall best performance is achieved by 6×6 patches with an F-measure of 0.661. (b) We now keep λ_1 fixed at the 0.3 and vary the term λ_2 while holding the rest of the model parameters fixed. The performance peaks with 1×1 pixel patches, with $\lambda_2 = 0.2$, with an F-measure of 0.672. For comparison, we also plot the segmentation result using ‘likelihood only’ (yellow dotted line) and likelihood plus per-pixel prior (yellow line). In all of the cases the shiftmap prior achieves superior results.

scribed in section 5.6. These images represent the 120th slice from 50 different individuals. The library consists of brain images extracted from the 120th of 100 different individuals that were not in the test set. The library images are used to learn a Gaussian colour model for each class, as a data likelihood model. The corresponding library labelmaps are used to learn a per-pixel prior for each class label. To evaluate the performance of our algorithm we use the $F_{0.5}$ measure.

As before we manually set some of the model parameters. To set the maximum allowed label (shift) in the shiftmap prior, we follow the same procedure as in the face experiment. In this case we set the maximum allowed shift to be 7 pixels in each of the horizontal and vertical directions. We set the ρ term to be $\frac{1-\theta_{correct}}{K}$ where K is the number of class labels. The pairwise cost then becomes $-\log[1 - \theta_{correct}K]$.

In the rest of this section, we will present experiments to evaluate the effects of the parameters λ_1 , λ_2 and the number of the chosen library labelmaps, on the performance of our algorithm.

5.8.1 Experiment 5: Varying the parameter λ_1

In this experiment we investigate the effect of the λ_1 parameter in Equation 5.17 on semantic segmentation of brains. We vary this term as $\lambda_1 = \{0.1, 0.2, \dots, 0.9, 1\}$ while we keep the rest of the parameters fixed as follows: we set the maximum number of shifts used in the MRF to 7, we set the number of closest library labelmaps to 1 and we set the term λ_2 in Equation 5.17 to 1 and we set the $\theta_{correct}$ term to be 0.88.

Figure 5.19a shows the results of varying the term λ_1 for a number of different patch sizes such as 3×3 , 6×6 , 9×9 , 12×12 pixel patches, as well as using 1×1 pixel patches i.e. all of the pixels in the image. The x-axis represents the value used for λ_1 and the y-axis shows the average $F_{0.5}$ for all 50 library images. Once again the results show that the term λ_1 has a greatly influence the results. The performance of all patch sizes consistently peaks with $\lambda_1 = 0.3$. The best performance is achieved by 6×6 patches, with an F-measure of 0.661. A small λ_1 encourages the predicted labelmap to be similar to the current estimate of labels from the likelihood model. For comparison, we plot the segmentation results using only the likelihood as well as the likelihood plus the learned per-pixel class label prior (MAP). These results are a $F_{0.5} = 0.597$ and $F_{0.5} = 0.636$ respectively. Once again, the shiftmap prior improves the performance.

5.8.2 Experiment 6: Varying the parameter λ_2

In this experiment we vary the term λ_2 in Equation 5.17 which is the weighting for the pairwise term in the MRF, while holding all the other parameters fixed as in experiment 1, and fixing the term $\lambda_1 = 0.3$. As before we repeat this experiment for different patch sizes as well as for pixels.

We plot the results for $\lambda_2 = [0, 2]$ in figure 5.19b. The x-axis represents the value used for λ_2 and the y-axis shows the average non-zero $F_{0.5}$ for all 50 library images. Note, the results improve when λ_2 is tuned, however the effect of this term is less, compared to that of λ_1 . The performance increases with a smaller value of λ_2 , which was not the case in our face experiments. A small value of λ_2 , means that the pairwise cost in the shiftmap is relatively low compared to the unary or the data cost. Thus, the shiftmap is allowed make more changes to the library labelmap making it closer to the current maximum likelihood estimate of class labels. This observation suggests that, compared to the labelmaps in the face library, the MRI labelmaps in the library are not very similar to those of the test images. This is most probably due to the fact that the fine details of the brain structure is different for two different individuals. This also explains why the shiftmap model prefers a finer patch-size or even single pixels to best predict the test labelmap. Another possible explanation is that the likelihood is weaker in this

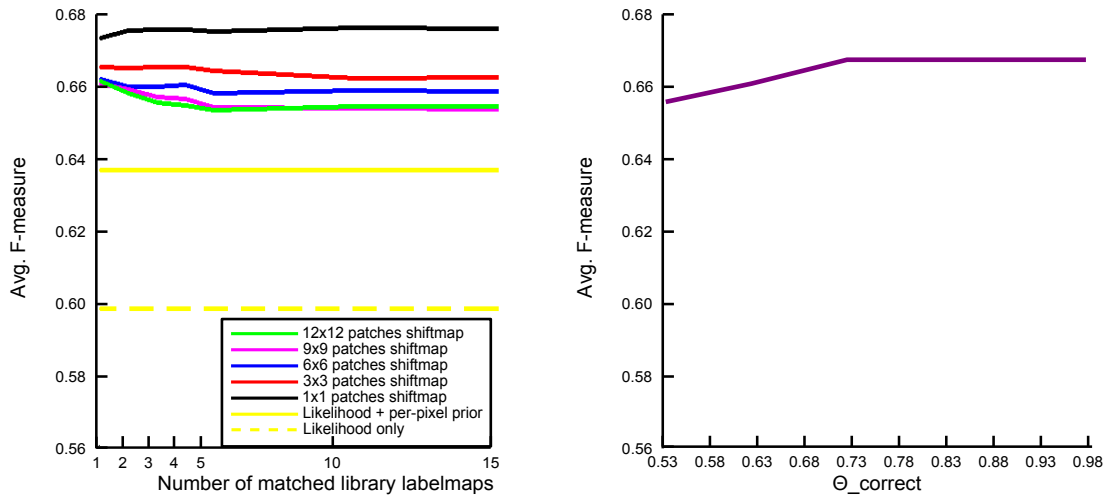


Figure 5.20: *The effect of the number of top matched library images and the term $\theta_{correct}$ on semantic segmentation of brains: (a) We relax the assumption of having only one library image that closely matched the test image. Instead we select a number of library images that closely match the test data. We vary this number to $[1, 2, 3, 5, 10, 15, 20]$. Despite the slight increase in performance for some patch-sizes such as 1×1 and 3×3 , overall the effect of changing the number of matched library labelmaps is not considerable. This may suggest that the library does not contain labelmaps that are very similar to the test images. (b) We vary the term $\theta_{correct}$ which is used as a measure of evidence for class labels given the shiftmap. The results peak with $\theta_{correct} \geq 0.73$ and an F-measure of 0.677*

case, hence it is better to make more changes. The best performance is achieved by 1×1 pixels and $\lambda_2 = 0.2$, with an F-measure of 0.672. The second best result is achieved by 3×3 patches and $\lambda_2 = 0.3$ with an F-measure of 0.665.

5.8.3 Experiment 7: Varying the number of top matched library labelmaps

So far we have used a single library labelmap which was the most likely match to the data likelihood. This implies that we have assumed that there is a single labelmap in the library which looks very similar to the true test class labels. This is however not always true as we may have a small and limited library. In this experiment we relax this assumption by choosing more than one library images and letting the model decide which one to use to predict the class labels of the test image. In the first iteration we choose the top l most likely library labelmaps, perform shiftmap alignment and then choose the library labelmap with the minimum cost.

In this experiment we vary the number of top matched library images as $l = \{1, 2, 3, 4, 5, 10, 15\}$ by keeping the optimal parameters λ_1 , λ_2 for each patch size, while holding the rest of the parameters the same as in experiment 4. The results are shown in figure 5.20a. Despite the slight increase in performance for some patch-sizes such as 1×1 and 3×3 ,

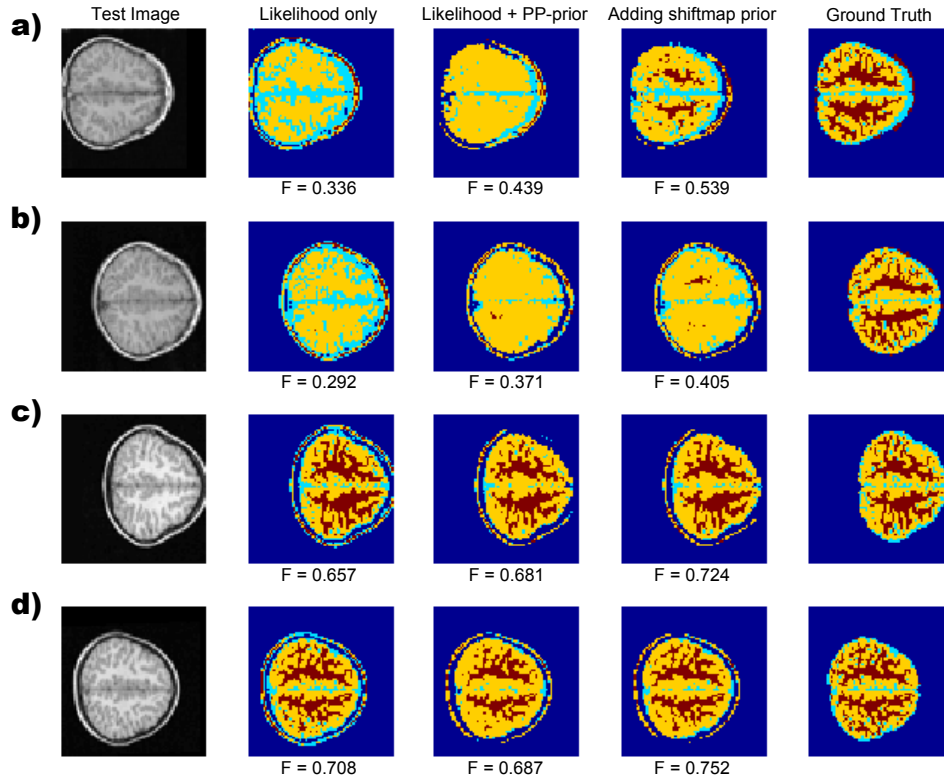


Figure 5.21: Example results on semantic segmentation of brains using the shiftmap prior: The colors cyan, yellow and red represent CSF, gray matter and the white matter respectively. (a-b) Examples where the gray matter pixels are largely missed by the likelihood and the per-pixel prior models, but are partially recovered with the shiftmap prior, resulting in an increased F -measure. (c) An example where the details of the class structure is improved by having more continuous regions of CSF and gray matter. (d) The shiftmap model improves the final segmentation, despite the likelihood and per-pixel prior model having worse segmentation than the ‘likelihood only’ model.

the overall effect of changing the number of matched library labelmaps is not considerable. This once again suggests that the library does not contain labelmaps that are very similar to the test images.

5.8.4 Experiment 8: Varying the term $\theta_{correct}$

The term $\theta_{correct}$ is a parameter of the probability of each class label conditioned on the shiftmap $Pr(\mathbf{C}|\mathbf{S})$. If this term is set to be close to 1. This implies that probability $Pr(C_i|S_i)$ becomes high when the pixel in the warped image resulting from the shift S_i has the same value as the current class label C_i . This probability is low otherwise.

In this experiment we vary the value of $\theta_{correct}$, while keeping the rest of the model parameters fixed at their optimal value. The results are shown in figure 5.20b. The performance

peaks with $\theta_{correct} \geq 0.73$. The overall effect of this term however, is not very significant in comparison to the rest of the model parameters.

5.8.5 Sample Results

Some example results of semantic segmentation of brains are shown in figure 5.21. The first and the last columns show the test image and the ground truth respectively. The second through fourth columns show the segmentation results using the likelihood model only, likelihood plus per-pixel label prior and with added shiftmap prior respectively. The colors cyan, yellow and red represent CSF, gray matter and the white matter respectively. The images in figure 5.21a-b, show two examples where one of the classes in this case the gray matter was largely missed in the segmentation results of the likelihood and the per-pixel prior models, but was partially recovered with the shiftmap prior. This results in an increased F-measure. Figure 5.21c shows an example where the details of the class structure were improved by having more continuous regions of CSF and gray matter. Finally, figure 5.21d showcases an example where using the per-pixel prior results in a worse segmentation than the ‘likelihood only’ model, but the shiftmap model improves the final segmentation regardless. We would like to note that the shiftmap prior has improved the segmentation results of each and every one of the 50 test images. The results of the brain segmentation are summarized in table 5.2.

Likelihood	Likelihood + pp-Prior	Shiftmap Prior 3×3 Patches	Shiftmap Prior-Pixels
0.597	0.636	0.665	0.677

Table 5.2: The average $F_{0.5}$ measure for results of semantic segmentation of brains across 50 test images.

5.9 Summary and Discussion

We have presented a patch-based model for semantic segmentation of objects which we have tested on segmenting faces and human brains. The work carried out in this chapter has made several contributions towards our research objectives: (i) we have demonstrated yet another object-related task in which patch-based representation has been used successfully. (ii) We have used a patch-based model to represent two object classes which are completely different in structure (e.g. faces and brains) and (iii) have shown that patches have been robust to the issues of variation in pose, and scale.

Our model was inspired by recent advances in computer graphics. In particular we have extended the ‘shiftmap’ model proposed by Pritch et al. [142] to work with class labels, and

used it in the form of a prior to segment parts of an object. Our results show that using a superior segmentation results are achieved when using a shiftmap prior. The results demonstrate that large patch sizes such as 9×9 pixels are optimal for semantic segmentation of faces, with an F-measure of 0.524. This compares favorably to that of the likelihood and per-pixel prior model which achieves an F-measure of 0.488. However, when dealing with segmentation of human brains, our results demonstrate that smaller patches such as 3×3 pixels, or using pixels entirely instead of patches is preferred. We suspect this is the between individual variation that exist in MRI scans of the brain. Thus, using pixels will help capturing such fine details when segmenting parts of a brain.

