

LEARNING PARTS-BASED REPRESENTATIONS OF DATA

by

David A. Ross

A thesis submitted in conformity with the requirements  
for the degree of Master of Science  
Graduate Department of Computer Science  
University of Toronto

Copyright © 2003 by David A. Ross

# Abstract

Learning Parts-Based Representations of Data

David A. Ross

Master of Science

Graduate Department of Computer Science

University of Toronto

2003

Many collections of data exhibit a common underlying structure: they consist of a number of parts or factors, each with a range of possible states. For example, in a collection of facial images, every image contains eyes, a nose, and a mouth, each of which has a number of appearances.

We propose a new method, Multiple Cause Vector Quantization, for the unsupervised learning of parts-based representations of data. Our technique automates the segmentation of the data dimensions into parts, while simultaneously learning a discrete model of the range of appearances of each part.

We pose MCVQ as a probabilistic graphical model, and derive an efficient variational-EM algorithm for learning and inference. We present applications of this model to problems in image decomposition, collaborative filtering, and document modeling.

# Acknowledgements

First I would like to thank Rich Zemel, my supervisor, for his hospitality and kindness. I appreciate effort you have given to ensure my research career got off to a good start.

Secondly I would like to thank Sam Roweis, my second reader, as well as Brendan Frey and Geoff Hinton. Your comments, suggestions, and advice have been invaluable for completing this work.

Thirdly I would like to thank Ruslan Salakhutdinov, Yee Whye Teh, and everyone else in Toronto's machine learning group for the friendly work environment you fostered.

Finally I would like to thank Karolina, my family, and my friends. I owe you all a considerable amount of my time, and intend to pay you back fully.

Research described herein has been supported by grants from the Ontario Graduate Scholarship Program (OGS), the Natural Sciences and Engineering Research Council of Canada (NSERC), and the Institute for Robotics and Intelligent Systems (IRIS).

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Parts . . . . .	1
1.2	Thesis Organization . . . . .	3
<b>2</b>	<b>Multiple Cause Vector Quantization</b>	<b>5</b>
2.1	Overview . . . . .	5
2.2	Generative Model . . . . .	6
2.3	Learning and Inference . . . . .	7
2.4	Related models . . . . .	9
2.5	Conclusion . . . . .	12
<b>3</b>	<b>Experiments</b>	<b>13</b>
3.1	Parts-based Image Decomposition: Shapes and Faces . . . . .	13
3.1.1	Overview . . . . .	13
3.1.2	Shape Images . . . . .	14
3.1.3	Face Images . . . . .	17
3.2	Collaborative Filtering . . . . .	18
3.3	Document Modeling . . . . .	20
3.4	Classification: Face Expressions . . . . .	21
3.4.1	AR Face Database . . . . .	22
3.4.2	MCVQ for feature selection . . . . .	23

<b>4</b>	<b>Discovering dependencies among part selections</b>	<b>25</b>
4.1	Introduction . . . . .	25
4.2	Model . . . . .	26
4.2.1	Unsupervised Case . . . . .	27
4.2.2	Supervised Case . . . . .	29
4.3	Experiments . . . . .	30
4.3.1	Supervised Case . . . . .	30
4.3.2	Unsupervised Case . . . . .	34
4.4	Discussion . . . . .	35
<b>5</b>	<b>Conclusions</b>	<b>36</b>
5.1	Discussion . . . . .	36
5.2	Future Directions . . . . .	37
5.2.1	Continuously Parameterized Part Models . . . . .	37
5.2.2	Alternative Variational Approximation . . . . .	38
	<b>Bibliography</b>	<b>41</b>

# List of Figures

2.1	Graphical model representation of MCVQ. . . . .	6
3.1	a) A sample of 24 training images from the Shapes dataset. b) A typical representation learned by MCVQ with 3 VQs and 5 states per VQ. c) Reconstruction of a test image: original (left) and reconstruction (right).	14
3.2	Other methods trained on shape images . . . . .	15
3.3	The reconstruction of two test images from the Faces dataset. . . . .	17
3.4	The MCVQ representation of two test users in the EachMovie dataset. . . . .	19
3.5	The average absolute deviation of predicted and true values of held-out ratings is compared for MCVQ and the aspect model. Note that the number of users per $x$ -bin decreases with increasing $x$ , as a user must rate at least $x + 1$ movies to be included. . . . .	20
3.6	The representation of two documents by an MCVQ model. . . . .	21
3.7	Examples of preprocessed images from the AR Face Database. . . . .	22
4.1	Graphical model with additional class variable, $\mathbf{y}^c$ , where $c$ is an index over the $C$ cases in our training data. Note that $\mathbf{y}^c$ can be either observed or unobserved. . . . .	27
4.2	Image regions explained by each VQ. . . . .	31
4.3	Samples generated from the supervised model . . . . .	33
4.4	Two training examples randomly selected from each of the three clusters.	35

5.1	Results on the shapes experiment, trained using the conventional mean-field variational posterior, equation (5.1). . . . .	39
5.2	Results on the shapes experiment using the entropic prior. . . . .	40

# List of Tables

3.1	Average root-mean-squared reconstruction error for various models trained on the shapes image data. . . . .	16
4.1	Class-conditional priors over mouth selections. . . . .	32
4.2	Number of images from each class assigned to each cluster. . . . .	34



# Chapter 1

## Introduction

### 1.1 Parts

Many collections of data exhibit a common underlying structure: they consist of a number of parts or factors, each with a range of possible states. When data are represented as vectors, parts manifest themselves as subsets of the data dimensions that take on values in a coordinated fashion. In the domain of digital images, these parts may correspond to the intuitive notion of the component parts of objects, such as the arms, legs, torso, and head of the human body. Prominent theories of computational vision, such as Biederman's *Recognition-by-Components* [3] advocate the suitability of a parts-based approach for recognition in both humans and machines. Recognizing an object by first recognizing its constituent parts, then validating their geometric configuration has several advantages:

1. Highly articulate objects, such as the human body, are able to appear in a wide range of configurations. It would be difficult to learn a holistic model capturing all of these variants.
2. Objects which are partially occluded can be identified as long as some of their parts are visible.

3. The appearances of certain parts may vary less under a change in pose than the appearance of the whole object. This can result in detectors which, for example, are more robust to rotations of the target object.
4. New examples from an object class may be recognized as simply a novel combination of familiar parts. For example a parts-base face detection system could generalize to detect faces with both beards and sunglasses, having been trained only on faces containing one, but not both, of these features.

The principal difficulty in creating such systems is determining which parts should be used, and identifying examples of these parts in the training data.

In the part-based detectors created by Mohan et al. [26] and Heisele et al. [13] parts were chosen by the experimenters based on intuition, and the component-detectors - support vector machines - were trained on image subwindows containing only the part in question. Obtaining these subwindows required that they be manually extracted from hundreds or thousands of training images.

In contrast, the parts-based detector created by Weber et al. [30] proposed a way to automate this process. First, potential parts - small subwindows of fixed size - were identified using the Förstner interest operator. Specifically, this selected regions containing intersecting lines, corners, or the centres of circular regions. Next, the number of potential parts was reduced using k-means clustering. Finally, while training the geometric model, the parts-set was further reduced, by selecting only those which lead to highest detection performance. The resulting detector relied on a very small number of parts (e.g. 3) corresponding to very small local features. Unlike the SVMs, which were trained on a range of appearances of the part, each of these part-detectors could identify only a single fixed appearance.

Parts-based representations of data can also be learned in an entirely unsupervised fashion. These parts can be used for subsequent supervised learning, but the models

constructed can also be valuable on their own. A parts-based model provides an efficient, distributed representation, and can aid in the discovery of causal structure within data. As an example, parts-based models are widely used by law enforcement agencies to produce composite sketches. These systems allow a witness to create a likeness of a suspect's face by selecting an appearance for each part, from a vocabulary of standard facial components.

The concept of parts can be applied to other types of data, such as bag-of-words text data, and preference data. In the former each data vector gives word count information - one vocabulary word per input dimension - for a single document. Parts, in this case, would correspond to collections of words with related frequencies. This includes words that often appear together, as well as words rarely appearing in the same document. In the latter, the vector contains ratings given by a human subject to a number of books, movies, et cetera. Parts would be formed from groups of related items, and appearances of a part would correspond to different attitudes towards the items.

This thesis describes a new method, Multiple Cause Vector Quantization, for the unsupervised learning of parts-based representations of data. Our technique automates the segmentation of the data dimensions into parts, while simultaneously learning a discrete model of the range of appearances of each part.

## 1.2 Thesis Organization

In Chapter 2 we introduce Multiple Cause Vector Quantization. We propose the model in 2.2, and in 2.3 derive an EM algorithm for learning and inference. In section 2.4 we relate MCVQ to similar parts-seeking techniques.

Experimental results using this method appear in Chapter 3, including applications

to image decomposition, collaborative filtering, document modeling, and classification of facial expressions.

In Chapter 4 we extend our model to handle data with class labels, learning these labels if they have not been provided.

Finally, in Chapter 5 we summarize our contributions, and suggest directions for further inquiry.

# Chapter 2

## Multiple Cause Vector Quantization

### 2.1 Overview

In this chapter we propose a stochastic generative model that can learn parts-based representations of high-dimensional data. Our key assumption is that the dimensions of the data can be separated into several disjoint subsets, or factors, which take on values independently of each other<sup>1</sup>. We assume each factor has a small number of discrete states, and model it using a *vector quantizer*. The selected states of each factor represent the *multiple causes* of the input. Given a set of training examples, our model learns the association of data dimensions with factors, as well as the states of each VQ. Inference and learning are carried out efficiently via variational algorithms.

This representational scheme is powerful due to its combinatorial nature: while a standard clustering/VQ method containing  $N$  states can represent at most  $N$  items, if we divide the  $N$  into VQs of  $J$  states each, we can represent  $J^{N/J}$  items. MCVQ is also especially appropriate for high-dimensional data in which many values may be unspecified for a given input case.