Our model has several interesting properties: (i) we have proposed a patch-based prior which can be combined with any likelihood model in Bayes rule to produce segmentation results. (ii) The data driven approach of our model allows us to exploit large databases of a given object, in the form of a library, to find a suitable segmentation. (iii) Our model is tolerant to the errors of the likelihood model and the lack of strong similarity between the library labelmaps and a given test image. Of course, if the likelihood is extremely weak, the final segmentation will not be excellent. However, the model parameters are set up such that, when the likelihood is weak, it relies more on the chosen library labelmap to predict the class label. In most cases, this results in the shiftmap prior recovering some classes that the likelihood had initially missed such as the nose, or the eyebrows in face images. (iv) The generative nature of our model, allows us to generate novel labelmaps by perturbing the class label of a given library image while preserving the structure of the object. (v) Finally, the model can be formulated as an energy maximization problem to exploit efficient learning and inference algorithms.

Our model has connections to the belief propagation algorithm. The iterative nature of our model to find the MAP estimate of the class labels, where at each iteration one term is fixed and the other two terms are updated, is similar to the message passing system in the belief propagation algorithm. In our model, there are three factors that contribute to the conditional probability of class labels given the test image. First, there is the likelihood of the data given each class label, then there is the probability of each class label conditioned on the shiftmap (i.e. the unary term), and the prior the over shiftmap (the pairwise term). To update the current estimate of the class labels at each iteration, we first keep the likelihood term fixed, and update the unary and the pairwise terms. Then we keep the unary and the pairwise terms fixed and update the likelihood term. This is in a way similar to the message update in the belief propagation method, in a sense that each time the contribution of two factors are used to update the third factor.

Our method is similar to the “Multiscale Conditional Random Fields for Image Labelling” by He et. al [77], in that we both use patches to represent class labels: our predicted labelmap is created as a composition of non-overlapping patches. In [77] features are used at two difference scales: global and regional. The former includes a set of larger patches which capture information about context and the relative position of class labels, and the latter captures finer details such as edges etc. The chosen library labelmap for each test image in our algorithm, can be thought of as the global feature in [77] which captures the structure of the object, while the patches extracted from this labelmap are equivalent to the regional features in [77] which are chosen such that they best represent the given test image. Note, despite using similar representation, we have a simpler inference model compared to [77], since we do not require sampling during inference.

Currently our model is used as a prior, guided by a likelihood term to predict a labelmap. However, the shiftmap prior can potentially be combined with a set of grammar rules such as those used in [73] to generate new labelmaps. The core of our model is based on permuting labelmaps in a consistent manner to create an unseen labelmap of any size or to complete a missing part of an image. For example, we could create a new labelmap by taking parts from several different library images and combining them using the alpha expansion algorithm [22]. Furthermore, one can enforce a set of rules in terms of a cost function to guide the generated output. For example, consider the task of generating a new house labelmap. We can start with an incomplete labelmap, where part of the roof, a window and part of the door are given. We can then use the shiftmap prior to complete the object according to the given grammar e.g. the roof is always higher than the door. Possible future directions include application of labelmap generation using shiftmap for medical image atlas generation and registration.

Chapter 6

Patch-based Identification

6.1 Introduction

So far we have presented patch-based models for categorization, regression and semantic segmentation tasks. To extend the diversity of tasks that patch-based representation can handle, in this chapter we will explore a patch-based representation for a completely different type of problem : identity recognition. In particular we will apply this to face recognition. The model we develop in this chapter shares the core properties of other models we have developed so far: (i) an image is represented as a set of non-overlapping regular grid of patches and (ii) there is a library of patches from which the test image is approximated. However, there is one major difference and that is the fact that we are now going to learn the library of patches from a set of training data, unlike our previous models where the library was predefined.

The area of face recognition has grown significantly in importance during the past decade. Several factors contribute to this growth. First, the availability of powerful computers and real-time hardware makes the face recognitions task easier to implement. Second, the emergence of large databases such as FERET and XM2VTS, provide sample images for testing the algorithms. Finally, systematic empirical evaluation techniques (FRT), allow for benchmark comparison of face recognition algorithms.

Face recognition is often associated with access control and surveillance. However, an automated non-invasive technique for establishing identity would have pervasive applications: almost every human-machine interface could be improved if the machine recognized the user and tailored its features according to their preferences. For example, a personal computer with a camera attached, could use face recognition to log in an individual instead of using the conventional username and password protocol. Face recognition can also be used for archiving personal photographs, where it can associate faces with persons. Subsequently, queries can be run on pictures to return all pictures with a specific group of people together. Other example

applications include virtual reality and video games.

6.2 Motivation

Most face recognition methods make decisions based on a *distance measure*. Images are projected down to a lower-dimensional feature space. Distances between feature space representations are used as the basis for recognition judgements. For example in an identification task, distance-based algorithms choose the gallery feature that is nearest to the probe feature.

A large body of research has investigated suitable choices of feature space within which to calculate distances. The goal is to find a space where position correlates strongly with identity, but not with extraneous factors such as pose and illumination. Turk and Pentland [161] selected the linear feature space that best approximated the image data. Subsequent work has variously investigated different linear feature spaces [14, 12, 44] and distance metrics [136].

One drawback of these distance-based methods is that they almost always entail a *unimodal* model of faces: they model the face manifold with a single cluster. However, in reality the face manifold has a more complex structure. Moreover, the distance metric (discriminability) is usually treated as independent of position in feature space. This implicitly assumes that the degree of uncertainty about identity (noise) is independent of the position in the feature space. This is almost certainly not the case.

Despite these problems, a particularly successful subclass of distance-based methods applies linear discriminant analysis (LDA) to face recognition [14, 167]. These techniques select the subspace that maximizes the ratio of between-class variation to within-class variation. LDA methods effectively eliminate or downweight directions in which the signal to noise ratio is bad.

Unfortunately, these algorithms cannot distinguish noise from unmodeled, but essentially deterministic factors such as pose or expression changes. For example, consider matching images of smiling and frowning faces: the LDA approach does not exploit the fact that the smiling face might be entirely predictable from the frowning version. Instead, it treats the mouth region as uncertain and bases its judgements on other parts of the image. In some cases this can be disastrous: with large pose changes, almost all of the identity signal is discarded as it shares the same subspace as the image changes due to pose variation. We propose a novel representation for faces that can potentially resolve these issues. Our approach is inspired by recent successes of *patch-based methods* which have shown to be highly effective for texture generation [46], super-resolution [62] and image denoising [148]. Our model represents a face image as *a composite picture made up of non-overlapping smaller patches*. Hence, we call it a “Mosaicface”. Each patch is taken from a library of discrete possibilities, which is learnt in the

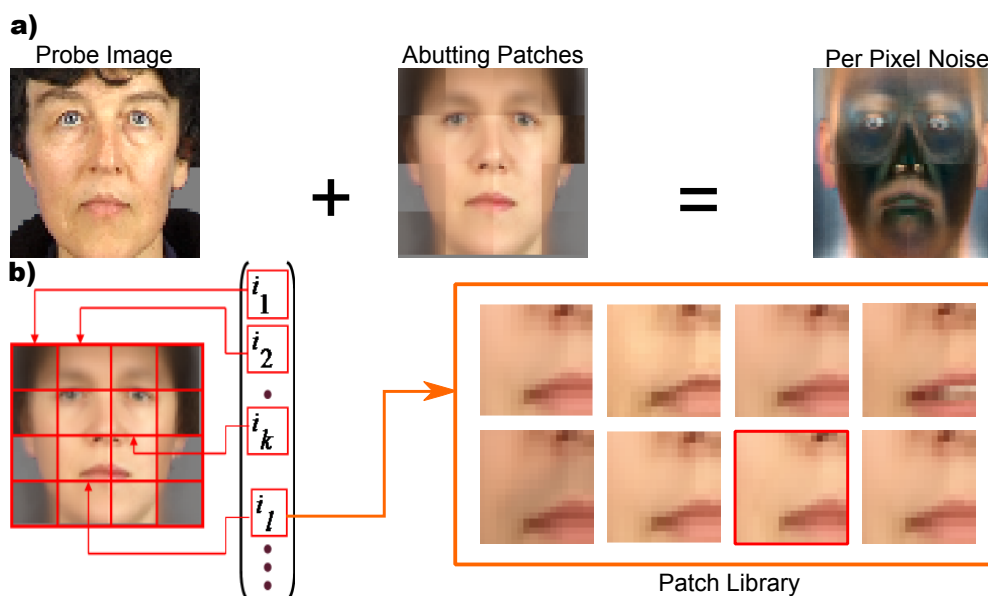


Figure 6.1: Patch-based representation for face recognition: (a) The face image is treated as a composite picture made up of non-overlapping smaller patches (a mosaicface). Each patch is taken from a fixed library. Observed image data consists of the library patches plus additive noise sampled from a per-pixel Gaussian noise model (top right indicates noise amplitude). (b) The image representation now consists of a list of indices to the library. Distance-based methods are inadequate here as there is no easy way to define distance between two such lists.

training stage. The face is now represented by a list of indices to the library (see figure 6.1).

Unfortunately, this representation poses a problem: there is no obvious way to define a distance metric between two such lists. To resolve this problem, we adopt the approach of [140] who applied a probabilistic framework to face recognition. They described each face as generated from an underlying hidden variable (a Latent Identity Variable or LIV) in the presence of additive noise. One noticeable aspect of their model is that the decision was not based on distances, and is hence suitable for our purposes.

In section 6.3 we describe the LIV approach of [140]. In section 6.4 we describe the basic mosaicface model. Results of face identification experiments on the XM2VTS database [3] and comparison to current methods are presented in section 6.4.2. In section 6.5 we extend this method to allow the algorithm to associate patches with multiple different appearances. For example, it might equate an open and blinking eye if they seem to be the *same* eye. We demonstrate the efficacy of this approach using a portion of the XM2VTS database with large lighting variations in section 6.5.2. We also test our method in the existence of expression variation and large image variation on FERET database in sections 6.5.3 and 6.5.4 respectively, before drawing conclusions in section 6.6.

6.3 Latent Identity Variables (LIVs)

We employ the probabilistic recognition framework used in [140]. In this framework the data \mathbf{x}_{ij} are generated from lower dimensional latent variables \mathbf{h}_i . These variables are considered to represent identity and are called Latent Identity Variables (LIV). Each observed data point is generated by:

$$\mathbf{x}_{ij} = f(\mathbf{h}_i, \theta) + \epsilon_{ij}, \quad (6.1)$$

where \mathbf{x}_{ij} is the j^{th} image of the i^{th} individual and $f(\mathbf{h}_i, \theta)$ is a function of the identity variables \mathbf{h}_i . The term θ contains the parameters of the model and ϵ_{ij} is an additive Gaussian noise term which explains any remaining variance and has a covariance Σ . To complete the model we need to define the prior probability distribution for latent variables denoted by $\Pr(\mathbf{h}_i)$. The probability distribution of the data conditioned on the latent identity variables \mathbf{h}_i is thus:

$$\Pr(\mathbf{x}_{ij}|\mathbf{h}_i) = \mathcal{G}[f(\mathbf{h}_i, \theta), \Sigma] \quad (6.2)$$

where $\mathcal{G}[\mathbf{a}, \mathbf{B}]$ is a Gaussian with mean \mathbf{a} and covariance \mathbf{B} . There have been various examples based on this framework such as using tied factor analysis (TFA) [141] and probabilistic linear discriminant analysis (PLDA)[139].

Learning: We aim to learn the parameters θ and Σ given data \mathbf{x}_{ij} . It would be easy to estimate these parameters if we knew \mathbf{h}_i . Likewise if we knew the parameters θ and Σ , it would be easy to infer the latent variables \mathbf{h}_i . This type of problem is well suited to the Expectation-Maximization (EM) algorithm [40] which iteratively maximizes

$$Q(\Theta_t, \Theta_{t-1}) = \sum_{i=1}^I \sum_{j=1}^J \int \Pr(\mathbf{h}_i|\mathbf{x}_{i1\dots iJ}, \Theta_{t-1}) \log[\Pr(\mathbf{x}_{ij}, \mathbf{h}_i)] d\mathbf{h}_i \quad (6.3)$$

where t is the iteration index and $\Theta = \{\theta, \Sigma\}$. In particular we alternate between (i) estimating a full posterior over the latent variables \mathbf{h}_i (Expectation or E-Step) and (ii) maximizing Equation 6.3 with respect to Θ_t (Maximization or M-Step). In the E-step, we fix the parameters $\Theta = \{\theta, \Sigma\}$ and calculate a posterior probability distribution over all the latent variables \mathbf{h}_i given data \mathbf{x}_{ij} . By Bayes' rule:

$$\Pr(\mathbf{h}_i|\mathbf{x}_{i1\dots iJ}, \Theta) = \frac{\prod_{j=1}^J \Pr(\mathbf{x}_{ij}|\mathbf{h}_i, \Theta) \Pr(\mathbf{h}_i)}{\Pr(\mathbf{x}_{i1\dots iJ}|\Theta)} \quad (6.4)$$

where we have assumed that each image of the same person was generated independently. Notice that we constrain the EM algorithm so that if two training samples belong to the same individual they are forced to have the same latent variable. In the parlance of the learning community, we are employing “equivalence constraints” [155]. In the M-Step, we maximize

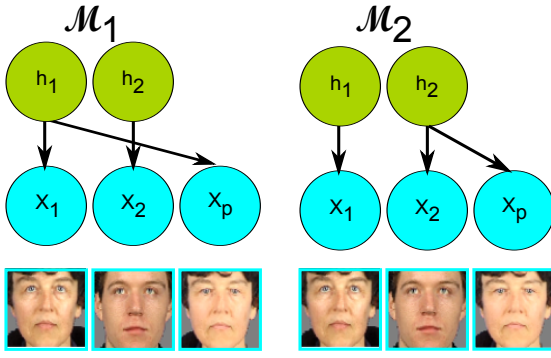


Figure 6.2: Face identification by model comparison: Standard LIV model. Observed gallery faces $\mathbf{x}_1, \mathbf{x}_2$ and observed probe \mathbf{x}_p are explained by underlying identity variables \mathbf{h} . In model \mathcal{M}_1 the probe \mathbf{x}_p matches gallery face \mathbf{x}_1 and hence they share identity variable \mathbf{h}_1 . In model \mathcal{M}_2 the probe \mathbf{x}_p matches gallery face \mathbf{x}_2 and these share an identity variable \mathbf{h}_2 .

the criteria in Equation 6.3 with respect to parameters $\Theta = \{\theta, \Sigma\}$ by taking the derivative and setting it to zero.