Material from this chapter and from the experiments in chapter 3 originally appeared

---

<sup>1</sup>Practically these factors are often not entirely independent, even when parts are involved. We investigate this situation further in Chapter 4

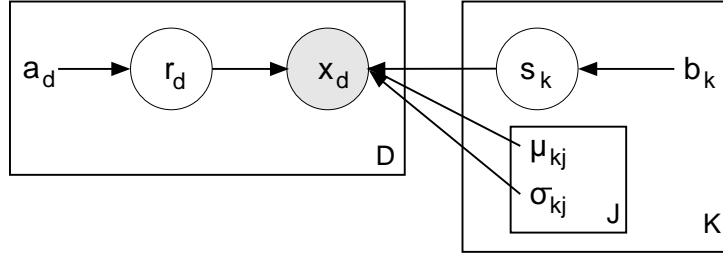


Figure 2.1: Graphical model representation of MCVQ. We let  $\mathbf{r}_{d=1}$  represent all the variables  $r_{d=1,k}$ , which together select a VQ for  $x_1$ . Similarly,  $\mathbf{s}_{k=1}$  represents all  $s_{k=1,j}$ , which together select a state of VQ 1. The plates depict repetitions across the appropriate dimensions for each of the three variables: the  $K$  VQs, the  $J$  states (codebook vectors) per VQ, and the  $D$  input dimensions. To extend this model to multiple data cases, we would include an additional plate over  $r$ ,  $x$ , and  $s$ .

in *Advances in Neural Information Processing Systems 15* [27].

## 2.2 Generative Model

In MCVQ we assume there are  $K$  factors, each of which is modeled by a vector quantizer with  $J$  states. To generate an observed data example of  $D$  dimensions,  $\mathbf{x} \in \mathfrak{R}^D$ , we stochastically select one state for each VQ, and one VQ for each dimension. Given these selections, a single state from a single VQ determines the value of each data dimension  $x_d$ .

The selections are represented as binary latent variables,  $S = \{s_{kj}\}$ ,  $R = \{r_{dk}\}$ , for  $d = 1 \dots D$ ,  $k = 1 \dots K$ , and  $j = 1 \dots J$ . The variable  $s_{kj} = 1$  if and only if state  $j$  has been selected from VQ  $k$ . Similarly  $r_{dk} = 1$  when VQ  $k$  has been selected for data dimension  $d$ . These variables can be described equivalently as multinomials,  $\mathbf{s}_k \in 1 \dots J$ ,  $\mathbf{r}_d \in 1 \dots K$ ; their values are drawn according to their respective priors,  $\mathbf{a}_k$  and  $\mathbf{b}_d$ . The graphical model representation of MCVQ is given in Fig. 2.1.

Assuming each VQ state specifies the mean as well as the standard deviation of a Gaussian distribution, and the noise in the data dimensions is conditionally independent, we have (where  $\theta = \{\mu_{dkj}, \sigma_{dkj}\}$ ):

$$P(\mathbf{x}|R, S, \theta) = \prod_d \prod_{k,j} \mathcal{N}(x_d; \mu_{dkj}, \sigma_{dkj})^{r_{dk} s_{kj}}$$

The resulting model can be thought of as a mixture model over  $J \times K$  possible states for each data dimension ( $x_d$ ). The single state  $kj$  is selected if  $s_{kj}r_{dk} = 1$ . Note that this selection has two components. The selection in the  $j$  component is made jointly for the different data dimensions, and in the  $k$  component it is made independently for each dimension.

## 2.3 Learning and Inference

The joint distribution over the observed vector  $\mathbf{x}$  and the latent variables is

$$P(\mathbf{x}, R, S|\theta) = P(R|\theta)P(S|\theta)P(\mathbf{x}|R, S, \theta) = \prod_{d,k} a_{dk}^{r_{dk}} \prod_{k,j} b_{kj}^{s_{kj}} \prod_{d,k,j} \mathcal{N}(x_d; \theta)^{r_{dk} s_{kj}} \quad (2.1)$$

Given an input  $\mathbf{x}$ , the posterior distribution over the latent variables,  $P(R, S|\mathbf{x}, \theta)$ , cannot tractably be computed, since all the latent variables become dependent.

We apply a variational EM algorithm to learn the parameters  $\theta$ , and infer latent variables given observations. We approximate the posterior distribution using a factored distribution, where  $g$  and  $m$  are variational parameters related to  $r$  and  $s$  respectively:

$$Q(R, S|\mathbf{x}, \theta) = \left( \prod_{d,k} g_{dk}^{r_{dk}} \right) \left( \prod_{k,j} m_{kj}^{s_{kj}} \right) \quad (2.2)$$

The variational free energy,  $\mathcal{F}(Q, \theta) = E_Q[-\log P(\mathbf{x}, R, S|\theta) + \log Q(R, S|\mathbf{x}, \theta)]$  is:

$$\begin{aligned} \mathcal{F} &= E_Q \left[ \sum_{d,k} r_{dk} \log(g_{dk}/a_{dk}) + \sum_{k,j} s_{kj} \log(m_{kj}/b_{kj}) + \sum_{d,k,j} r_{dk} s_{kj} \log \mathcal{N}(x_d; \theta) \right] \\ &= \sum_{k,j} m_{kj} \log m_{kj} + \sum_{d,k} g_{dk} \log g_{dk} + \sum_{d,k,j} g_{dk} m_{kj} \varepsilon_{dkj} \end{aligned}$$

where  $\varepsilon_{dkj} = \log \sigma_{dkj} + \frac{(x_d - \mu_{dkj})^2}{2\sigma_{dkj}^2}$ . The negative of the free energy  $-\mathcal{F}$  is a lower bound on the log likelihood of generating the observations. The variational EM algorithm improves this bound by iteratively improving  $-\mathcal{F}$  with respect to  $Q$  (E-step) and to  $\theta$  (M-step).

Let  $C$  be the set of training cases, and  $Q^c$  be the approximation to the posterior distribution over latent variables given the training case (observation)  $c \in C$ . We further constrain this variational approach, forcing the  $\{g_{dk}^c\}$  to be consistent across all observations  $\mathbf{x}^c$ . Hence these parameters relating to the gating variables that govern the selection of a factor for a given observation dimension, are not dependent on the observation. This approach encourages the model to learn representations that conform to this constraint. That is, if there are several posterior distributions consistent with an observed data vector, it favours distributions over  $\{\mathbf{r}_d\}$  that are consistent with those of other observed data vectors. Under this formulation, only the  $\{m_{kj}^c\}$  parameters are updated during the E step for each observation  $c$ :

$$m_{kj}^c = b_{kj} \exp\left(-\sum_d g_{dk} \varepsilon_{dkj}^c\right) / \sum_{\rho=1}^J b_{k\rho} \exp\left(-\sum_d g_{d\rho} \varepsilon_{d\rho k}^c\right)$$

The M step updates the parameters,  $\mu$  and  $\sigma$ , from each latent state  $kj$  to each input dimension  $d$ , the gating variables  $\{g_{dk}\}$ , and the priors  $\{a_{dk}\}$  and  $\{b_{kj}\}$ :

$$g_{dk} = a_{dk} \exp\left(-\frac{1}{C} \sum_{c,j} m_{kj}^c \varepsilon_{dkj}^c\right) / \sum_{\rho=1}^K a_{d\rho} \exp\left(-\frac{1}{C} \sum_{c,j} m_{j\rho}^c \varepsilon_{dj\rho}^c\right)$$

$$\mu_{dkj} = \frac{\sum_c m_{kj}^c x_d^c}{\sum_c m_{kj}^c} \quad \sigma_{dkj}^2 = \frac{\sum_c m_{kj}^c (x_d^c - \mu_{dkj})^2}{\sum_c m_{kj}^c}$$

$$a_{dk} = g_{dk} \quad b_{kj} = \frac{1}{C} \sum_c m_{kj}^c$$

A slightly different model formulation restricts the selections of VQs,  $\{r_{dk}\}$ , to be the same for each training case. Variational EM updates for this model are identical to those above, except that the  $\frac{1}{C}$  terms in the updates for  $g_{dk}$  disappear. In practice, we obtain



good results by replacing this  $\frac{1}{C}$  term with an inverse temperature parameter, that is annealed during learning. This can be thought of as gradually moving from a generative model in which the  $r_{dk}$ 's can vary across examples, to one in which they are the same for each example.

The inferred values of the variational parameters specify a posterior distribution over the VQ states, which in turn implies a mixture of Gaussians for each input dimension. Below we use the mean of this mixture,  $\hat{x}_d^c = \sum_{k,j} m_{kj}^c g_{dk} \mu_{dkj}$ , to measure the model's reconstruction error on case  $c$ . (In practice the posterior probability lies almost exclusively on a single choice of state per VQ, and a single choice of VQ per data dimension, thus the resulting mixture of Gaussians has a single predominant mode.)

A variational approximation is just one of a number of possible approaches to performing the intractable inference (E) step in MCVQ. One alternative would be to approximate the posterior with a set of samples drawn from the true posterior via Gibbs' sampling. Details of this approach for a closely related model can be found in [10].

## 2.4 Related models

MCVQ falls into the expanding class of unsupervised algorithms known as *factorial methods*, in which the aim of the learning algorithm is to discover multiple independent causes, or factors, that can well characterize the observed data. Its direct ancestor is Cooperative Vector Quantization [32, 14, 10], which has a very similar generative model to MCVQ, but lacks the stochastic selection of one VQ per data dimension. Instead, a data vector is generated cooperatively - each VQ selects one vector, and these vectors are summed to produce the data (again using a Gaussian noise model). The contrast between these approaches mirrors the development of the competitive mixture-of-experts algorithm [18] which grew out of the inability of a cooperative, linear combination of experts to decompose inputs into separable experts.

Unfortunately CVQ can learn unintuitive global features which include both additive and subtractive effects. A related model, non-negative matrix factorization (NMF) [20, 21, 24], proposes that each data vector is generated by taking a non-negative linear combination of non-negative basis vectors. Since each basis vector contains only non-negative values, it is unable to ‘subtract away’ the effects of other basis vectors it is combined with. This property encourages learning a basis of sparse vectors, each capturing a single instantiation of one of the independent latent factors, for example a local feature of an image. Like NMF, given non-negative data MCVQ will learn a non-negative basis, taken only in non-negative combinations. Unlike MCVQ, NMF provides no mechanism for learning compositional structure - how basis images or parts may be combined to form a valid whole. Rather, it considers any non-negative linear combination of basis vectors to be equally suitable, and hence NMF and MCVQ models differ in the range of novel examples they can generate<sup>2</sup>. Recent work such as [22] suggests that non-negativity alone may not be sufficient to ensure the learned basis corresponds to localized parts.

MCVQ also resembles a wide range of generative models developed to address image segmentation [31, 15, 19]. These are generally complex, hierarchical models designed to focus on a different aspect of this problem than that of MCVQ: to dynamically decide which pixels belong to which objects. The chief obstacle faced by these models is the unknown pose (primarily limited to position) of an object in an image, and they employ learned object models to find the single object that best explains each pixel. MCVQ adopts a more constrained solution with respect to part locations, assuming that these are consistent across images, and instead focuses on the assembling of input dimensions into parts, and the variety of instantiations of each part. The constraints built into MCVQ limit its generality, but also lead to rapid learning and inference, and enable it to scale up to high-dimensional data.

Connections can also be made between MCVQ and algorithms for *biclustering*, which

---

<sup>2</sup>For a concrete example, refer to the experiments on shape images appearing in Section 3.1.

aim to produce a simultaneous clustering of both the rows and the columns of the data matrix [25]. Biclustering has recently become popular in bioinformatics as a tool for analyzing DNA microarray data, which presents the expression levels for different genes under multiple experimental conditions as a matrix [8]. Assuming column-vector data, the selection of a VQ for each data dimension in MCVQ produces a clustering of the rows. MCVQ differs from other biclustering methods in that it produces not one but  $K$  clusterings of the columns, one for each of the  $K$  VQs. In Chapter 4 we present an extension that combines the clusterings, allowing MCVQ to produce a single biclustering of the data.

Finally, MCVQ also closely relates to sparse matrix decomposition techniques, such as the *aspect model* [16], a latent variable model which associates an unobserved class variable, the aspect  $z$ , with each observation. Observations consist of co-occurrence statistics, such as counts of how often a specific word occurs in a document. The latent Dirichlet allocation model [4] can be seen as a proper generative version of the aspect model: each document/input vector is not represented as a set of labels for a particular vector in the training set, and there is a natural way to examine the probability of some unseen vector. MCVQ shares the ability of these models to associate multiple aspects with a given document, yet it achieves this in a slightly different manner. The aspect and LDA models propose that each document – a list of exchangeable words – is generated by sampling an aspect, then sampling a word from the aspect, for each word in the document. On the other hand MCVQ models the aggregate word counts of a document. For each word in the vocabulary, its entire document frequency is generated according to the dictates of a stochastically-selected aspect (VQ). The stochastic selection leads to a posterior probability stipulating a soft mixture over aspects for each word. In the following chapter we present some initial experiments examining whether MCVQ can match the successful application of the aspect model to information retrieval and collaborative filtering problems, after evaluating it on image data.

As an addendum for the interested reader, two early factorial models of note are the Harmonium model of Freund and Haussler [9], and the Factorial Hidden Markov Model of Ghahramani and Jordan [12].

## 2.5 Conclusion

We have presented a novel method for learning factored representations of data which can be efficiently learned, and employed across a wide variety of problem domains. MCVQ combines the cooperative nature of some methods, such as CVQ, NMF, and LSA, that use multiple causes to generate input, with competitive aspects of clustering methods. In addition, it gains combinatorial power by splitting the input into subsets, and can readily handle sparse, high-dimensional data.

In the following chapter we explore applications of this method to several problem domains.

# Chapter 3

## Experiments

### 3.1 Parts-based Image Decomposition: Shapes and Faces

#### 3.1.1 Overview

In this section we demonstrate MCVQ’s ability to learn a parts-based representation of digital images, using two different data sets. The parts learned by MCVQ are fixed subsets of the data dimensions, corresponding to fixed regions of the images. MCVQ does not attempt to compensate for transformations (e.g. translation, rotation, scale) of the object in the image window, thus is restricted to images that have been normalized for variations in pose. The first data set consists of artificially generated images of shapes, while the second consists of approximately normalized human faces.

Reconstruction performance is used to evaluate the quality of the learned models. By reconstruction we mean, given a data example, using the model to generate a new example that is as similar as possible to the original. The fidelity with which we can reconstruct data examples (both training and testing) reveals how well the model captures the essential features of the data distribution.

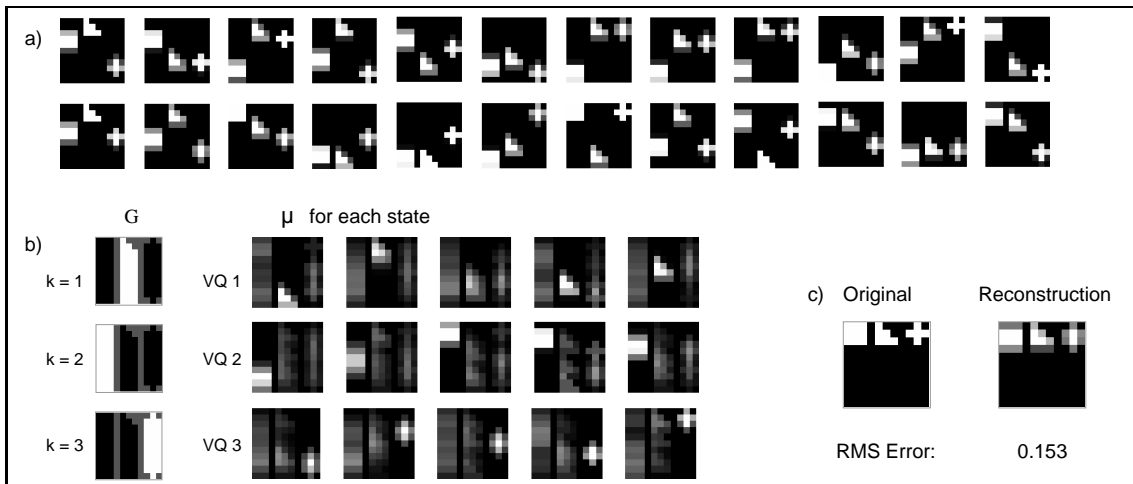


Figure 3.1: a) A sample of 24 training images from the Shapes dataset. b) A typical representation learned by MCVQ with 3 VQs and 5 states per VQ. c) Reconstruction of a test image: original (left) and reconstruction (right).

### 3.1.2 Shape Images

The first dataset used to test our model consisted of  $11 \times 11$  gray-scale images, as pictured in Fig. 3.1a. Each image in the set contains three shapes: a box, a triangle, and a cross. The horizontal position of each shape is fixed, but the vertical position is allowed to vary, uniformly and independently of the positions of the other shapes. Using nine possible locations for each shape, we generated a data set of  $9^3 = 729$  shape images.

A model containing 3 VQs, 5 states each, was trained on 100 of the shape images. In this experiment, and all experiments reported herein, annealing proceeded linearly from an integer less than  $C$  to 1. The learned representation, pictured in Fig. 3.1b, clearly shows the specialization of each VQ to one of the shapes.

The training set was selected so that none of the examples depict cases in which all three shapes are located near the top of the image. Despite this handicap, MCVQ is able to learn the full range of shape positions, and can accurately reconstruct such an image (Fig. 3.1c). In contrast, standard unsupervised methods such as Vector Quantization

(Fig. 3.2a) and Principal Component Analysis (Fig. 3.2b) produce holistic representations of the data, in which each basis vector tries to account for variation observed across the entire image. Non-negative matrix factorization does produce a parts-based representation (Fig. 3.2c), but captures less of the data’s structure. Unlike MCVQ, NMF does not group related parts, and its generative model does not limit the combination of parts to only produce valid images. For example, the NMF model could readily generate an image with two or more triangles, while the MCVQ model could not.

Cooperative Vector Quantization, trained using the same model size as MCVQ, is able to capture much of the part-structure. Figure 3.2d shows an example of a CVQ model, where each circled set of five images depicts the states learned by one of the three VQs. As can be seen in the third VQ (the five right-most vectors) the state vectors do contain additive and subtractive global features, similar to those found by PCA. Unlike MCVQ, these global effects are not “masked away” by a per-pixel stochastic selection. When training CVQ using the variational algorithm given in [10], we have found it to frequently converge to local minima that poorly describe the parts-based structure.

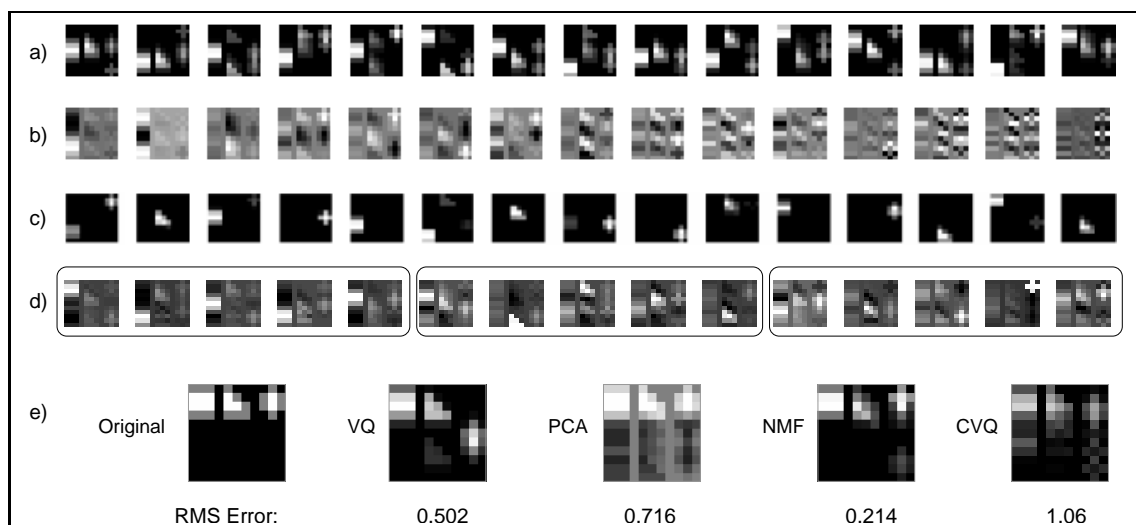


Figure 3.2: Other methods trained on shape images: a) VQ, b) PCA, c) NMF, and d) CVQ. e) Reconstruction of a test image by the three methods (cf. Fig. 3.1c).