Recognition: In recognition, we compare the likelihood of the data under different models. Each model assigns identity variables to explain the observed faces in a different way. If the current model hypothesis is that two faces belong to the same person, they share the same identity variable \mathbf{h}_i . If not, then they have different identity variables. To calculate the likelihood of each of these models we need to know the values of the constituent identity variables \mathbf{h}_i . Unfortunately, noise in the generation process means we can never know the exact values of these variables. Therefore we marginalize (integrate out) these variables. The inference process therefore assesses the probability that two faces had a common identity cause \mathbf{h}_i , without ever committing to exactly what that cause was.

Figure 6.2 shows the models for face identification with two gallery images $\mathbf{x}_1, \mathbf{x}_2$ and a probe image \mathbf{x}_p . Model \mathcal{M}_1 associates the probe image with the first image in the gallery and Model \mathcal{M}_2 considers the probe image as matching to the second image in the gallery. The likelihood of the data under these two models are

$$Pr(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_p | \mathcal{M}_1) = \int Pr(\mathbf{x}_1, \mathbf{x}_p | \mathbf{h}_1) Pr(\mathbf{h}_1) d\mathbf{h}_1 \int Pr(\mathbf{x}_2 | \mathbf{h}_2) Pr(\mathbf{h}_2) d\mathbf{h}_2 \quad (6.5)$$

$$Pr(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_p | \mathcal{M}_2) = \int Pr(\mathbf{x}_1 | \mathbf{h}_1) Pr(\mathbf{h}_1) d\mathbf{h}_1 \int Pr(\mathbf{x}_2, \mathbf{x}_p | \mathbf{h}_2) Pr(\mathbf{h}_2) d\mathbf{h}_2$$

Note that both likelihood and prior terms in these expressions were part of the original model definition. We combine these models, with suitable priors and calculate a posterior probability for the model (and hence the match) using Bayes' rule:

$$Pr(\mathcal{M}_k | \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_p) = \frac{Pr(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_p | \mathcal{M}_k) Pr(\mathcal{M}_k)}{\sum_{l=1}^N Pr(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_p | \mathcal{M}_l) Pr(\mathcal{M}_l)} \quad (6.6)$$

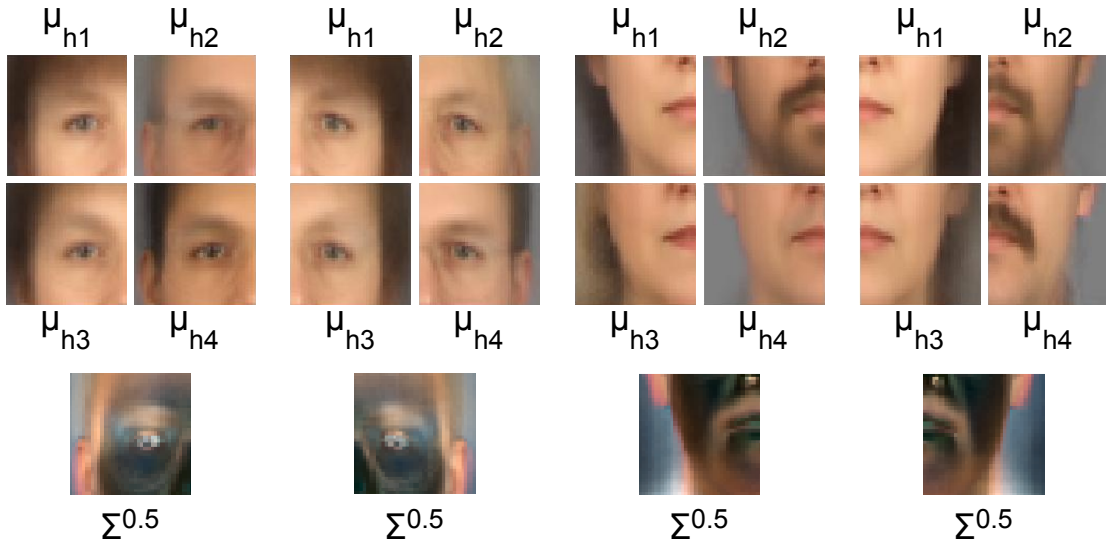


Figure 6.3: Learnt patch library. For each position in the image we learn a library containing several patches (here 4). At every position there is a covariance representing how much the training images deviate from the library. In this case the covariance is axis-oriented Gaussian noise, with amplitude as depicted.

6.4 Mosaicface Model

In this section we present our patch based LIV model. We divide the image into patches, and learn a separate model for each patch. These patch models are assumed to be independent so we take the product of patch likelihoods to calculate the overall likelihood in Equation 6.6. With this in mind, we now describe the model for one patch. Each patch is generated by:

$$\mathbf{x}_{ij} = \mu_{h_i} + \epsilon_{ij} \quad (6.7)$$

where \mathbf{x}_{ij} denotes the pixel intensities of the patch belonging to the j^{th} sample of the i^{th} individual. Equation 6.7 states that the observed data \mathbf{x}_{ij} is generated by taking the h_i^{th} library patch μ_{h_i} and adding the zero mean Gaussian noise term ϵ_{ij} with covariance Σ_{h_i} . Notice that instead of a continuous representation of identity, the variable h_i is discrete and can take values $h_i \in \{1..L\}$ where L is the number of the library items.

More formally, we can write the conditional probability of the data sample \mathbf{x}_{ij} given the identity variable h_i as

$$Pr(\mathbf{x}_{ij}|h_i) = \mathcal{G}[\mu_{h_i}, \Sigma_{h_i}] \quad (6.8)$$

$$Pr(h_i = l) = f_l \quad (6.9)$$

where f_l is the prior probability of choosing the l^{th} patch.

Learning: We aim to learn the model parameters $\Theta = \{\mu_{1..L}, \Sigma_{1..L}, f_{1..L}\}$ corresponding to the library patches, associated uncertainties and prior probabilities. In the E-Step we calculate

the posterior distribution $Pr(\mathbf{h}_i | \mathbf{x}_{i1...iJ}, \Theta^{t-1})$ over the latent identity variables \mathbf{h}_i given all the samples of a person $\mathbf{x}_{i1...iJ}$ as in Equation 6.4.

In the M-Step we update the parameters of the model by maximizing the criteria in Equation 6.3 with respect to each of $\Theta = \{\mu_{1..L}, \Sigma_{1..L}, f_{1..L}\}$. Taking derivatives with respect to the components of Θ , equating to zero and re-arranging gives the following update equations.

$$\begin{aligned}\mu_l &= \frac{\sum_{i=1}^I \sum_{j=1}^J Pr(h_i = l | \mathbf{x}_{ij}) \mathbf{x}_{ij}}{\sum_{i=1}^I \sum_{j=1}^J Pr(h_i = l | \mathbf{x}_{ij})} \\ \Sigma_l &= \frac{\sum_{i=1}^I \sum_{j=1}^J Pr(h_i = l | \mathbf{x}_{ij}) (\mathbf{x}_{ij} - \mu_l)(\mathbf{x}_{ij} - \mu_l)^T}{\sum_{i=1}^I \sum_{j=1}^J Pr(h_i = l | \mathbf{x}_{ij})} \\ f_l &= \frac{\sum_{i=1}^I \sum_{j=1}^J Pr(h_i = l | \mathbf{x}_{ij})}{\sum_{l=1}^L \sum_{i=1}^I \sum_{j=1}^J Pr(h_i = l | \mathbf{x}_{ij})}\end{aligned}\quad (6.10)$$

where the posterior probabilities $Pr(h_i = l | \mathbf{x}_{ij})$ were calculated in the E-Step using the previous parameters Θ^{t-1} . In terms of mixture of Gaussians, the parameters in Equation 6.10 represent the parameters of the L mixture components. In particular, the terms μ_l and Σ_l represent the mean and the covariance of the l^{th} Gaussian component and the term f_l denotes the mixture weight (i.e. the prior probability of the l^{th} Gaussian component).

Examples of the learnt library contents for a small scale model are depicted in figure 6.3.

6.4.1 Experiment 1 - Mosaicface Identification

In Experiment 1 we investigate face identification with the mosaicface model on the XM2VTS database [3]. Initially, we perform only minimal preprocessing. Faces are segmented from the background using an iterative graph-cuts procedure. An affine warp is used to register the images to a standard template using three hand-marked keypoints. The final image size was $70 \times 70 \times 3$ and the raw pixels were used as input. No photometric normalization occurred. The training and test images are non-overlapping and the gallery and probe images are from different sessions (the first and last respectively). These characteristics mean that recognition performance will never be high, but it is easy to compare the relative inferential power of algorithms. We divided the data into a training set of 195 individuals and a test set of 100 individuals. The training individuals each have 8 sample images. The test set contains 100 gallery images and 100 matching probe images taken from the first and last recording sessions respectively. We train models with grid resolutions 4×4 , 8×8 , 16×16 , 20×20 and 32×32 patches. For each resolution, we train libraries containing $L=2,4,8$ and 16 different patch images. We restrict the full model so that the covariances $\Sigma_{1..L}$ are axis oriented (diagonal) and the same for all patches. We use 20 iterations of the EM algorithm to learn each model, initializing all parameters to random values.

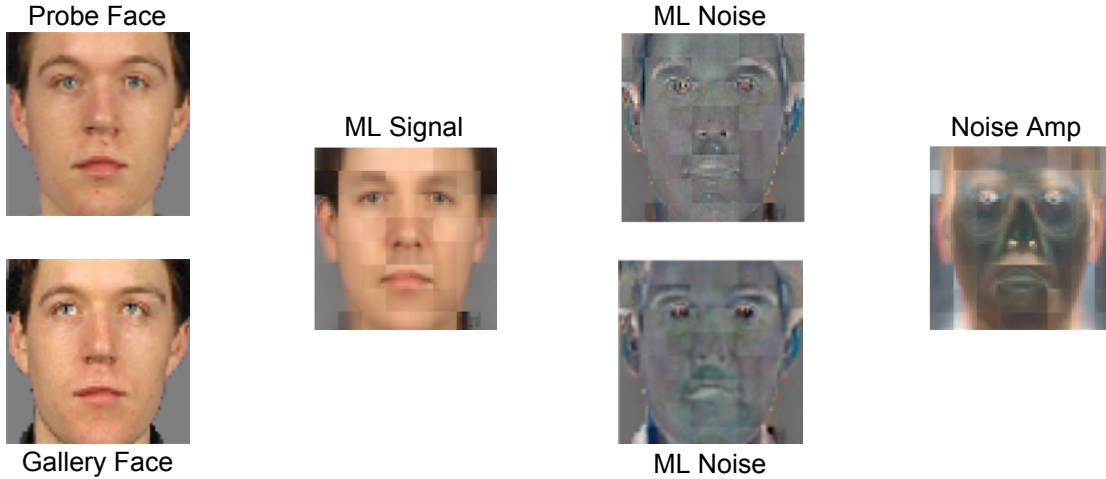


Figure 6.4: *Maximum likelihood (ML) reconstruction of probe and gallery images:* The ML signal consists of the patches that are most likely to be responsible for the observed images. The ML noise is the difference between the images and the ML signal. We model this as being drawn from a per-pixel Gaussian distribution with amplitude as shown on the right.

The mosaicface model is tested on 100 different probe images. In recognition patches are assumed to be independent. Hence the total likelihood of the image is given by

$$Pr(X^{1..K}|\mathcal{M}_n) = \prod_{k=1}^K Pr(X^k|\mathcal{M}_n) \quad (6.11)$$

where $Pr(X^k|\mathcal{M}_n)$ is the likelihood of the k 'th patch under this model. Each of these individual patch likelihoods were calculated by marginalizing over the identity variable h as in Equation 6.6. In this case h is a discrete variable, so the integrals become sums. We combine these likelihoods with uniform priors and calculate a posterior probability for the match using Equation 6.6. We consider the recognition to be correct when the maximum a posteriori (MAP) model matches the probe image with the correct gallery image. In recognition we are implicitly asking whether probe and gallery images had the same underlying cause (library patch). We actually considered all possible values of the library patch when we marginalized over the uncertainty in the identity variable $h = \{1 \dots L\}$ that represents the index to the library. However, for the purposes of visualization, we display the most probable cause (library patch) at each position in the image. Figure 6.4 shows an example of this visualization. The probe and gallery faces are both approximated by a common cause, but the noise differs in each case - this explains the difference between the observed images.