As an empirical comparison, we tested the reconstruction error of each of the aforementioned methods on an independent test set consisting of the remaining 629 images. Since each method has one or more free parameters (e.g. the # of principal components) we chose to relate models with similar description lengths. We informally define description length to be the number of bits required to represent the model, plus the number of bits to encode all the test examples using the model. This metric balances the large model cost and small encoding cost of VQ/MCVQ with the small model cost and large encoding cost of PCA/NMF. Specifically, using  $V$  basis vectors or states, PCA/NMF models contain  $V$  real-valued vectors, while VQ models contain  $2V$ , and MCVQ models contain  $2V + K$ , where  $K$  is the number of VQs. To encode a single example, by indicating the most likely states to have generated it, VQ and MCVQ require only  $\log_2 V$  and  $K \log_2(V/K)$  bits respectively. On the other hand, PCA and NMF require a vector of  $V$  real numbers to encode each example (PCA focuses only on minimizing the cost of reconstruction errors [14]).

Model	Parameters	RMS Error
MCVQ	3 VQs, 12 states	0.21
PCA	12 components	0.22
MCVQ	2 VQs, 18 states	0.26
MCVQ	4 VQs, 9 states	0.26
MCVQ	8 VQs, 4 states	0.34
MCVQ	6 VQs, 6 states	0.35
NMF	12 basis vectors	0.35
VQ	38 vectors	0.49

Table 3.1: Average root-mean-squared reconstruction error for various models trained on the shapes image data.

Using a description length of about  $5.9 \times 10^5$  bits, and pixel values ranging from -1



to 1, the average root-mean-squared reconstruction error was calculated, and is shown in Table 3.1. Note that this metric may be useful in determining the number of VQs. We computed the r.m.s. error for various MCVQ model sizes, keeping the total number of states (nearly) the same. The model with 3 VQs, the correct number for this data set, achieves the lowest reconstruction error of all the MCVQ models.

### 3.1.3 Face Images

As a more interesting visual application, we trained our model on the face images from the CBCL Face database #1 [1]. The dataset consists of  $19 \times 19$  gray-scale images, each containing a single frontal or near-frontal face. An MCVQ model of 6 VQs with 12 states each was trained on 2000 of the training images, requiring 15 iterations of EM to converge. As with shape images, the model learned a parts-based representation of the faces.

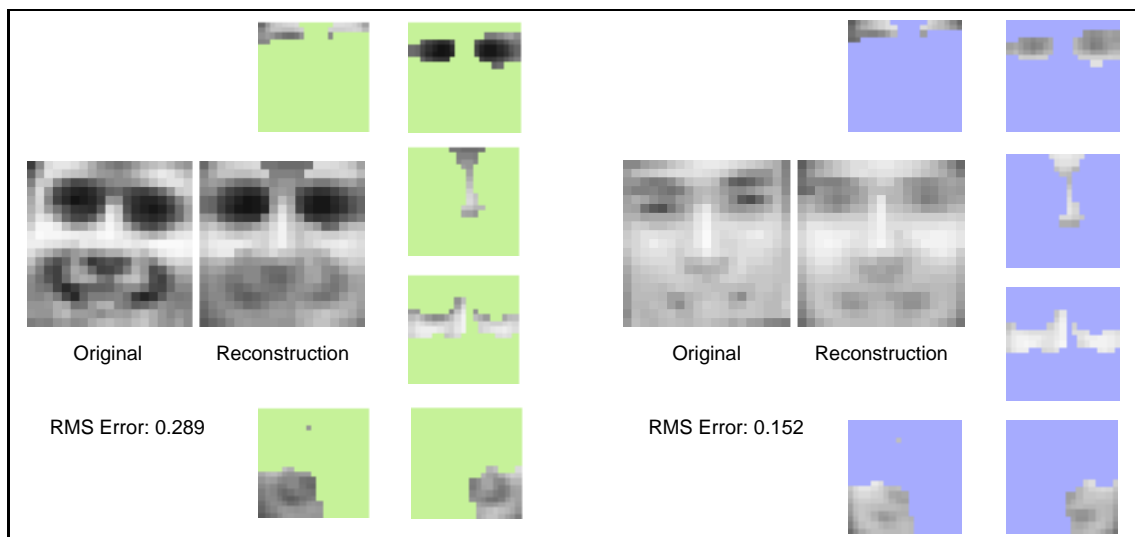


Figure 3.3: The reconstruction of two test images from the Faces dataset. Beside each reconstruction are the parts—restricted for simplicity to the most active state in each of the six VQs—used to generate it. Each part  $j \in k$  is represented by its gated prediction  $(g_{dk} * m_{kj})$  for each image pixel  $i$ .

The reconstruction of two test images, along with the specific parts used to generate each, is illustrated in Fig. 3.3. It is interesting to note that the pixels comprising a single part need not be physically adjacent (e.g. the eyes) as long as their appearances are correlated.

We again compared the reconstruction error of MCVQ with VQ, PCA, and NMF. The training and testing sets contained 1800 and 629 images respectively<sup>1</sup>. Using a description length of  $1.5 \times 10^6$  bits, and pixel values ranging from -1 to 1, the average root-mean-squared reconstruction error was 0.12 for PCA, 0.20 for NMF, 0.23 for MCVQ (both 3 and 6 VQs), and 0.28 for VQ.

## 3.2 Collaborative Filtering

The application of MCVQ to image data assumes that the images are normalized, i.e., that the head is in a similar pose in each image. Normalization can be difficult to achieve in some image contexts; however, in many other types of applications, the input representation is more stable. For example, many information retrieval applications employ bag-of-words representations, in which a given word always occupies the same input element.

We test MCVQ on a collaborative filtering task, utilizing the EachMovie dataset, where the input vectors are ratings by users of movies, and a given element always corresponds to the same movie. The original dataset contains ratings, on a scale from 1 to 6, of a set of 1649 movies, by 74,424 users. In order to reduce the sparseness of the dataset, since many users rated only a few movies, we only included users who rated at least 75 movies and movies rated by at least 126 users, leaving a total of 1003 movies and

---

<sup>1</sup>The CBCL Face database #1 contains standard training and test sets of 2429 and 472 images respectively. Unfortunately the standard testing images are significantly different from the training images; the test faces appear at different scales, rotations, and levels of noise than the training faces. To compensate for this, we restricted our attention to the standard training set, randomly selecting images for our new data split.

	VQ 1	VQ 2	VQ 3	VQ 4
User 1	<b>The Fugitive</b> 5.8 (6)	<b>Pulp Fiction</b> 5.5 (4)	<b>Cinema Paradiso</b> 5.6 (6)	<b>Shawshank Redemption</b> 5.5 (5)
	<b>Terminator 2</b> 5.7 (5)	<b>Godfather: Part II</b> 5.3 (5)	<b>Touch of Evil</b> 5.4 (-)	<b>Taxi Driver</b> 5.3 (6)
	<b>Robocop</b> 5.4 (5)	<b>Silence of the Lambs</b> 5.2 (4)	<b>Rear Window</b> 5.2 (6)	<b>Dead Man Walking</b> 5.1 (-)
	Kazaam 1.9 (-)	Brady Bunch Movie 1.4 (1)	Jean de Florette 2.1 (3)	Billy Madison 3.2 (-)
	Rent-a-Kid 1.9 (-)	Ready to Wear 1.3 (-)	Lawrence of Arabia 2.0 (3)	Clerks 3.0 (4)
	Amazing Panda Adventure 1.7 (-)	A Goofy Movie 0.8 (1)	Sense & Sensibility 1.6 (-)	Forrest Gump 2.7 (2)
User 2	<b>Best of Wallace &amp; Gromit</b> 5.6 (-)	<b>Tank Girl</b> 5.5 (6)	<b>Mediterraneo</b> 5.3 (6)	<b>Sling Blade</b> 5.4 (5)
	<b>The Wrong Trousers</b> 5.4 (6)	<b>Showgirls</b> 5.3 (4)	<b>Three Colors: Blue</b> 4.9 (5)	<b>One Flew ... Cuckoo's Nest</b> 5.3 (6)
	<b>A Close Shave</b> 5.3 (5)	<b>Heidi Fleiss...</b> 5.2 (5)	<b>Jean de Florette</b> 4.9 (6)	<b>Dr. Strangelove</b> 5.2 (5)
	Robocop 2.6 (2)	Talking About Sex 2.4 (5)	Jaws 3-D 2.2 (-)	The Beverly Hillbillies 2.0 (-)
	Dangerous Ground 2.5 (2)	Barbarella 2.0 (4)	Richie Rich 1.9 (-)	Canadian Bacon 1.9 (4)
	Street Fighter 2.0 (-)	The Big Green 1.8 (2)	Getting Even With Dad 1.5 (-)	Mrs. Doubtfire 1.7 (-)

Figure 3.4: The MCVQ representation of two test users in the EachMovie dataset. The 3 most conspicuously high-rated (bold) and low-rated movies by the most active states of 4 of the 8 VQs are shown, where conspicuousness is the deviation from the mean rating for a given movie. Each state’s predictions,  $\mu_{dkj}$ , can be compared to the test user’s true ratings (in parentheses); the model’s prediction is a convex combination of state predictions. Note the intuitive decomposition of movies into separate VQs, and that different states within a VQ may predict very different rating patterns for the same movies.