In figure 6.5a we plot the percentage of correct matches as a function of the $n \times n$ resolution of the patch grid and the number of items in the library for each patch. Recognition performance depends on the number of library items with a peak at 16 items. The results also demonstrate

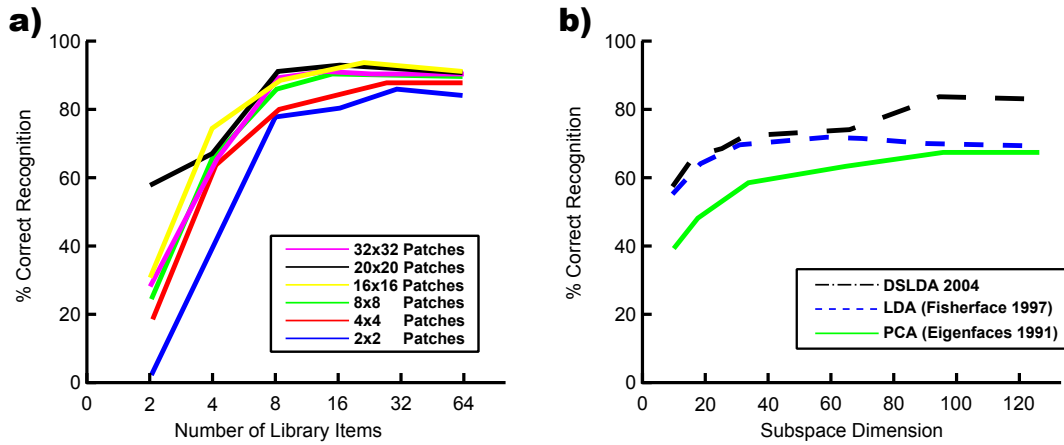


Figure 6.5: Results on XM2VTS database: (a) % correct closed-set identification performance for mosaicface model on XM2VTS database as a function of grid resolution (abscissa) and number of library items L (different curves). (b) % correct closed-set identification performance on XM2VTS database for PCA [161], LDA [14] and dual space LDA [167] algorithms.

a performance change as the image is divided into a finer resolution grid, reaching a peak with 20×20 patches. Peak overall observed performance was 92% correct recognition with a grid resolution of 16×16 patches and learning $L=16$ or $L=32$ library items for each patch. The same performance was observed with 20×20 patches and $L=8$ or $L=16$ library items per patch.

To establish exactly how powerful our technique is, we plot percentage of correct matches for the same task for a number of distance-based models in figure 6.5b. In each case, we show performance as a function of the dimension of the subspace. For LDA models, the dimension of both the within- and between-individual subspaces were set to this value. None of these algorithms approaches the performance of mosaicfaces. The best result is for dualspace LDA [167] with 84% recognition. We conclude that despite only having a discrete representation of the face, the mosaicfaces algorithm can produce competitive inferential power, when datasets and the experimental protocol are exactly matched.

6.4.2 Experiment 2 - Mosaicfaces with Gabor Features

Raw luminance values are not ideal inputs for face recognition. In order to improve performance, we pre-processed the image by filtering with a bank of 24 Gabor filters (4 orientations \times 3 scales \times 2 phases). We retained only responses at the center of each image grid position, so that the pixel data from each original grid position was replaced by a vector of length 24 regardless of the number of pixels in the original patch. Each vector of Gabor responses was normalized to have length one. We used orientations of $0, \pi/2, \pi, 3\pi/2$. The angular frequencies of the sinusoidal components were $1/4, 1/8$ and $1/16$ for spherical envelopes of standard deviation 2, 4 and 8 pixels respectively.

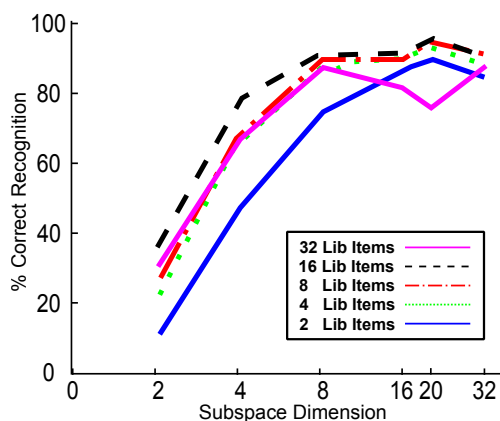


Figure 6.6: Results using Gabor features % correct first match identification performance for mosaicface model on Gabor-filtered XM2VTS database as a function of grid resolution (abscissa) and number of library items (different curves). Note slight performance increase compared to figure 6.5.

In figure 6.6 we plot the percentage of correct matches as a function of the $n \times n$ resolution of the patch grid and the number of items in the library for each patch, but now using the Gabor filtered data. Recognition performance follows the same general pattern as in figure 6.5, with a peak performance near 16 library items and a 20×20 grid resolution. However, peak performance was now increased from 92% with pixel data to 96% with Gabor responses. This was achieved with 16 library items and a grid size of 20×20 . It is possible that a denser sampling of Gabor filter data in terms of position, scale and orientation would increase performance further.

6.5 Mosaicface Model with Multiple Patch Appearances

The mosaicface representation produces promising results, but like many algorithms, it cannot cope well with very non-linear changes in the image. For example, if someone smiles, the relevant image patches may completely change their appearance. Other algorithms such as LDA [14] deal with this problem by systematically downweighting parts of the image that vary in this way. However, this discards valuable information: the smile tells us something about the neutral expression and vice-versa. In this section we present a system that attempts to exploit this information to improve recognition performance.

In order to do this, we introduce a second set of latent variables into our model, which lie between the latent identity variables h_i and the images \mathbf{x}_{ij} (see figure 6.7). We term these latent appearance variables, and denote them by a_{ij} . In the learning phase we now learn multiple patches associated with the same identity $h_i = l$. The appearance variable a_{ij} (which is discrete) determines which of these patches we see in a particular image. However, in recognition, the appearances are all treated as equivalent so we can match a smiling to a neutral face. In terms of conditional probabilities, we write:

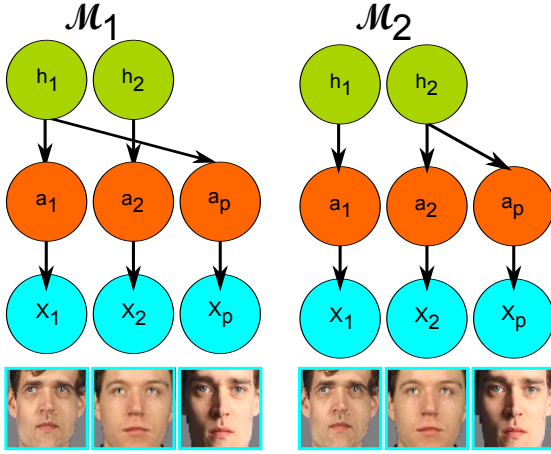


Figure 6.7: Face identification by model comparison: Appearance Model There is now a second latent variable $\mathbf{a}_{1,2,p}$ associated with each data point $\mathbf{x}_{1,2,p}$ that modulates the appearance of the observed data.

$$Pr(\mathbf{x}_{ij}|h_i = l, a_{ij} = m) = \mathcal{G}[\mu_{l,m}, \Sigma_{l,m}] \quad (6.12)$$

$$Pr(h_i = l, a_{ij} = m) = f_{lm} \quad (6.13)$$

Notice that the Gaussian that describes the image patch is now indexed by both the identity and appearance variables. Once more, however, the identity variable remains the same for all images of the same person (it is only indexed by i). However, the appearance variable can take different values for every image (it is indexed by i and j).

6.5.1 Learning Multiple Appearances

E-Step: In the E-Step, we use all of the images $x_{i1\dots iJ}$ of the i 'th individual to calculate joint posterior distribution over all the latent variables pertaining to these images (h_i and $a_{i\dots j}$). In order to do this, we decompose this joint posterior into:

$$Pr(h_i, a_{i1\dots iJ}|\mathbf{x}_{i1\dots iJ}) = \prod_{j=1}^J Pr(a_{ij}|h_i, \mathbf{x}_{ij})Pr(h_i|\mathbf{x}_{i1\dots iJ}) \quad (6.14)$$

We treat each of these components in turn. By Bayes rule, the terms in the product can be calculated as:

$$Pr(a_{ij}|h_i, \mathbf{x}_{ij}) = \frac{Pr(\mathbf{x}_{ij}|a_{ij}, h_i)Pr(a_{ij}|h_i)}{\sum_{a_{ij}} Pr(\mathbf{x}_{ij}|a_{ij}, h_i)Pr(a_{ij}|h_i)} \quad (6.15)$$

where $\sum_{a_{ij}}$ indicates summing over all the discrete values $a_{ij} = m$ of the appearance variable. the likelihood term comes from Equation 6.12 and the prior term comes from the discrete distribution in Equation 6.13. The second term in Equation 6.14 depends on all of the images for that individual and is also calculated by Bayes rule:

$$\begin{aligned} Pr(h_i|\mathbf{x}_{i1} \dots \mathbf{x}_{iJ}) &= \frac{\prod_{j=1}^J Pr(\mathbf{x}_{ij}|h_i)Pr(h_i)}{\sum_{h_i} \prod_{j=1}^J Pr(\mathbf{x}_{ij}|h_i)Pr(h_i)} \\ &= \frac{\prod_j \sum_{a_{ij}} Pr(\mathbf{x}_{ij}, a_{ij}|h_i)Pr(h_i)}{\sum_{h_i} \prod_{j=1}^J \sum_{a_{ij}} Pr(\mathbf{x}_{ij}, a_{ij}|h_i)Pr(h_i)} \end{aligned} \quad (6.16)$$

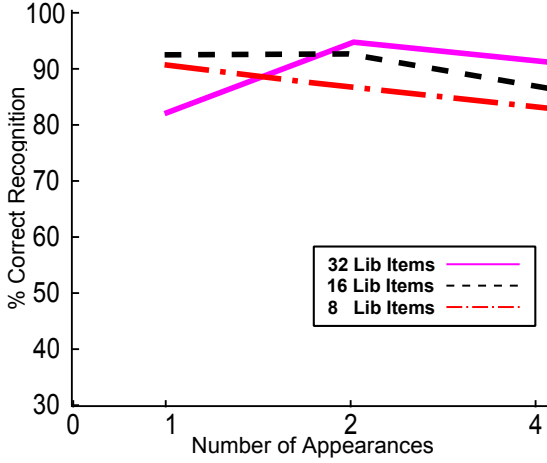


Figure 6.8: Results of multiple-appearance model on XM2VTS database % correct identification performance for XM2VTS database as a function of the number of appearances in the model and number of library items. Patch grid resolution was held constant at 16×16 . Extending the standard model by adding more appearances does not improve results.

where \sum_{h_i} indicates summing over all possible values $h_i = l$ of the discrete identity variable. The likelihood of the data given the identity variable is calculated by marginalizing the joint likelihood with respect to the identity variable. The components of this can be found in Equations 6.12 and 6.14.

M-Step: In the M-Step, we update the model parameters $\Theta = \{f_{lm}, \mu_{lm}, \Sigma_{lm}\}$ by maximizing the equivalent expression to Equation 6.3. Taking derivatives with respect to these parameters, setting to zero and re-arranging yields the following update rules:

$$\begin{aligned} \mu_{lm}^{[t+1]} &= \frac{\sum_{i=1, j=1}^{i=I, j=J} Pr(h_i = l, a_{ij} = m | \mathbf{x}_{ij}, \Theta^{[t]}) \mathbf{x}_{ij}}{\sum_{i=1, j=1}^{i=I, j=J} Pr(h_i = l, a_{ij} = m | \mathbf{x}_{ij}, \Theta^{[t]})} \\ \Sigma_{lm}^{[t+1]} &= \frac{\sum_{i=1, j=1}^{i=I, j=J} Pr(h_i = l, a_{ij} = m | \mathbf{x}_{ij}, \Theta^{[t]}) (\mathbf{x}_{ij} - \mu_{lm}^{[t]})^2}{\sum_{i=1, j=1}^{i=I, j=J} Pr(h_i = l, a_{ij} = m | \mathbf{x}_{ij}, \Theta^{[t]})} \\ f_{lm}^{[t+1]} &= \frac{\sum_{i=1, j=1}^{i=I, j=J} Pr(a_{ij} = m | h_i = l, \mathbf{x}_{ij}) Pr(h_i = l | \mathbf{x}_{ij})}{\sum_{h_i} \sum_{a_{ij}} Pr(h_i, a_{ij})} \end{aligned} \quad (6.17)$$

Recognition with Multiple Appearances

As before we pose recognition as model comparison (see figure 6.7). Now there are two latent variables (identity and appearance) and we must marginalize over both to calculate the data likelihoods for Equation 6.6. For example, the likelihood that gallery image \mathbf{x}_1 and probe image \mathbf{x}_p were generated from the same identity variable \mathbf{h}_1 is given by:

$$Pr(\mathbf{x}_{1,p}) = \sum_{h_1} \left[\sum_{a_1} Pr(x_1, h_1, a_1) \sum_{a_p} Pr(x_p, h_1, a_p) \right] \quad (6.18)$$

The order of summation has the following interpretation. We are calculating the likelihood that the data points have the same identity *regardless* of the appearance of each. Similarly to section 6.3 we combine these likelihoods, with suitable priors and calculate a posterior probability for the match using Equation 6.6.

Experiment 3 replicates exactly the protocol of Experiment 2. However, we now train and test with 1, 2 or 4 appearances for each library item, depending on the condition. The condition with 1 appearance is identical to the original mosaicface model from Experiment 2. To simplify matters we restrict our comparison to a single grid size (16×16). In figure 6.8 we examine the effect of using multiple appearances for the frontal XM2VTS data as a function of the number of items in the library. For the range of parameters chosen, peak performance was 94% and was achieved with 2 appearances and 32 library items. The results do not demonstrate a clear advantage to using multiple appearances. We conjecture that the variation between probe and gallery images is small and attributable to causes such as weak registration process, small lighting, pose and illumination effects. With our limited training set, it is not possible to comprehensively learn about these variations, although our model is theoretically capable of this.

6.5.2 Experiment 3 - Modeling Illumination Change

A more suitable test for the multiple appearance model is to recognize faces across large changes in viewing conditions. To this end, we consider matching using the lighting subset of the XM2VTS database. In particular, we use images of 295 individuals viewed across four different sessions in two different lighting conditions (frontal and left-side). All data was pre-processed as in Experiments 2 and 3. Once more the first 195 individuals were used for training and the last 100 used for testing identification performance. Gallery and probe sets were taken from the first and last recording session respectively. All gallery data was front-lit whereas all probe data was side-lit. However, neither training nor identification algorithms were provided information concerning the particular lighting condition in any image. As in experiment 2 we learnt models with 1, 2 and 4 appearances, but used only one grid resolution (16×16). In figure 6.9, we show an example gallery / probe pair, with the relevant library patches, learnt with 2 different appearances. Note, in each of the ML signal images, the model chose a patch from the library with the appearance that correctly matches each of the probe or the gallery images. For example, a front-lit appearance patch was chosen for the gallery image, whereas a side-lit patch was chosen for the probe image.

In figure 6.10a we plot % first match identification performance of our algorithm as a function of the number of appearance components in the model and the number of identity items in the library (compare to figure 6.8). Using data with large illumination changes there is a clear change in pattern. With only a single appearance for each item in the library (the original mosaicface model), performance is very poor. For example, with 16 library items only 47% performance was achieved (compared to 92% for data without large illumination

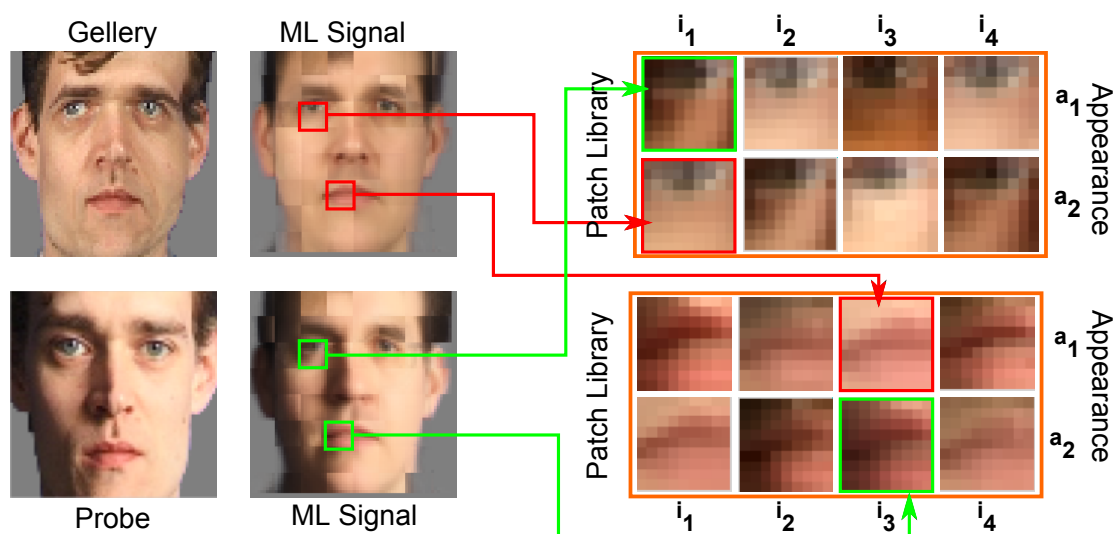


Figure 6.9: Multiple appearance model: illumination variation Each patch can now take multiple appearances that are treated equivalently in recognition. In this case the library (right) consists of four different identities, each of which has two possible appearances. This allows us to equate the front-lit face (top) with the side-lit face (bottom).

differences). However, when we add a second appearance component to the model we get a peak performance of 92% (compared to a peak of 94% for data without large illumination differences). With four appearance components there is a slight performance decrease (note there are genuinely only two difference viewing conditions), but results are still far superior to the original mosaicface model. We conclude that for this situation the multiple appearances model copes well with the changes in image illumination. To establish the relative inferential power of our algorithm we also measured performance for exactly the same dataset with standard distance-based inference methods. We concatenate all of the Gabor responses from all grid positions to make one large image vector and use this $16 \times 16 \times 24$ data vector as the input data representation. In figure 6.10b we plot results of several comparable distance-based algorithms for illumination data (compare to figure 6.5b). The large illumination variation is sufficient to severely limit recognition performance, which is comparable to that for the original (single appearance) mosaicface model. It is notable that the linear discriminant analysis based method does not cope well with the large lighting variation despite exploiting knowledge about within-individual variation: this algorithm downweights information about identity that is in the same subspace as the lighting change. In contrast, our algorithm exploits this information.

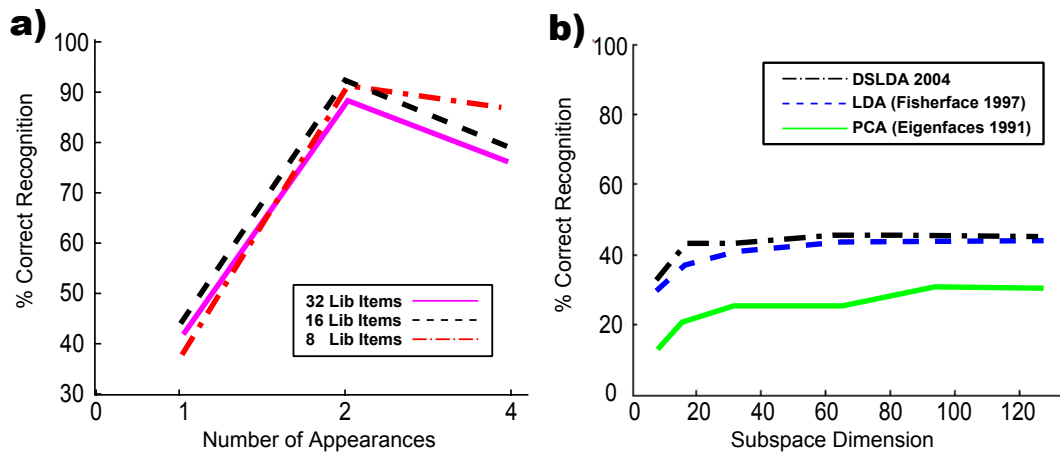


Figure 6.10: Results on XM2VTS Lighting database (a) multiple-appearance model: % correct classification as a function of the number of patch appearances and library items, using a 16×16 patch-grid. The single appearance model performs poorly. Adding a second appearance significantly improves the performance. (b) distance based methods: % correct classification as a function of subspace dimension. Performance is significantly retarded relative to when there is no illumination variation (figure 6.5b)

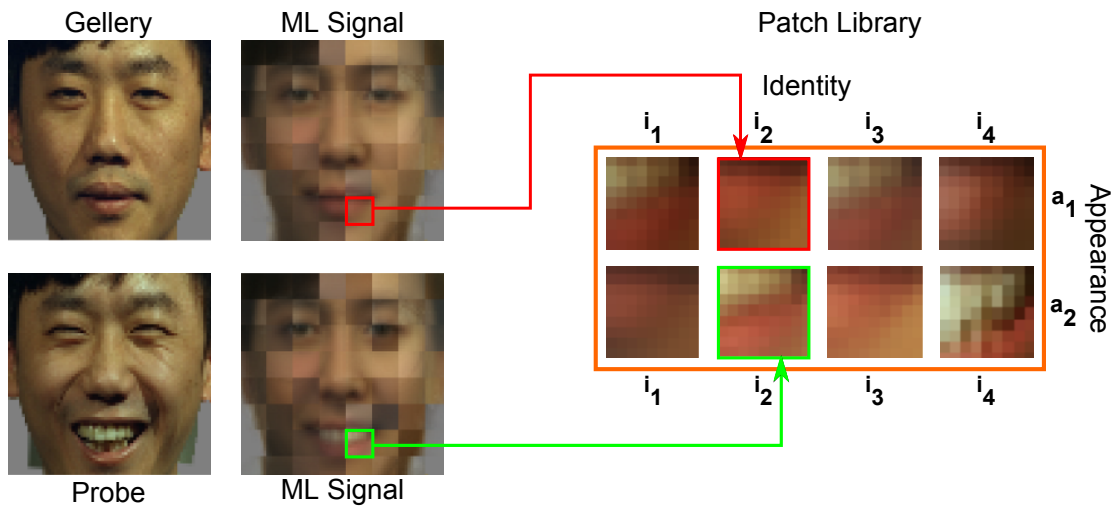


Figure 6.11: Multiple appearance model: expression variation. Each patch can now take multiple appearances that are treated equivalently in recognition. In this case the library (right) consists of four different identities, each of which has two possible appearances. This allows us to equate the neutral face (top) with the smiling face (bottom).

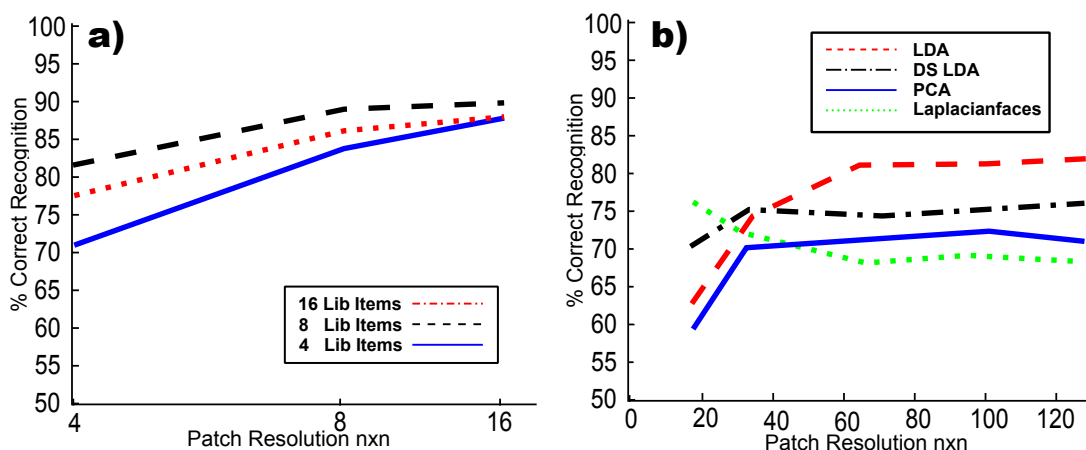


Figure 6.12: Results of multiple-appearance model on FERET database (a) The results of mosaicface appearance model on FERET database fa/fb subset. Once more performance improves with increasing patch resolution and size of the learnt library. Peak performance is at 91% correct. (b) Performance on same task for PCA[161], LDA [14], Laplacianfaces [76] and dual space LDA [167] algorithms.

6.5.3 Experiment 4 - Expression Variation

The multiple-appearance model improved performance in the XM2VTS database which only contains minor variability compared to real-world images. In Experiment 3, we examine performance using the fa/fb subset of the FERET database which contains significant expression variation. We divide the database into a training set of 400 individuals and a test set of 100 individuals. The training individuals have two sample images (one smiling and one neutral) which are used to learn the mosaicface appearance model. The test set contains 100 gallery (neutral expression) and 100 probe (smiling) images. We learn models with two appearances to account for the two facial expressions in the dataset. Figure 6.11 shows examples of probe/gallery images with library patches learnt with 2 appearances.

In figure 6.12a we plot percent correct recognition performance as a function of the ($n \times n$) patch grid resolution and the number of items in the library for the FERET data. Peak performance for this dataset was 91% correct recognition with 16×16 patch grid resolution, $L=8$ library patches and $M=2$ latent appearance variables. In figure 6.12b we plot performance for four contemporary algorithms as a function of the subspace dimension. The peak performance was for the LDA method with 82% correct. We had conjectured that the mosaicface model would show a relative improvement in this case as it is designed to cope with the nonlinear expression variation. While we still achieve better results, there is little evidence that the gap has widened. One possible reason for this is that all models have just concluded that the mouth region is unreliable. In the mosaicface model, this would occur if the library variances in this part of the image were larger so the contribution to the overall likelihood was small. In LDA

models this would occur if the mouth pixels were downweighted in all of the basis functions. We may have inadvertently assessed the ability of these algorithms to *ignore* the mouth region rather than to *exploit* it.

6.5.4 Experiment 5 - Large Image Variation

In this experiment we cropped out a 40×24 mouth region from the original 70×70 image for all of the training and test data in Experiment 4. Now each algorithm is forced to exploit the mouth region. We repeated experiment 4 with these smaller regions. For the mosaicface model, the peak performance was 33% correct with 8×8 grid resolution, $L = 16$ library patches and $M=2$ appearances. For the competing approaches peak performance was for the LDA algorithm [14] which yielded to 25% correct recognition with a subspace dimension of 128. The mosaicfaces algorithm does extract information from highly varying regions more successfully than competing methods.