5831 users. The remaining dataset was still very sparse, as the maximum user rated 928 movies, and the maximum movie was rated by 5401 users. We split the data randomly into 4831 users for a training set, and 1000 users in a test set. We ran MCVQ with 8 VQs and 6 states per VQ on this dataset. An example of the results, after 18 iterations of EM, is shown in Fig. 3.4.

Note that in the MCVQ graphical model (Fig. 2.1), all the observation dimensions are leaves, so an input variable whose value is not specified in a particular observation vector will not play a role in inference or learning. This makes inference and learning with sparse data rapid and efficient.

We compare the performance of MCVQ on this dataset to the aspect model. We

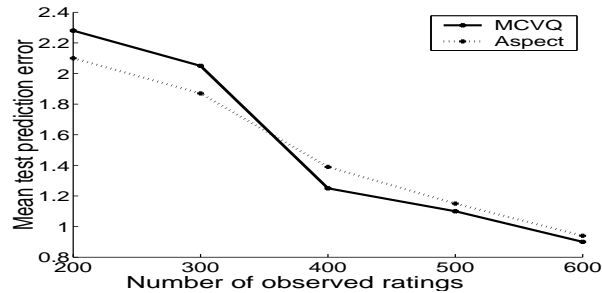


Figure 3.5: The average absolute deviation of predicted and true values of held-out ratings is compared for MCVQ and the aspect model. Note that the number of users per  $x$ -bin decreases with increasing  $x$ , as a user must rate at least  $x + 1$  movies to be included.

implemented a version of the aspect model, with 50 aspects and truncated Gaussians for ratings, and used “tempered EM” (with smoothing) to fit the parameters[17]. For both models, we train the model on the 4831 users in the training set, and then, for each test user, we let the model observe some fixed number of ratings and hold out the rest. We evaluate the models by measuring the absolute difference between their predictions for a held-out rating and the user’s true rating, averaged over all held-out ratings for all test users (Fig. 3.5).

For further analysis of MCVQ applied to collaborative filtering tasks, the reader is referred to [5].

### 3.3 Document Modeling

MCVQ can also be used for information retrieval from text documents, by employing the bag-of-words representation. We present preliminary results on the NIPS corpus <sup>2</sup>, which consists of the full text of the NIPS conference proceedings, volumes 0 to 12. The data was pre-processed to remove common words (e.g. the), and those appearing in fewer than five documents, resulting in a vocabulary of 14,265 words. For each of the 1740 papers in the corpus, we generated a vector containing the number of occurrences of each

<sup>2</sup>Available at <http://www.cs.toronto.edu/~roweis/data.html>

Predictive Sequence Learning in Recurrent Neocortical Circuits R. P. N. Rao & T. J. Sejnowski				The Relevance Vector Machine Michael E. Tipping			
<b>afferent</b>	<b>ekf</b>	<b>latent</b>	<b>ltp</b>	<b>svms</b>	<b>hme</b>	<b>similarity</b>	<b>extraction</b>
<b>lgn</b>	<b>niranjan</b>	<b>som</b>	<b>gerstner</b>	<b>svm</b>	<b>svr</b>	<b>classify</b>	<b>net</b>
<b>interneurons</b>	<b>freitas</b>	<b>detection</b>	<b>zador</b>	<b>margin</b>	<b>svs</b>	<b>classes</b>	<b>weights</b>
<b>excitatory</b>	<b>kalman</b>	<b>search</b>	<b>soma</b>	<b>kernel</b>	<b>hyperparameters</b>	<b>classification</b>	<b>functions</b>
<b>membrane</b>	<b>wp</b>	<b>data</b>	<b>depression</b>	<b>risk</b>	<b>kopf</b>	<b>class</b>	<b>units</b>
query	critic	mdp	spline	jutten	chip	barn	mdp
documents	stack	pomdps	tresp	pes	ocular	correlogram	pomdps
chess	suffix	prioritized	saddle	cpg	retinal	interaural	littman
portfolio	nuclei	singh	hyperplanes	axon	surround	epsp	prioritized
players	knudsen	elevator	tensor	behavioural	cmos	bregman	pomdp

Figure 3.6: The representation of two documents by an MCVQ model with 8 VQs and 8 states per VQ. For each document we show the states selected for it from 4 VQs. The bold (plain) words for each state are those most conspicuous by their above (below) average predicted frequency.

word in the vocabulary. These vectors were normalized so that each contained the same number of words. A model of 8 VQs, 8 states each, was trained on the data, converging after 15 iterations of EM. A sample of the results is shown in Fig. 3.6.

When trained on text data, the values of  $\{g_{dk}\}$  provide a segmentation of the vocabulary into subsets of words with correlated frequencies. Within a particular subset, the words can be positively correlated, indicating that they tend to appear in the same documents, or negatively correlated, indicating that they seldom appear together.

### 3.4 Classification: Face Expressions

As motivated in the introduction to this thesis, parts-based representations are useful for feature extraction - to produce a new representation of the data which may be more useful for subsequent supervised learning. In this capacity, we apply MCVQ to the supervised classification of facial expressions.

### 3.4.1 AR Face Database

The following experiments were performed using a subset of the AR Face Database [23], which consists of images of frontal faces of 126 subjects under a number of different conditions. The five conditions included in our data set were: 1) anger, 2) neutral, 3) scream, 4) smile, and 5) sunglasses. We grouped our data into five classes, based on the condition depicted.

The data set in its raw form contains faces which, although roughly centred, appear at different locations, angles, and scales. Since MCVQ does not attempt to compensate for these differences, we manually aligned each face such that the eyes always appeared in the same location. Next we cropped the images tightly around the face, and subsampled to reduce the size to  $29 \times 22$  pixels. Finally, we converted the image data to grayscale, with pixel values ranging from -1 to 1. Examples of the preprocessed images can be seen in Figure 3.7.



Figure 3.7: Examples of preprocessed images from the AR Face Database. These images, from left to right, correspond to the conditions: anger, neutral, scream, smile, and sunglasses.

From this data, training and testing sets containing 1000 and 275 examples respectively were constructed. Each set contained an equal number of images from all of the classes.

### 3.4.2 MCVQ for feature selection

In this experiment we investigated the suitability of MCVQ as a dimensionality reduction technique, with a goal of learning a discriminative set of features to assist subsequent classification. We began by learning an MCVQ model with 15 VQs, 11 states each, from the training set. To represent each training example, we used the posterior probabilities of the state-selection variables, the  $m_{kj}$ 's, as estimated using our variational approximation. This reduced representation consisted of  $15 \times 11$  dimensions, but only  $15 \times 10$  degrees of freedom.

Using the reduced representation of our data, we trained a support vector machine classifier, specifically a  $\nu$ -SVM [29, 28] with a Gaussian kernel. The parameters for the experiment, chosen using 10-fold cross validation, were  $\nu = 0.8$  and  $\sigma^2 = 330 \approx 2 \times (15 \times 11)$ .

During testing we used the MCVQ model to estimate the  $m_{kj}$ 's of the test examples, and classified them using the SVM. The classification rate was 0.77 (212 correct of 275).

#### Alternate classification techniques

As baseline comparisons, we attempted the same classification task using a support vector machine trained on the pixel values only, and using a human subject.

For the support vector machine, we again used a  $\nu$ -SVM with Gaussian kernel and parameters selected by cross validation. The specific parameters employed were  $\nu = 0.4$  and  $\sigma^2 = 255 \approx 0.4 \times \#$  pixels per image. The SVM achieved a classification rate was 0.84 (230 of 275 correct).

The human subject, after studying the labelled training data, was able to correctly classify 0.77 (213 of 275) of the test images.

In both cases, the vast majority of the errors were caused by *anger* images being misclassified as *neutral*, or vice versa. (76% of the errors for the SVM and 85% for the human.)

## Discussion

In this experiment we have observed that the reduced representation learned by MCVQ contains most of the class-discriminative information contained in the original data set. Classification performance using MCVQ's learned features equaled that of a human on this difficult task, despite using less than one quarter as many degrees of freedom as are present in the original images.

Increasing the number of VQs and states in the model did not improve classification performance.

A possible explanation for the superior performance of the SVM on raw pixel values is that the experimental setup, namely using well-normalized images without occlusion, does not leverage the advantages of parts-based classifiers, as previously described.



# Chapter 4

## Discovering dependencies among part selections

### 4.1 Introduction

The multiple cause vector quantization model consists of a number of vector quantizers which compete amongst themselves to describe each dimension of a high-dimensional data space. The generative process it proposes to produce an observed data vector includes a number of latent selections. First, each VQ selects a single state as a proposed data vector. Then, for each dimension, one of the VQs is selected to “write to” the actual data vector (with additive Gaussian noise).

Although MCVQ assumes these selections to be independent, clearly in real world data there are dependencies. For example, consider the case of modeling human faces. As shown in 3.1, MCVQ can be trained such that each VQ captures a different part or region of the face, and each state represents a possible appearance of that part. If one part were to select an appearance with high pixel intensities, due to lighting conditions or skin tone, then it seems likely that the other parts should appear similarly light.

In this chapter we propose a method of learning these dependencies between part

selections by introducing an additional higher-level cause, or “class” variable, on which the state selections are conditioned. We examine two settings: first we consider the case in which the class variable is observed, corresponding to labeled training data, and second we consider the case in which it is unobserved, and must be inferred.

In the observed case this allows us to learn distributions over state selections based on the class of the data, and to classify new data based on the inferred posterior over state selections. In the unobserved case, we learn a probabilistic clustering over data cases, based on their state selections. When treated generatively, both of these cases allow us, given only a single vocabulary of parts, to produce data from any of the classes by appropriately setting the new variable.

## 4.2 Model

A multiple cause vector quantization model consists of  $K$  vector quantizers, each with  $J$  states. A state is made up of a mean vector and a diagonal covariance matrix, specifying a Gaussian distribution in data space. To generate a data vector  $\mathbf{x} \in \mathfrak{R}^D$  from a given model, a state is stochastically selected for each VQ, and a VQ is selected for each dimension  $x_d$  of the data vector. We represent these selections using the binary latent variables  $\mathbf{R} = \{r_{dk}\}$  and  $\mathbf{S} = \{s_{kj}\}$ , exactly as described in section 2.2.

Suppose that each data vector comes from one of  $N$  different classes, for some positive integer  $N$ . We assume that for a given data vector the selections of the states for each VQ (the  $\mathbf{S}$ 's) depend on the class, but that the selections of VQ per data dimension (the  $\mathbf{R}$ 's) do not. Specifically, for training case  $\mathbf{x}$ , we introduce a new multinomial variable  $\mathbf{y}$  which selects exactly one of the  $N$  classes.  $\mathbf{y}$  can be thought of as a collection of binary variables  $\{y_n\}$  for  $n = 1 \dots N$  or equivalently as an indicator vector  $\mathbf{y} \in \{0, 1\}^N$ , where  $y_n = 1$  if and only if class  $n$  has been selected. Using a Multinomial( $\boldsymbol{\beta}$ ) prior distribution for  $\mathbf{y}$ , the prior over selections takes the following form (cf. section 2.3):

$$\begin{aligned}
P(\mathbf{R}, \mathbf{S}, \mathbf{y}) &= P(\mathbf{R})P(\mathbf{S}|\mathbf{y})P(\mathbf{y}) \\
&= \left( \prod_{dk} g_{dk}^{r_{dk}} \right) \left( \prod_{nkj} b_{nkj}^{s_{kj} y_n} \right) \left( \prod_n \beta_n^{y_n} \right)
\end{aligned} \tag{4.1}$$

The graphical model representation is given in Figure 4.1.

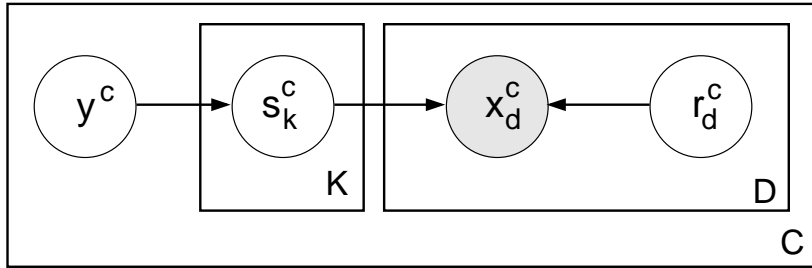


Figure 4.1: Graphical model with additional class variable,  $\mathbf{y}^c$ , where  $c$  is an index over the  $C$  cases in our training data. Note that  $\mathbf{y}^c$  can be either observed or unobserved.

Note that for each class  $n$  there is a different prior distribution over the state selections. We represent these distributions with  $\{b_{nkj}\}_{nkj}$ , where  $b_{nkj}$  is the probability of selecting state  $j$  from VQ  $k$  given class  $n$ . Since the model contains  $N$  priors over  $\mathbf{S}$ , one to be selected for each data vector, then we can think of this model as incorporating an additional vector quantization, this time over distributions for  $\mathbf{S}$ .

If the class variable  $\mathbf{y}$  corresponds to an observed label for each data vector, then the new model can be thought of as a standard MCVQ model, but with the requirement that we learn a different prior over  $\mathbf{S}$  for each class. On the other hand, if  $\mathbf{y}$  is unobserved, then the new model learns a VQ over the space of possible priors for  $\mathbf{S}$ , rather than the single maximum likelihood point estimate learned in standard MCVQ.

### 4.2.1 Unsupervised Case

In the unsupervised case,  $\mathbf{y}^c$ , for each training case  $c = 1 \dots C$  is an unobserved variable, like  $\mathbf{R}^c$  and  $\mathbf{S}^c$ . As in standard MCVQ, the posterior  $P(\mathbf{R}, \mathbf{S}, \mathbf{y}|\mathbf{x}, \theta)$  over latent vari-

ables cannot tractably be computed. Instead, we use the following mean-field variational approximation:

$$Q(\mathcal{R}, \mathcal{S}, \mathcal{Y}) = \left( \prod_{cdk} g_{dk}^{r_{dk}^c} \right) \left( \prod_{cnkj} m_{nkj}^c s_{kj}^{y_n^c} \right) \left( \prod_{cn} z_n^c y_n^c \right)$$

Note that, as in standard MCVQ, we restrict the posterior selections of VQ made for each training case to be identical, i.e.  $P(\mathbf{R}^1) = P(\mathbf{R}^2) = \dots = P(\mathbf{R}^c)$ .

Using the variational posterior, the prior (4.1), and the standard likelihood, we obtain the following expression for the Free Energy:

$$\begin{aligned} \mathcal{F} &= -E_Q[\log P(\mathcal{X}, \mathcal{R}, \mathcal{S}, \mathcal{Y}|\theta)] - \mathcal{H}(Q) \\ &= -E_Q \left[ \sum_c \log P(\mathbf{x}^c | \mathbf{R}^c, \mathbf{S}^c, \mathbf{y}^c, \theta) + \sum_c \log P(\mathbf{R}^c, \mathbf{S}^c, \mathbf{y}^c) \right] - \mathcal{H}(Q) \\ &= \sum_{cdk} g_{dk} \log \frac{g_{dk}}{a_{dk}} + \sum_{cnkj} m_{nkj}^c z_n^c \log \frac{m_{nkj}^c}{b_{nkj}} + \sum_{cn} z_n^c \log \frac{z_n^c}{\beta_n} \\ &\quad + \sum_{cdkj} g_{dk} \left( \sum_n m_{nkj}^c z_n^c \right) \varepsilon_{dkj}^c + \frac{CD}{2} \log(2\pi) \end{aligned}$$

By differentiating  $\mathcal{F}$  with respect to each of the parameters and latent variables, and solving for their respective minima, we obtain the EM updates used for learning the model.

The E-step updates for  $m_{nkj}^c$  and  $z_n^c$  are:

$$\begin{aligned} m_{nkj}^c &\propto b_{nkj} \exp \left( - \sum_d g_{dk} \varepsilon_{dkj}^c \right) \\ z_n^c &\propto \beta_n \exp \left( \sum_{kj} m_{nkj}^c \log \frac{b_{nkj}}{m_{nkj}^c} - \sum_{dkj} g_{dk} m_{nkj}^c \varepsilon_{dkj}^c \right) \end{aligned}$$

The M-step updates for  $a_{dk}$ ,  $g_{dk}$ ,  $\mu_{dkj}$ , and  $\sigma_{dkj}$  are unchanged from standard MCVQ, except for the substitution of  $\sum_n (m_{nkj}^c z_n^c)$  in place of  $m_{kj}^c$ , wherever it appears. The updates for  $\beta_n$  and  $b_{nkj}$  are:

$$\beta_n = \frac{1}{C} \sum_c z_n^c \qquad b_{nkj} = \frac{\sum_c m_{nkj}^c z_n^c}{\sum_c z_n^c}$$

A useful interpretation of the latent variable  $\mathbf{y}$  is that, for a given data vector, it indicates the assignment of that datum to one of  $N$  clusters. Specifically,  $z_n^c$  can be thought of as the posterior probability that example  $c$  belongs to cluster  $n$ .

### 4.2.2 Supervised Case

In the supervised case, we are given a set of labelled training data  $(\mathbf{x}^c, \mathbf{y}^c)$  for  $c = 1 \dots C$ . Note that this case is essentially the same as the unsupervised case - we can obtain the supervised updates by constraining  $\mathbf{z}^c = \mathbf{y}^c$ , for  $c = 1 \dots C$ . Since the class is known, we may now drop the subscript  $n$  from  $m_{nkj}^c$ .

As stated earlier, when the  $\mathbf{y}$ 's are given, we learn a different prior over state selections for each class  $n$ :

$$b_{nkj} = \frac{1}{C\beta_n} \sum_c m_{kj}^c y_n^c$$

where the prior probability of observing class  $n$ ,  $\beta_n$ , can be calculated from the training labels:

$$\beta_n = \frac{1}{C} \sum_c y_n^c$$

The posterior probabilities of selecting each state to have generated example  $c$ ,  $m_{kj}^c$ , depends only on the prior corresponding to  $c$ 's class.

$$m_{kj}^c \propto b_{y^c kj} \exp\left(-\sum_d g_{dk} \varepsilon_{dkj}^c\right)$$

From the probability model described above we can derive a method for classifying new test cases. Given an unlabelled testing vector,  $\mathbf{x}$ , and a model  $\theta$ , we can estimate the most likely class  $\hat{\mathbf{y}}$  using the rule:

$$\begin{aligned}
\hat{\mathbf{y}} &= \operatorname{argmax}_{\mathbf{y}} P(\mathbf{x}, \mathbf{y}, |\theta) \\
&= \operatorname{argmax}_{\mathbf{y}} \sum_{\mathbf{R}} \sum_{\mathbf{S}} P(\mathbf{x}|\mathbf{R}, \mathbf{S}, \theta) P(\mathbf{R}, \mathbf{S}, \mathbf{y})
\end{aligned}
\tag{4.2}$$

This rule poses a computational difficulty, since it requires we sum over all possible configurations of the latent selection variables  $\mathbf{R}$  and  $\mathbf{S}$ . For a model with  $K$  VQs,  $J$  states each, operating on  $D$ -dimensional data, the above sum would contain  $K^D J^K$  terms, which is intractable for all but the smallest settings of  $J, K$ , and  $D$ .

An approach for dealing with this summation is to use a Monte Carlo approximation. If we draw a set  $\{(\mathbf{R}_q, \mathbf{S}_q)\}_q$  of  $Q$  samples from the prior distribution  $P(\mathbf{R}, \mathbf{S}|\mathbf{y})$ , we can approximate the summation with  $P(\mathbf{x}, \mathbf{y}|\theta) \approx \sum_q P(\mathbf{x}|\mathbf{R}_q, \mathbf{S}_q, \theta) P(\mathbf{y})/Q$ , giving us the approximate classification rule:

$$\hat{\mathbf{y}} = \operatorname{argmax}_{\mathbf{y}} \frac{P(\mathbf{y})}{Q} \sum_q P(\mathbf{x}|\mathbf{R}_q, \mathbf{S}_q, \theta)
\tag{4.3}$$

## 4.3 Experiments

In this section we present experimental results obtained by training the above models on images taken from the AR Face Database. The data is described in detail in section 3.4.