6.6 Summary and Discussion

We have introduced a discrete patch-based representation for face recognition and demonstrated good recognition performance. For exactly matched tasks, the mosaicfaces algorithm achieved higher performance than several contemporary face recognition algorithms, one of which [167] has been demonstrated to provide state of the art performance. As such, we have made progress towards our first research objective of finding a representation that is applicable to many object-related tasks.

Our algorithm is not the first face recognition method to use patches. However, to the best of our knowledge, it is the first to use an entirely discrete representation of face images. Most previous work (e.g. [118]) that extracts patches use conventional distance based methods on the (still continuous) patch data. In other areas of computer vision, discrete patch-based representations have been learnt from training data (e.g. [104, 135]). However, the task of identity recognition has not been addressed.

We extended the mosaicface model to account for multiple possible appearances and demonstrate that this improves results. This extension provides a truly multi-modal probabilistic representation of a given individual. Interestingly, the elastic bunch graph matching algorithm of Wiskott et al. [173] used a very similar multi-modal representation to help register images for face recognition. However, after registration, this representation was abandoned and a conventional distance measure was used to support recognition.

We demonstrate that the multi-appearance approach supports better recognition performance than LDA based methods [14, 167] in the presence of large illumination differences.

Our method explicitly learns about the equivalence of image data under different illuminations. However, the LDA based methods down-weight the importance of image differences that linearly co-vary with luminance changes and hence throw out useful information.

The mosaicface model describes the face manifold with a mixture of Gaussians, which is an extremely general model: with large amounts of training data we could potentially learn a detailed model of the non-linear variations in face images due to both signal (identity) and noise (illumination, expression and pose). This is a promising approach for recognition but caution should be applied in interpreting our results: although we produce high performance in the presence of illumination variation, it should be emphasized that we observed the *same* illumination variation in both training and test. It is not known how performance would generalize to previously unseen viewing conditions.

Since we originally carried out this work, Fu and Prince [66] have developed a multi-scale PLDA model, which investigates the effect of the spatial support of signal and noise basis function in frontal face recognition. Their model is similar to mosaicfaces in that they represent face images as a set of non-overlapping patches. However, the latent variables used in their model are continuous as opposed to the discrete identity variables used in mosaicfaces. They conclude that performance is improved when signal is modeled locally and noise is modeled globally. They report state of the art performance on XM2VTS and Yale databases, in the presence of illumination and expression variation. Note, the multi-scale idea proposed by Fu and Prince can be combined with our mosaicfaces model with relative ease, to improve performance.

Our algorithm uses a compact representation of faces (we need only store the likelihood that each patch was generated from each library element), and most of these will be near to zero. To consider a new face, we must calculate all of these likelihoods. This operation is comparable in computational demand to projection to feature space in distance-based methods. We have not yet exploited one advantage of our mosaicface model which is to learn individual covariances for each patch. This effectively means that estimates of the within-individual noise vary depending on the position in image space, and may yield increased performance. A variational version in which we allow uncertainty in the library contents may also improve results.

The mosaicface model developed in this chapter, shares the core properties of all previous models developed in this thesis: (i) it represents images as a set of non-overlapping grid of patches and (ii) there is a library of patches which is used to approximate a test images. However, there are some fundamental differences. Most significant difference is that, in the mosaicface model we learn the library patches from a set of training examples using a Mixture of Gaussian model. In addition, the library images are not registered to the test image using

cross-correlation as we did in our segmentation model. Moreover, we have learned the equivalences of different patches in the mosaicface model. This means if two training samples belong to the same individual they are forced to have the same latent variable. Such “equivalence constraints” were not used in our previous models.

One of the limitations of our model is that it requires the faces to be registered to a common template. Hence, it may not work very well on faces with varying poses, as it is difficult to register such faces. Since this work, Li and Prince [107] have proposed a method based on probabilistic linear discriminant analysis (PLDA) which integrates face registration and recognition, and demonstrate improved performance in both frontal and cross-pose face recognition.

Unfortunately, our face recognition model was only tested on constrained databases such as FERET and XM2VTS, due to the lack of unconstrained databases at the time. Since this work was done, unconstrained databases such as ‘labeled faces in the wild’ (LFW) [84] was released, which contains face images taken outdoors and in uncontrolled environments. Our model has not been tested on this database, but we suspect that it would not produce cutting edge results without modification.

Chapter 7

Conclusions

We have investigated patch-based object representation for a series of computer vision tasks including: classification, regression, semantic segmentation and identification. Our models were motivated by two limitations of most current models: (i) the lack of extensive testing on large databases of “real world images” and (ii) the lack of a general object representation model that can be applied to a variety of object classes without major alterations. To overcome the first limitation we build models that are capable of handling large databases of images taken in uncontrolled environments, and to resolve the latter we use a patch-based representation and show that it can be successfully used for a variety of vision tasks. We will first summarize the key findings of each chapter in section 7.1. We will then present the main contributions in section 7.2. Finally, we will discuss limitations and future work in sections 7.3 and 7.4 respectively.

7.1 Summary

In chapter 3, we have presented a model for within-category classification. This is the task of predicting attributes that are unique to an object class. For example, we build a model that can predict gender and eyewear in faces, pose in pedestrians and phenotype in cells. Note, the within-category classification task is different from conventional object categorization as it involves distinguishing subtle differences within an object category. The key contributions of the presented model are: (i) it exploits very large available datasets (tens of thousands) unlike some current methods such as SVM which have problems handling large input data. (ii) It achieves state of the art results on gender (in uncontrolled environments) and cell phenotype with 90% and 100% correct classification rates respectively. (iii) It has lower complexity than competing methods such as SVMs. Our model scales linearly with the size of the training and library data in training and the size of the library in testing. (iv) The model currently runs at 2 seconds per image on a standard PC and can be easily coded in a Graphics Processing Unit

(GPU) to achieve a much higher speed. The main findings are as follows: (i) 6×6 pixel patches preserve enough information about object attributes such as gender, pose and phenotype when using 60×60 pixel images. (ii) A Bayesian formulation of the problem results in significant increase in performance compared to the maximum likelihood solution.

We have also presented a model for regression type problems in chapter 4 which we have applied to face pose estimation. The key contributions of this model are as follows: (i) to the best of our knowledge this is the first time patch-based representation has been used for solving a regression problem. (ii) we have provided the first large database of face pose in uncontrolled environments which has received interest in the wider computer vision community. (iii) We achieve promising results (0.88 correlation between the true and the estimated pose). (iv) The encoding of the test images in terms of an index to patch library results in a high data compression. We require 27 times less space to store the image. (v) Our model has much less complexity compared to manifold learning methods such as LLE and Isomap. (vi) The model presented here is a general model which extends naturally to all other image-based regression problems. Future work will include using the model to predict other continuous parameters such as human pose.

In chapter 5, we have presented a model for semantic segmentation of objects, where we use patches to represent class labels. The model extends a recent development in computer graphics known as “shiftmap” editing to be used with patches for semantic object segmentation. We find optimal relative shifts over a library of labelmaps and use it in the form of a prior to predict class labels. The model exploits fast energy minimization algorithms. We present a graph construction for multi-class labeling, that is suitable for various patch sizes. We have tested our model to parse faces and human brains. We provide an iterative algorithm for finding the most likely class label given each test image. Our results demonstrate improved segmentation. The generative aspect of our shiftmap prior model, shows promise for further interesting applications. For example, it can be combined with a set of grammar rules such as those used in [73] to generate new labelmaps.

Finally, we have introduced a discrete representation for face recognition in chapter 6 and demonstrated good recognition performance. For exactly matched tasks, our proposed mosaicfaces algorithm achieved higher performance than several contemporary face recognition algorithms, one of which [167] was demonstrated to provide state of the art performance at the time. To the best of our knowledge, our model is the first to use an entirely discrete representation of face images. Our model describes the face manifold with a mixture of Gaussians, which is an extremely general model: with large amounts of training data we could potentially learn

a detailed model of the non-linear variations in face images due to both signal (identity) and noise (illumination, expression and pose). One of the limitations of our model is that it requires the faces to be registered to a common template.

7.2 Contributions

In this thesis, we have investigated patch-based representation as part of the global problem of finding a general representation model for visual objects. All of the work completed makes progress towards addressing our research objectives.

We have demonstrated that patch-based representation can be successfully used for a variety of vision tasks including: within-category classification [6], regression [5], semantic segmentation and identification [4]. In each of these tasks the patch-based model was used to represent a number of different object classes including faces, cells, pedestrians and human brains. We have achieved promising results for all of these tasks. As such, we have contributed towards our first two research objectives of: finding a representation model such that (i) it can be used across a wide variety of object-related tasks and (ii) it can represent multiple object classes.

Furthermore, we have shown that a patch-based representation model can be easily trained and tested on very large databases with tens of thousands of images, at a reasonable time. Finally, throughout this work we have used images taken in uncontrolled environments and have demonstrated that patch-based representation is robust to many of the challenges described in section 1.2 including: partial occlusion, background clutter, changes in illumination, expression and pose. As such we have made progress towards our remaining two research objectives of: finding a representation model that (iii) can handle large databases of “real world” images and (iv) is robust to challenges in section 1.2.

7.3 Limitations

Fixed Scale Currently, all our models use a fixed size patch to represent objects. We vary the size of the patch for each object class, and find the best size suited to the given task. However, our models do not exploit multiple patch sizes simultaneously. A simple solution to this problem would be to use a pyramid of different patch sizes at any given task for each object class, and combine their contributions in inference.

Contour Representation Currently, we use rectangular patches to represent objects. Although we have shown that such patches can be successfully used for a variety of vision tasks including semantic segmentation. We suspect that rectangular patches would not be suit-

able for representing contours of objects, particularly for object segmentation tasks. For example, in our semantic segmentation model, a patch is allowed to contain more than one class label, however we do not have an explicit way of representing the boundary between these classes. One possible solution to this problem would be to use irregularly shaped patches such as the ones used in [93], which would be better suited to representing object contours.

7.4 Future work

3D Patches So far, we have used 2D patches to represent object in a given image, and have shown that they can be successfully used for a variety of object related tasks. This motivates us to extend our current models to use 3D patches for object representation. Many computer vision tasks such as action recognition or tracking use video data as input. Similarly, in medical image analysis one deals with data volumes such as a full 3D MRI scan of the brain. We believe that 3D patches or ‘cubes’ could be potentially used for the above applications to capture spatio-temporal information and to represent 3D objects.

Articulated Objects In this thesis, we have developed patch-based models for structured classes of objects that show a certain regularity such as, faces or pedestrians etc. In the future, we would like to investigate the capabilities of patch-based models to represent articulated objects such as horses, or cows etc (see figure 1.2d). We would expect patches to have a reasonable performance on tasks such as detection and classification. But we suspect that rectangular patches may find some tasks like segmentation more challenging.

Other Object-related Applications One of the properties of patch-based models is that it allows dense representation of an object. This property encourages future applications of this model for object *generation* (see figure 1.1g). Furthermore, we would like to use patch-based representation for bounding-box object localization.

General N-D Signal Processing Several properties of patch-based representation gives it a great potential to be used in general N-D signal processing problems. First of all, it is a general representation model that does not assume any specific format. Also the proposed data encoding method, where each patch is represented as an index to a predefined library, results in an extremely more compact representation of a given signal. This can be especially valuable for very high dimensional data and could help (i) greatly reduce the parameter space, (ii) improve the signal to noise ratio and (iii) help avoid the curse of dimensionality.

Appendices

Appendix A

Calculating The Dirichlet Posterior Integral

In a Bayesian framework we integrate over the model parameters to find the likelihood of the test data given a class as follows:

$$Pr(\mathbf{y}_p | \mathcal{C} = c, \mathbf{x}_{\bullet cp}) = \int Pr(\mathbf{y}_p | \theta_{cp\bullet}) Pr(\theta_{cp\bullet} | \mathbf{x}_{\bullet cp}) d\theta_{cp\bullet} \quad (\text{A.1})$$

The first term in Equation A.1 is the likelihood term i.e. The probability of the test data conditioned on the model parameters. This likelihood has a multinomial which is defined as θ_{cpl^*} :

$$Pr(\mathbf{y}_p | \theta_{cp\bullet}) = Pr(l^* | \theta_{cp\bullet}) = \theta_{cpl^*} \quad (\text{A.2})$$

The second term in Equation A.1 is the posterior distribution over the parameters $\theta_{cp\bullet}$ was achieved as a result of multiplying a multinomial likelihood and the conjugate prior which is a Dirichlet distribution hence it has the form of a Dirichlet distribution defined in Equation A.3 as:

$$Pr(\theta_{cp\bullet} | \mathbf{x}_{\bullet cp}) \propto \frac{I!}{f_{cp1}! \dots f_{cpN}!} \frac{\Gamma(\sum_l \alpha_l)}{\prod_l \Gamma(\alpha_l)} \prod_l \theta_{cpl}^{f_{cpl} + \alpha_l - 1} \quad (\text{A.3})$$

Now replacing Equation A.3 and Equation A.2 in Equation A.1 we get:

$$\begin{aligned} Pr(\mathbf{y}_p | \mathcal{C} = c, \mathbf{x}_{\bullet cp}) &= \theta_{cpl^*} \int \frac{\Gamma(\sum_l (\alpha_l + f_{cpl}))}{\prod_l \Gamma(\alpha_l + f_{cpl})} \prod_l \theta_{cpl}^{f_{cpl} + \alpha_l - 1} d\theta_{cp\bullet} \\ &= \int \frac{\Gamma(\sum_l (\alpha_l + f_{cpl}))}{\prod_l \Gamma(\alpha_l + f_{cpl})} \prod_l \theta_{cpl}^{\delta_{l^*l} + f_{cpl} + \alpha_l - 1} d\theta_{cp\bullet} \end{aligned} \quad (\text{A.4})$$

Where $\delta_{l^*l} = 1$ iff $l^* = l$ and zero otherwise.