### 4.3.1 Supervised Case

#### Learning class-conditional priors

In our first experiment we trained a supervised model on the entire data set, with a goal of learning a different prior over state selections for each of the five classes of image (anger, neutral, scream, smile, and sunglasses).

The model consisted of 5 VQs, 10 states each. The image regions that each VQ learned to explain are shown in Figure 4.2. For each VQ  $k$  we have plotted, as a grayscale image,

the prior probability of each pixel selecting  $k$  (i.e.  $a_{dk}$ ,  $d = \text{pixel index}$ ). The two regions which we expected to be the most class discriminative, the mouth and the eyes, were captured by VQs 3 and 4 respectively.



Figure 4.2: Image regions explained by each VQ. The prior probability of a VQ being selected for each pixel is plotted as a gray value between 0=black and 1=white.

The priors over state selection for these VQs varied widely depending on the class of the image being considered. As an example, in Table 4.1 we have plotted the prior probability, given the class, of selecting each of the states from VQ 3. In the figure we can see that the prior probabilities closely matched our intuition as to which mouth shapes corresponded to which facial expressions. For example the first state (from the top) appears to depict a smiling mouth. Accordingly, the *smile* class assigned it the highest prior probability, 0.34, while the other classes each assigned it 0.05 or less. Also, if we examine the *scream* column, we see that the highest prior probabilities were assigned to the second and tenth states - both widely screaming mouths.

Note that for *sunglasses*, which does not presuppose a mouth shape, the prior showed a preference for the more *neutral* mouths. The explanation for this is simply that most subjects in the data adopted a neutral expression when wearing sunglasses.

One method for qualitatively evaluating the suitability of a class-conditional prior is to use it to generate novel images from the model, and see how well they match the class. Samples drawn this way using each of the five priors can be seen in Figure 4.3.

In addition to learning a prior for each class, this framework allows us the flexibility to generate images from “hybrid” classes by combining the learned priors. For example,

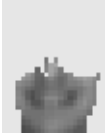
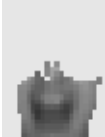

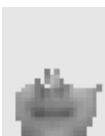
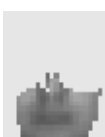
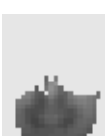
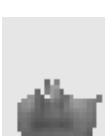
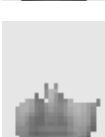
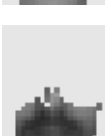
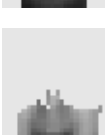
State Mean	anger	neutral	scream	smile	sunglasses
	0.03	0.03	0.02	0.34	0.05
	0.02	0.01	0.36	0.02	0.02
	0.05	0.15	0.01	0.11	0.11
	0.18	0.22	0.02	0.17	0.20
	0.19	0.20	0.01	0.02	0.19
	0.12	0.09	0.23	0.18	0.10
	0.23	0.18	0.01	0.03	0.15
	0.08	0.04	0.04	0.02	0.10
	0.07	0.06	0.04	0.07	0.06
	0.02	0.02	0.26	0.04	0.02

Table 4.1: Class conditional priors over mouth selections. Each column represents one of the states from VQ 3. The rows show, for each class, the prior probability of selecting each state from VQ 3. The images displayed are the mean of each state, masked (multiplied) by the prior probability of each pixel selecting VQ 3.



the *sunglasses* images in the training set all have neutral mouth shapes. By combining the priors for *sunglasses* and *scream*, we were able to define a new class. This new prior was created by averaging the *sunglasses* and *scream* priors for VQs 1,2, and 5, using the *scream* prior for VQ 3, and using the *sunglasses* prior for VQ 4. An image randomly generated using the hybrid prior is shown in Figure 4.3.



Figure 4.3: On the left are examples generated from each of the five priors learned, one for each class. On the right is an image generated using a combination of the priors for *scream* and *sunglasses* (see text). (The images do not include the Gaussian noise described in the generative process.)

### MCVQ classifier

Our second experiment tested the effectiveness of the MCVQ classifier proposed in section 4.2.2. Using the training data we constructed an MCVQ model with observed class variables. Applying the approximate classification rule given by (4.3), and using 10000 samples, a classification rate was obtained. The experiment was repeated for different numbers of VQs, keeping the total number of states fixed at 150. The results have been tabulated below:

# VQs	States per VQ	Classification Rate
6	25	$169/275 \approx 0.61$
15	10	$165/275 = 0.60$
25	6	$173/275 \approx 0.63$

As stated in section 3.4, a support vector machine trained on raw pixel values achieves 0.84 accuracy in this classification experiment.

### 4.3.2 Unsupervised Case

To evaluate the performance of the unsupervised model, we trained a model on a subset of the five classes, with the hope that it would learn clusters corresponding to the original classes.

The data set was restricted to contain only three classes - *neutral*, *scream*, and *sunglasses* - totalling 765 images. To prevent the images from being clustered simply by their overall brightness levels, for this experiment we performed equalization of the intensity histogram for each of the training images. The model we trained consisted of 15 VQs, 10 states each, and the latent class variable had 3 settings (i.e. 3 clusters).

In Table 4.2 we see the relationship between learned clusters and classes. The first cluster corresponded to the *sunglasses* class, containing all but two of the *sunglasses* images. The second and third clusters contained approximately equal numbers of *neutral* and *scream* images. Closer examination revealed that, of those not wearing sunglasses, 88% of the males had been placed in cluster 2, and 80% of the females in cluster 3. Examples of training images assigned to each of the clusters are shown in Figure 4.4.

cluster	1	2	3	cluster	1	2	3
neutral	0	154	101	female	112	46	184
scream	0	139	116	male	141	247	35
sunglasses	253	0	2	totals	253	293	219
totals	253	293	219				

Table 4.2: Number of images from each class assigned to each cluster. We consider an image to belong to the cluster with the highest posterior probability ( $z_n^c$ ).



Figure 4.4: Two training examples randomly selected from each of the three clusters.

## 4.4 Discussion

In this chapter we have presented an extension to MCVQ that allows higher-level causes or relationships to be learned from the data. Specifically, assuming the data comes from a pre-specified number of classes, our extension models the relationships between data vectors, based on the state selections each class favours in an MCVQ model.

Given a set of labelled data, such as facial images classified by the expression of the subject, we were able to learn a single vocabulary of parts, and the likelihood of each part appearing in images of a given class. These probabilities are of interest since, by applying Bayes' rule, we can discover how the possible states for each feature affect what class a data vector will belong to.

We proposed a supervised classification method based on using these class-conditional likelihoods for discrimination. Although the method was moderately successful at classifying facial expressions, its performance was poor in comparison with other standard techniques.

Finally, we have shown that when the data are not labelled, we can learn a clustering of the data into classes while simultaneously learning the relationships described above.

# Chapter 5

## Conclusions

### 5.1 Discussion

Parts-based representations provide an efficient framework for modelling vector-valued data. They allow a large number of patterns to be described using selections from a small number of discrete alternatives. Learning parts from data can provide valuable insights into the causes that interact to generate the observations.

We have presented Multiple Cause Vector Quantization, an unsupervised method for learning parts-based representations of data. It improves upon related methods in that it can be trained efficiently, is able to group learned components into alternative appearances of the same part, and can be applied to any type of vector-valued data.

Like many clustering methods, MCVQ requires the model size, namely the number of VQs and the number of states per VQ, to be specified a priori.

When trained on image data, each VQ learns to model a local feature. The locality property is obtained without the need to explicitly include it in our prior distributions. In fact it would be undesirable to do so, since in certain instances the model can be improved by grouping two or more disjoint regions, such as the eyes of a face, in the same part.

MCVQ is restricted in its visual applications, since it does not attempt to compensate for transformations (e.g. translation, rotation) of the object in the image window. Fortunately, this drawback does not apply to other sorts of data, such as bag-of-words text and preference data.

MCVQ provides a framework suitable for additional levels of learning, such as grouping data vectors by the parts they contain. When used to cluster data in this fashion, MCVQ learns how the various appearances of each part contribute to the resulting label assigned to a data vector.

## 5.2 Future Directions

When experimenting with MCVQ, obtaining good results often required a judicious choice of the model size parameters - most importantly the number of VQs. Ideally we would like these parameters to be learned from the data. In that aim, we would like to investigate the application of Bayesian model selection techniques, such as Variational Bayesian EM [2], to MCVQ.

We conclude with an overview of two additional directions for further research.

### 5.2.1 Continuously Parameterized Part Models

To model the range of appearances of each part, MCVQ uses vector quantization<sup>1</sup>, resulting in a discrete number of possibilities. An alternative to this would be to use a continuously parametrized model, such as factor analysis.

The model proposed by factor analysis (e.g. [11]) assumes that the data  $\mathbf{x} \in \mathfrak{R}^D$  has been generated as a linear combination of  $J$  basis vectors, usually with  $J \ll D$ , plus axis-aligned Gaussian noise. More specifically, if we let  $\Lambda \in \mathfrak{R}^{D \times J}$  be our basis or *factor loading* matrix,  $\mathbf{s} \in \mathfrak{R}^J$  be the latent representation of  $\mathbf{x}$ ,  $\Psi$  be a diagonal covariance

---

<sup>1</sup>or, more accurately, a mixture of Gaussians with axis-aligned covariance

matrix, and  $\mu$  be the mean of the data set, then:

$$\mathbf{x}|\mathbf{s} \sim \mathcal{N}(\Lambda\mathbf{s} + \mu, \Psi)$$

To extend this model, we would allow  $K$  factor analyzers to competitively generate the data, where for each dimension  $d$  of each data vector, one winning factor analyzer (FA) is selected. Using  $r_{dk}$ , as before, as random variables indicating these selections, the likelihood would be:

$$P(\mathbf{x}|\mathbf{R}, \mathbf{S}) = \prod_{dk} \mathcal{N}(x_d ; \Lambda_d^k \mathbf{s}_k + \mu_k, \Psi_d^k)^{r_{dk}}$$

where  $\Lambda_d^k$  indicates the  $d^{th}$  row of the  $k^{th}$  factor loading matrix, and  $\Psi_d^k$  indicates the variance in the  $d^{th}$  dimension of the  $k^{th}$  FA. Following both FA and MCVQ, the priors for the latent variables would be:

$$P(\mathbf{S}) = \prod_{ck} \mathcal{N}(\mathbf{s}_k ; \mathbf{0}, \mathbf{I}) \quad P(\mathbf{R}) = \prod_{dk} a_{dk}^{r_{dk}}$$

This probabilistic model is nearly identical to MCVQ, but allows  $s_k$  to be a linear combination of basis vectors, rather than a stochastic selection. This would provide a significant advantage when, for example, modelling the overall brightness level of a part, which MCVQ must model using a quantization over possible levels.