Turning the term inside the integral to a complete Dirichlet distribution which sums to one we can rewrite Equation A.4 as the following:

$$\begin{aligned}
 Pr(\mathbf{y}_p | \mathcal{C}=\mathcal{c}, \mathbf{x}_{\bullet cp}) &= \frac{\Gamma(\sum_l (f_{cpl} + \alpha_l))}{\prod_l \Gamma(f_{cpl} + \alpha_l)} \frac{\prod_l \Gamma(\delta_{l^*l} + f_{cpl} + \alpha_l)}{\Gamma(\sum_l (\delta_{l^*l} + f_{cpl} + \alpha_l))} \\
 &= \frac{f_{cpl^*} + \alpha_{l^*}}{\sum_l (f_{cpl} + \alpha_l)} \tag{A.5}
 \end{aligned}$$

Bibliography

- [1] <http://vision.cs.brown.edu/humaneva/index.html>.
- [2] <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>.
- [3] <http://www.ee.surrey.ac.uk/Research/VSSP/xm2vtsdb>.
- [4] J. Aghajanian and S. Prince. Mosaicfaces: a discrete representation for face recognition. In *WACV*, pages 1–8, 2008.
- [5] J. Aghajanian and S. Prince. Face pose estimation in uncontrolled environments. *BMVC*, 2009.
- [6] J. Aghajanian, J. Warrell, S. Prince, P. Li, and J. Rohn. Patch-based Within-Object Classification. In *ICCV*, 2009.
- [7] M. Andriluka, S. Roth, and B. Schiele. Pictorial structures revisited: People detection and articulated pose estimation. In *Proc. CVPR*, volume 1, page 4, 2009.
- [8] S. Avidan and A. Shamir. Seam carving for content-aware image resizing. *ACM Transactions on Graphics*, 26(3):10, 2007.
- [9] K. Babalola, T. Cootes, C. Twining, V. Petrovic, and C. Taylor. 3D brain segmentation using active appearance models and local regressors. *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2008*, pages 401–408, 2008.
- [10] V. Balasubramanian, J. Ye, and S. Panchanathan. Biased Manifold Embedding: A Framework for Person-Independent Head Pose Estimation. In *CVPR*, pages 1–7, 2007.
- [11] S. Baluja and H. Rowley. Boosting Sex Identification Performance. *International Journal of Computer Vision*, 71(1):111–119, 2007.
- [12] M. Bartlett, J. Movellan, and T. Sejnowski. Face recognition by independent component analysis. *Neural Networks, IEEE Transactions on*, 13(6):1450–1464, 2002.
- [13] B. Baum and J. Rohn. MRC Laboratory For Molecular Cell Biology.

- [14] P. Belhumeur, J. Hespanha, D. Kriegman, et al. Eigenfaces vs. Fisherfaces: recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.
- [15] C. BenAbdelkader and P. Griffin. A local region-based approach to gender classification from face images. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*.
- [16] A. Berg, F. Grabler, and J. Malik. Parsing images of architectural scenes. In *IEEE 11th International Conference on Computer Vision, 2007. ICCV 2007*, pages 1–8, 2007.
- [17] C. Bishop. *Neural networks for pattern recognition*. Oxford University Press, USA, 1995.
- [18] C. Bishop and S. O. service). *Pattern recognition and machine learning*. Springer, 2006.
- [19] A. Blake and M. Isard. 3D position, attitude and shape input using video tracking of hands and lips. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 185–192. ACM, 1994.
- [20] V. Blanz, S. Romdhani, and T. Vetter. Face identification across different poses and illuminations with a3D morphable model. *Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on*, pages 192–197, 2002.
- [21] V. Blanz and T. Vetter. A morphable model for the synthesis of 3D faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 187–194, 1999.
- [22] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239, 2002.
- [23] M. Breitenstein, D. Kuettel, T. Weise, L. Van Gool, and H. Pfister. Real-time face pose estimation from single range images. In *CVPR*, pages 1–8, 2008.
- [24] R. Brunelli and T. Poggio. HyperBF Networks for Gender Classification. *Image Understanding 1992*, 1992.
- [25] M. Burl, M. Weber, and P. Perona. A probabilistic approach to object recognition using local photometry and global geometry. *Computer VisionECCV98*, page 628, 1998.
- [26] A. Carpenter, T. Jones, M. Lamprecht, C. Clarke, I. Kang, O. Friman, D. Guertin, J. Chang, R. Lindquist, J. Moffat, et al. CellProfiler: image analysis software for identifying and quantifying cell phenotypes. *Genome Biol*, 7(10):R100, 2006.

- [27] H. C.G. Determination of Ego-Motion From Matched Points. *Proc. Alvey Vision Conf., Cambridge, UK*, 1987.
- [28] V. Cheung, B. Frey, and N. Jojic. Video epitomes. *IJCV*, 76(2):141–152, 2008.
- [29] V. Cheung, N. Jojic, and D. Samaras. Capturing long-range correlations with patch models. In *CVPR*, volume 1-8, 2007.
- [30] T. Cho, M. Butman, S. Avidan, and W. Freeman. The patch transform and its applications to image editing. In *CVPR*, pages 1–8, 2008.
- [31] K. Choi, M. Carcassoni, and E. Hancock. Estimating 3D facial pose using the EM algorithm. *Face Recognition: From Theory to Applications*, pages 412–423, 1998.
- [32] T. Cootes, G. Edwards, C. Taylor, et al. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001.
- [33] T. Cootes and C. Taylor. Mixture model for representing shape variation. *Image and Vision Computing*, 17(8):567–573, 1999.
- [34] T. Cootes, C. Taylor, D. Cooper, J. Graham, et al. Active shape models-their training and application. *Computer vision and image understanding*, 61(1):38–59, 1995.
- [35] T. Cootes, K. Walker, and C. Taylor. View-based active appearance models. In *Fourth IEEE International Conference on Automatic Face and Gesture Recognition, 2000. Proceedings*, pages 227–232, 2000.
- [36] N. Costen, M. Brown, and S. Akamatsu. Sparse models for gender classification. *Proceedings of the 6th International Conference on Automatic Face and Gesture Recognition*, 2004.
- [37] G. Cottrell and J. Metcalfe. EMPATH: face, emotion, and gender recognition using holons. *Proceedings of the 1990 conference on Advances in neural information processing systems 3 table of contents*, pages 564–571, 1990.
- [38] G. Csurka, C. Bray, C. Dance, and L. Fan. Visual categorization with bags of keypoints. *Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22, 2004.
- [39] N. Dalai, B. Triggs, I. Rhone-Alps, and F. Montbonnot. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, 2005.
- [40] A. Dempster, N. Laird, and D. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

- [41] J. Domke, A. Karapurkar, and Y. Aloimonos. Who killed the directed model. In *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR 2008)*(24–26 June 2008, Anchorage, AL, USA), 2008.
- [42] J. Domke, A. Karapurkar, and Y. Aloimonos. Who killed the directed model? In *CVPR*, pages 1–8, 2008.
- [43] G. Edwards, T. Cootes, and C. Taylor. Face recognition using active appearance models. *ECCV*, page 581, 1998.
- [44] G. Edwards, C. Taylor, and T. Cootes. Learning to Identify and Track Faces in Image Sequences. *Int. Conf. on Face and Gesture Recognition*, 1998.
- [45] A. Efros, A. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *ICCV*, pages 726–733, 2003.
- [46] A. Efros and W. Freeman. Image quilting for texture synthesis and transfer. In *SIGGRAPH*, pages 341–346, 2001.
- [47] J. Efros. Scene Completion Using Millions of Photographs. *ACM Transactions on Graphics*, 26(3), 2007.
- [48] E. Elagin, J. Steffens, and H. Neven. Automatic pose estimation system for human faces based on bunchgraph matching technology. In *AFGR*, pages 136–141, 1998.
- [49] A. Evans. The NIH MRI study of normal brain development i. *NeuroImage*, 30:184–202, 2006.
- [50] M. Everingham and A. Zisserman. Regression and classification approaches to eye localization in face images. In *Proc. Int. Conf. Autom. Face and Gesture Recog*, 2006.
- [51] I. Fasel, B. Fortenberry, and J. Movellan. A generative framework for real time object detection and classification. *Computer Vision and Image Understanding*, 98(1):182–210, 2005.
- [52] L. Fei-Fei, R. Fergus, and P. Perona. A Bayesian approach to unsupervised one-shot learning of object categories. *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1134–1141, 2003.
- [53] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.
- [54] P. Felzenszwalb and D. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, 2005.

- [55] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. *Computer Vision and Pattern Recognition, Anchorage, USA, June, 2008*.
- [56] X. Feng, C. Williams, and S. Felderhof. Combining belief networks and neural networks for scene segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 467–483, 2002.
- [57] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on, 2, 2003*.
- [58] V. Ferrari, M. Marin-Jimenez, and A. Zisserman. Progressive search space reduction for human pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pages 1–8, 2008.
- [59] M. Fischler and R. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, 100(22):67–92, 1973.
- [60] W. Forstner. A Framework for Low Level Feature Extraction. *European Conference on Computer Vision, 1994*.
- [61] W. Freeman and E. Adelson. The design and use of steerable filters. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 13(9):891–906, 2002.
- [62] W. Freeman, E. Pasztor, and O. Carmichael. Learning Low-Level Vision. *International Journal of Computer Vision*, 40(1):25–47, 2000.
- [63] W. Freeman and M. Roth. Orientation histograms for hand gesture recognition. In *International Workshop on Automatic Face and Gesture Recognition*, volume 12. Citeseer, 1995.
- [64] B. Frey and N. Jovic. Transformed component analysis: Joint estimation of spatial transformations and image components. In *iccv*, page 1190. Published by the IEEE Computer Society, 1999.
- [65] Y. Fu and T. Huang. Graph embedded analysis for head pose estimation. In *AFGR*, page 6, 2006.
- [66] Y. Fu and S. Prince. Investigating the spatial support of signal and noise in face recognition. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 131–138. IEEE, 2010.

- [67] B. Golomb, D. Lawrence, and T. Sejnowski. Sexnet: A neural network identifies sex from human faces. *Advances in Neural Information Processing Systems*, 3:572–577, 1991.
- [68] A. Graf and F. Wichmann. Gender Classification of Human Faces. *LECTURE NOTES IN COMPUTER SCIENCE*, pages 491–500, 2002.
- [69] R. Gross, I. Matthews, and S. Baker. Eigen light-fields and face recognition across pose. *Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on*, pages 1–7, 2002.
- [70] R. Gross, I. Matthews, and S. Baker. Active appearance models with occlusion. *Image and Vision Computing*, 24(6):593–604, 2006.
- [71] N. Grujic, S. Ilic, V. Lepetit, and P. Fua. 3d facial pose estimation by image retrieval. In *8th IEEE International Conference on Automatic Face and Gesture Recognition*, 2008.
- [72] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1):389–422, 2002.
- [73] F. Han and S. Zhu. Bottom-up/top-down image parsing with attribute grammar. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(1):59–73, 2008.
- [74] J. Hays and A. Efros. IM2GPS: estimating geographic information from a single image. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.
- [75] J. Hays and A. Efros. Scene completion using millions of photographs. *CACM*, 51(10):87–94, 2008.
- [76] X. He, S. Yan, Y. Hu, et al. Face recognition using Laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):328–340, 2005.
- [77] X. He, R. Zemel, and M. Carreira-Perpinan. Multiscale conditional random fields for image labeling. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, 2004.
- [78] J. Heinzmann and A. Zelinsky. 3-D facial pose and gaze point estimation using a robust real-time tracking paradigm. In *AFGR*, pages 142–147, 1998.
- [79] K. Heller and Z. Ghahramani. A Simple Bayesian Framework for Content-Based Image Retrieval. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2:2110–2117, 2006.
- [80] D. Hoiem, A. Efros, and M. Hebert. Putting objects into perspective. In *CVPR*, 2006.

- [81] A. Holub, M. Welling, and P. Perona. Combining generative models and fisher kernels for object recognition. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005*, volume 1, 2005.
- [82] A. Holub, M. Welling, and P. Perona. Hybrid generative-discriminative visual categorization. *International Journal of Computer Vision*, 77(1):239–258, 2008.
- [83] Y. Hu, L. Chen, Y. Zhou, and H. Zhang. Estimating face pose by facial asymmetry and geometry. In *AFGR*, pages 651–656, 2004.
- [84] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.
- [85] A. Hyvarinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. J. Wiley New York, 2001.
- [86] K. Ikuta, H. Kage, K. Sumi, K. Tanaka, and K. Kyuma. SOM-Based Continuous Category Learning for Age Classification by Facial Images. *Lecture Notes in Computer Science*, 4985:569–576, 2008.
- [87] H. Ishikawa. Exact optimization for Markov random fields with convex priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1333–1336, 2003.
- [88] N. Jojic, B. Frey, and A. Kannan. Epitomic analysis of appearance and shape. In *ICCV*, volume 2, 2003.
- [89] E. Jones and S. Soatto. Layered active appearance models. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005*, volume 2, 2005.
- [90] M. Jones and T. Poggio. Multidimensional morphable models: A framework for representing and matching object classes. *International Journal of Computer Vision*, 29(2):107–131, 1998.
- [91] T. Jones, A. Carpenter, D. Sabatini, and P. Golland. Methods for high-content, high-throughput image-based cell screening. In *Proceedings of the Workshop on Microscopic Image Analysis with Applications in Biology held in association with MICCAI06 (Medical Image Computing and Computer-Assisted Intervention) held in Copenhagen, Denmark, October 5, 2006*, pages 65–72.
- [92] E. Kalogerakis, O. Vesselova, J. Hays, A. A. Efros, and A. Hertzmann. Image sequence geolocation with human travel priors. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV '09)*, 2009.

- [93] A. Kannan, J. Winn, and C. Rother. Clustering appearance and shape by learning jigsaws. *NIPS*, 19:657, 2007.
- [94] A. Konoplev. Luxand FaceSDK.
- [95] N. Kruger, M. Potzsch, T. Maurer, and M. Rinne. Estimation of face position and pose with labeled graphs. In *BMVC*, pages 735–743, 1996.
- [96] M. Kumar, P. Torr, and A. Zisserman. Extending pictorial structures for object recognition. In *Proc. BMVC*, 2004.
- [97] M. Kumar, P. Torr, and A. Zisserman. OBJ CUT. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)-Volume 1-Volume 01*, page 25, 2005.
- [98] N. Kumar, P. Belhumeur, and S. Nayar. FaceTracer: A Search Engine for Large Collections of Images with Faces. *ECCV*, 2008.
- [99] S. Kumar and M. Hebert. Discriminative random fields. *International Journal of Computer Vision*, 68(2):179–201, 2006.
- [100] V. Kwatra, A. Schodl, I. Essa, G. Turk, and A. Bobick. Graphcut textures: Image and video synthesis using graph cuts. *ACM Transactions on Graphics*, 26(3):227–286, 2003.
- [101] J. Laferte, F. Heitz, P. Perez, and E. Fabre. Hierarchical statistical models for the fusion of multiresolution image data. In *PROCEEDINGS-SPIE THE INTERNATIONAL SOCIETY FOR OPTICAL ENGINEERING*, pages 42–42, 1995.
- [102] A. Lanitis, C. Draganova, and C. Christodoulou. Comparing different classifiers for automatic age estimation. *Systems, Man and Cybernetics, Part B, IEEE Transactions on*, 34(1):621–628, 2004.
- [103] J. Lasserre, A. Kannan, J. Winn, and U. Cambridge. Hybrid learning of large jigsaws. In *Proc. CVPR*, 2007.
- [104] D. Lee and S. H.S. Learning the parts of objects by nonnegative matrix factorization. *Nature*, pages 788–791, 1999.
- [105] A. Leonardis and H. Bischof. Robust recognition using eigenimages. *Computer Vision and Image Understanding*, 78(1):99–118, 2000.
- [106] M. Leung and Y. Yang. First Sight: A Human Body Outline Labeling System. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, page 359377, 1995.

- [107] P. Li and S. Prince. Joint and implicit registration for face recognition. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1510–1517. IEEE, 2009.
- [108] S. Li. Markov random field models in computer vision. *Lecture Notes in Computer Science*, 801:361–370, 1994.
- [109] S. Li, Q. Fu, L. Gu, B. Scholkopf, Y. Cheng, and H. Zhang. Kernel machine based learning for multi-view face detection and pose estimation. In *ICCV*, volume 2, 2001.
- [110] S. Li, X. Peng, X. Hou, H. Zhang, and Q. Cheng. Multi-view face pose estimation based on supervised isa learning. In *AFGR*, pages 100–105, 2002.
- [111] S. Li and Z. Zhang. FloatBoost Learning and Statistical Face Detection. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, pages 1112–1123, 2004.
- [112] Y. Li, L. Shapiro, and J. Bilmes. A generative/discriminative learning algorithm for image classification. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005*, volume 2, 2005.
- [113] G. Little, K. S., B. J., and P. S. A methodology for evaluating robustness of face recognition algorithms with respect to changes in pose and illumination angle. In *ICASSP*, 2005.
- [114] E. Loupiaz, N. Sebe, S. Bres, and J. Jolion. Wavelet-based salient points for image retrieval. *Image Processing, 2000. Proceedings. 2000 International Conference on*, 2, 2000.
- [115] D. Lowe. Object recognition from local scale-invariant features. *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, 2, 1999.
- [116] X. Lu, H. Chen, and A. Jain. Multimodal Facial Gender and Ethnicity Identification. *LECTURE NOTES IN COMPUTER SCIENCE*, 3832:554, 2006.
- [117] X. Lu and A. Jain. Automatic feature extraction for multiview 3D face recognition. In *7th International Conference on Automatic Face and Gesture Recognition, 2006. FGR 2006*, pages 585–590, 2006.
- [118] S. Lucey and T. Chen. Learning Patch Dependencies for Improved Pose Mismatched Face Verification. *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Volume 1*, pages 909–915, 2006.

- [119] Y. Ma, Y. Konishi, K. Kinoshita, S. Lao, and M. Kawade. Sparse bayesian regression for head pose estimation. In *ICPR*, 2006.
- [120] E. Mäkinen and R. Raisamo. An experimental comparison of gender classification methods. *Pattern Recognition Letters*, 29(10):1544–1556, 2008.
- [121] T. Malisiewicz and A. Efros. Beyond categories: the visual memex model for reasoning about object relationships. *Neural Inf. Proc. Systems*, 2009.
- [122] Y. Matsumoto and A. Zelinsky. An algorithm for real-time stereo vision implementation of headpose and gaze direction measurement. In *AFGR*, pages 499–504, 2000.
- [123] I. Matthews and S. Baker. Active appearance models revisited. *International Journal of Computer Vision*, 60(2):135–164, 2004.
- [124] B. Moghaddam and M. Yang. Learning Gender with Support Faces. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, pages 707–711, 2002.
- [125] U. Mohammed. Generative Models For Face Recognition. 2006.
- [126] U. Mohammed, S. Prince, and J. Kautz. Visio-lization Generating Novel Facial Images. In *SIGGRAPH*, 2009.
- [127] A. Mohan, C. Papageorgiou, and T. Poggio. Example-based object detection in images by components. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(4):349–361, 2001.
- [128] A. Moore, S. Prince, and J. Warrell. Lattice Cut-Constructing superpixels using layer constraints. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2117–2124. IEEE, 2010.
- [129] F. Moosmann, B. Triggs, and F. Jurie. Fast discriminative visual codebooks using randomized clustering forests. *Advances in neural information processing systems*, 19:985, 2007.
- [130] L. Morency, P. Sundberg, and T. Darrell. Pose estimation using 3d view-based eigenspaces. In *AMFG*, pages 45–52, 2003.
- [131] H. Murase and S. Nayar. Visual learning and recognition of 3-d objects from appearance. *International Journal of Computer Vision*, 14(1):5–24, 1995.
- [132] K. Okada, S. Akamatsu, and C. Von der Malsburg. Analysis and synthesis of pose variations of human faces by a linear PCMAP model and its application for pose-invariant facerecognition system. In *AFGR*, pages 142–149, 2000.

- [133] M. Oren, C. Papageorgiou, P. Sinha, E. Osuna, and T. Poggio. Pedestrian detection using wavelet templates. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 193–199, 1997.
- [134] E. Osuna, R. Freund, and F. Girosit. Training support vector machines: an application to face detection. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 130–136, 1997.
- [135] F. B. J. Pal, C. and J. N. Learning montages of transformed latent images as representations of objects that change in appearance. *ECCV*, pages 715–731, 2002.
- [136] V. Perlibakas. Distance measures for PCA-based face recognition. *Pattern Recognition Letters*, 25(6):711–724, 2004.
- [137] P. Phillips, H. Moon, S. Rizvi, and P. Rauss. The FERET evaluation methodology for face-recognition algorithms. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(10):1090–1104, 2000.
- [138] S. Prince and J. Aghajanian. Gender classification in uncontrolled settings using additive logistic models. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 2557–2560. IEEE, 2010.
- [139] S. Prince and J. Elder. Probabilistic linear discriminant analysis for inferences about identity. In *Proceedings of the IEEE 11th International Conference on Computer Vision (ICCV07)*, page 18.
- [140] S. Prince and J. Elder. Tied factor analysis for face recognition across large pose changes.
- [141] S. Prince, J. Warrell, J. Elder, and F. Felisberti. Tied factor analysis for face recognition across large pose differences. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(6):970–984, 2008.
- [142] Y. Pritch, E. Kav-Venaki, and S. Peleg. Shift-Map Image Editing.
- [143] L. Rabiner et al. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.
- [144] D. Ramanan. Learning to parse images of articulated bodies. *Advances in Neural Information Processing Systems*, 19:1129, 2007.
- [145] C. Rasmussen and C. Williams. *Gaussian processes for machine learning*. Springer, 2006.
- [146] B. Raytchev, I. Yoda, and K. Sakaue. Head pose estimation by nonlinear manifold learning. In *ICPR*, volume 4, 2004.

- [147] S. Romdhani, S. Gong, and A. Psarrou. A multi-view nonlinear active shape model using kernel pca. *British Machine Vision Conference*, 99:483–492, 1999.
- [148] S. Roth and M. Black. Fields of experts: A framework for learning image priors. In *CVPR*, volume 2, 2005.
- [149] H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(1):23–38, 1998.
- [150] Y. Saatci and C. Town. Cascaded Classification of Gender and Facial Expression using Active Appearance Models. *Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition*, 80:393–400, 2006.
- [151] D. Schlesinger and B. Flach. Transforming an arbitrary minsum problem into a binary one. *Dresden University of Technology*, 2006.
- [152] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: A local SVM approach. In *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 3-Volume 03*, page 36. IEEE Computer Society, 2004.
- [153] S. Sclaroff and J. Isidoro. Active blobs. *ICCV*, 1998.
- [154] G. Shakhnarovich, P. Viola, and B. Moghaddam. A unified learning framework for real time face detection and classification. In *Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on*, pages 14–21, 2002.
- [155] N. Shental, A. Bar-Hillel, T. Hertz, and D. Weinshall. Computing Gaussian mixture models with EM using equivalence constraints. *Advances in Neural Information Processing Systems*, 16:465–472, 2004.
- [156] J. Shotton, A. Blake, and R. Cipolla. Contour-based learning for object detection. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005*, pages 503–510, 2005.
- [157] J. Shotton, J. Winn, C. Rother, and A. Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. *Lecture Notes in Computer Science*, 3951:1, 2006.
- [158] J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman. Discovering object categories in image collections. *Proc. ICCV*, 1:65, 2005.
- [159] E. Sudderth, A. Torralba, W. Freeman, and A. Willsky. Learning hierarchical models of scenes, objects, and parts. *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, 2, 2005.

- [160] X. Tan and B. Triggs. Enhanced local texture feature sets for face recognition under difficult lighting conditions. In *Proceedings of the 3rd international conference on Analysis and modeling of faces and gestures*, pages 168–182. Springer-Verlag, 2007.
- [161] M. Turk and A. Pentland. Face recognition using eigenfaces. *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on*, pages 586–591, 1991.
- [162] S. Ullman, E. Sali, and M. Vidal-Naquet. A fragment-based approach to object representation and classification. *Visual Form 2001*, pages 85–100, 2001.
- [163] V. Vapnik, S. Golowich, and A. Smola. Support vector method for function approximation, regression estimation, and signal processing. *Advances in Neural Information Processing Systems 9*, page 281, 1997.
- [164] T. Vetter and T. Poggio. Linear object classes and image synthesis from a single example image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):733–742, 1997.
- [165] P. Viola and M. Jones. Rapid Object Detection Using a Boosted Cascade of Simple Features. *IEEE COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION*, 1, 2001.
- [166] J. Wang, E. Sung, and R. Venkateswarlu. Determining pose of a human face from a single monocular image. In *BMVC*, 2003.
- [167] X. Wang and X. Tang. Dual-space linear discriminant analysis for face recognition. *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, 2.
- [168] J. Warrell and S. Prince. Labelfaces: parsing facial features by multiclass labeling with an epitome prior. In *Proceedings of the 16th IEEE international conference on Image processing*, pages 2453–2456. IEEE Press, 2009.
- [169] J. Warrell, S. Prince, and A. More. Epitomized Priors for Multi-labeling Problems. *CVPR*, 2009.
- [170] M. Weber, M. Welling, and P. Perona. Unsupervised learning of models for recognition. *Proc. ECCV*, 1:18–32, 2000.
- [171] J. Winn, A. Criminisi, and T. Minka. Object Categorization by Learned Universal Visual Dictionary. *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, 2, 2005.

- [172] J. Winn and J. Shotton. The layout consistent random field for recognizing and segmenting partially occluded objects. 2006.
- [173] L. Wiskott, J. Fellous, N. Kruger, and C. Von der Malsburg. Face recognition by elastic bunch graph matching. In *Computer Analysis of Images and Patterns: 7th International Conference, CAIP'97: Kiel, Germany, September 10-12, 1997: Proceedings*, page 456. Springer, 1997.
- [174] B. Wu, H. Ai, C. Huang, and S. Lao. Fast rotation invariant multi-view face detection based on real Adaboost. In *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*, pages 79–84, 2004.
- [175] W. Wu, Z. QiSen, and W. Mingjun. A method of vehicle classification using models and neural networks. In *Vehicular Technology Conference, 2001. VTC 2001 Spring. IEEE VTS 53rd*, volume 4, 2001.
- [176] Z. Yang and H. Ai. Demographic Classification with Local Binary Patterns. *LECTURE NOTES IN COMPUTER SCIENCE*, 4642:464, 2007.
- [177] Y. Zhou, L. Gu, and H. Zhang. Bayesian tangent shape model: Estimating shape and pose parameters via bayesian inference. In *CVPR*, volume 1, 2003.