### 5.2.2 Alternative Variational Approximation

In section 2.3 we advocate the use of a non-standard mean-field variational approximation. Specifically our approximation restricts the posterior probability of VQ selection, for each data dimension, to be the same across all training cases. The typical mean-field approximation would be

$$Q(\mathcal{R}, \mathcal{S}) = \left( \prod_{cdk} g_{dk}^c \right)^{r_{dk}^c} \left( \prod_{ckj} m_{kj}^c \right)^{s_{kj}^c} \quad (5.1)$$

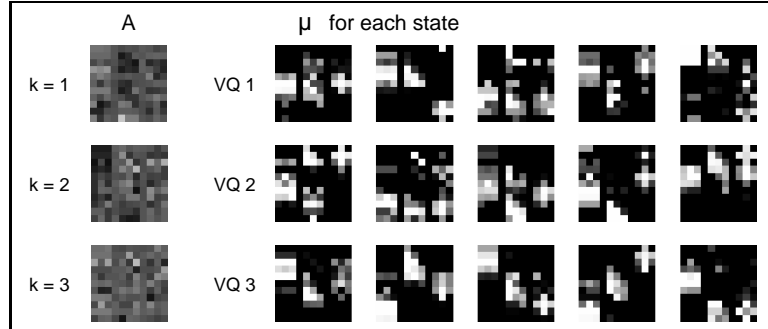


Figure 5.1: Results on the shapes experiment, trained using the conventional mean-field variational posterior, equation (5.1).

(contrast with (2.2)), where we allow the posterior probability  $g_{dk}^c$  of choosing VQ  $k$  for the  $d^{\text{th}}$  dimension to vary for each training case  $c$ . In this setting, the prior probabilities  $a_{dk}$  indicate the association of data dimensions with VQs.

Experiments with this variation gave poor results. The prior distribution over VQ selections  $\mathbf{a}_d$  for each VQ  $d$  had high entropy, thus the VQs did not specialize to different parts of the data. An example of such a model, trained on the shapes data as described in section 3.1, is shown in Figure 5.1.

One approach to improving the parts-based representations learned by this model would be to incorporate a hyper-prior distribution over the VQ selections, the  $r_{dk}$ 's, that would encourage their distributions to have low entropy. An example of such a prior is Brand's *entropic prior* [6, 7]. Specifically, we could add

$$P(\mathbf{A}) \propto \prod_d \exp(-\alpha \text{Entropy}(\mathbf{a}_d)) = \prod_d \exp(\alpha \sum_k a_{dk} \log a_{dk})$$

to the complete likelihood (2.1). This has the effect of adding the term  $-\alpha \sum_{dk} a_{dk} \log a_{dk}$  to the free energy, increasing it by  $\alpha$  times the entropy of the  $\mathbf{a}_d$  distributions.

Unfortunately, given a reasonable amount of data, the standard entropic prior ( $\alpha = 1$ ) has little influence on the free energy, compared with the contribution made by the likelihood. The prior's influence could be increased by using a larger value for the parameter

$\alpha$ . This will lead to models with lower VQ selection entropies, but has the unappealing feature that the prior needs to be adjusted based on the amount of data observed.

Initial experiments using the entropic prior have yielded poor results, in terms of learning a model that captures parts and is suitable for further learning. For example, Figure 5.2 shows a typical model learned during the shapes experiment, using  $\alpha = 15$ .

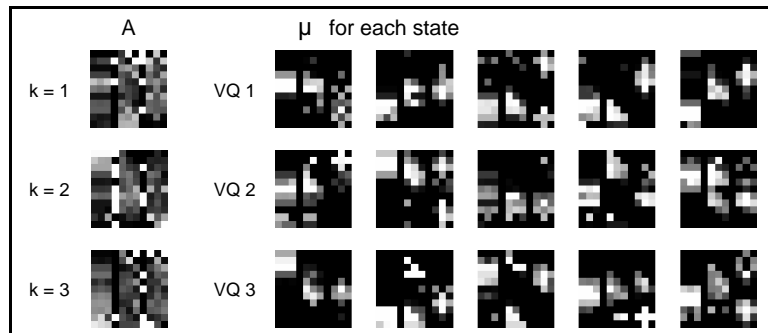


Figure 5.2: Results on the shapes experiment using the entropic prior.

In the future, we plan to investigate methods of improving the performance when using the entropic prior, as well as exploring other alternative variational approximations.



# Bibliography

- [1] CBCL face database #1. MIT Center For Biological and Computation Learning. <http://www.ai.mit.edu/projects/cbcl>.
- [2] M.J. Beal and Z. Ghahramani. The Variational Bayesian EM Algorithm for Incomplete Data: with Application to Scoring Graphical Model Structures. In *Bayesian Statistics 7*, 2003.
- [3] I. Biederman. Recognition-by-Components: A Theory of Human Image Understanding. *Psychological Review*, (94):115–147, 1987.
- [4] D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent Dirichlet Allocation. In S. Becker T. Dietterich and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*. MIT Press, Cambridge, MA, 2002.
- [5] C. Boutilier and R.S. Zemel. Online queries for collaborative filtering. In *Ninth International Workshop on Artificial Intelligence and Statistics*, 2002.
- [6] M. Brand. Pattern discovery via entropy minimization. In *Proc. Artificial Intelligence and Statistics 7*, 1998.
- [7] M. Brand. Structure Learning in Conditional Probability Models via an Entropic Prior and Parameter Extinction. *Neural Computation*, 11(5):1155–1182, 1999.

- [8] Y. Cheng and G.M Church. Biclustering of Expression Data. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology (ISMB)*, 2000.
- [9] Y. Freund and D. Haussler. Unsupervised Learning of Distributions of Binary Vectors Using 2-Layer Networks.
- [10] Z. Ghahramani. Factorial learning and the EM algorithm. In G. Tesauro, D.S. Touretzky, and T.K. Leen, editors, *Advances in Neural Information Processing Systems 7*. MIT Press, Cambridge, MA, 1995.
- [11] Z. Ghahramani and G.E. Hinton. The EM algorithm for mixtures of factor analyzers. Technical Report CRG-TR-96-1, 1996.
- [12] Z. Ghahramani and M.I. Jordan. Factorial Hidden Markov Models. In David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8. MIT Press, Cambridge, MA, 1995.
- [13] B. Heisele, T. Poggio, and M. Pontil. Face Detection in Still Gray Images. A.I. Memo 1687, Massachusetts Institute of Technology, May 2000.
- [14] G. Hinton and R.S. Zemel. Autoencoders, minimum description length, and Helmholtz free energy. In G. Tesauro J. D. Cowan and J. Alspector, editors, *Advances in Neural Information Processing Systems 6*. Morgan Kaufmann Publishers, San Mateo, CA, 1994.
- [15] G.E. Hinton, Z. Ghahramani, and Y.W. Teh. Learning to parse images. In S.A. Solla, T.K. Leen, and K.R. Muller, editors, *Advances in Neural Information Processing Systems 12*. MIT Press, Cambridge, MA, 2000.
- [16] T. Hofmann. Probabilistic latent semantic analysis. In *Proc. of Uncertainty in Artificial Intelligence, UAI'99*, Stockholm, 1999.

- [17] T. Hofmann. Learning what people (don't) want. In *European Conference on Machine Learning*, 2001.
- [18] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive Mixtures of Local Experts. *Neural Computation*, 3(1):79–87, 1991.
- [19] N. Jojic and B.J. Frey. Learning flexible sprites in video layers. In *CVPR*, 2001.
- [20] D.D. Lee and H.S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, October 1999.
- [21] D.D. Lee and H.S. Seung. Algorithms for non-negative matrix factorization. In T.K. Leen, T. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*. MIT Press, Cambridge, MA, 2001.
- [22] S. Li, X. Hou, and H. Zhang. Learning Spatially Localized, Parts-Based Representation. In *CVPR*, 2001.
- [23] A.M. Martinez and R. Benavente. The AR face database. Technical Report 24, CVC, 1998.
- [24] B.W. Mel. Think positive to find parts. *Nature*, 401:759–760, October 1999.
- [25] B. Mirkin. *Mathematical Classification and Clustering*. Kluwer Academic Publishers, 1996.
- [26] 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.
- [27] D.A. Ross and R.S. Zemel. Multiple Cause Vector Quantization. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*. MIT Press, Cambridge, MA, 2003.

- [28] B. Schölkopf and A. J. Smola. *Learning With Kernels*. MIT Press, Cambridge, MA, 2002.
- [29] B. Schölkopf, A. J. Smola, R. Williamson, and P. Bartlett. New support vector algorithms. *Neural Computation*, 12:1207–1245, 2000.
- [30] M. Weber, W. Einhäuser, M. Welling, and P. Perona. Viewpoint-Invariant Learning and Detection of Human Heads. In *IEEE International Conference on Automatic Face and Gesture Recognition*, 2000.
- [31] C. Williams and N. Adams. DTs: Dynamic trees. In M.J. Kearns, S.A. Solla, and D.A. Cohn, editors, *Advances in Neural Information Processing Systems 11*. MIT Press, Cambridge, MA, 1999.
- [32] R.S. Zemel. *A Minimum Description Length Framework for Unsupervised Learning*. PhD thesis, Dept. of Computer Science, University of Toronto, Toronto, Canada, 1993.