University of Glasgow | School of Computing Science

Generative Probabilistic Models for Image Retrieval

Vassilios Stathopoulos

Submitted for the Degree of Doctor of Philosophy
University of Glasgow

February 2012

Contents

Mathematical Notation

This section introduces the mathematical notation and a list of functions and symbols frequently used throughout this Thesis.

Vectors are denoted with lower case bold Roman letters, such as $\boldsymbol{x}$, and are assumed to be column vectors. A superscript T denotes vector or matrix transposition, e.g. $\boldsymbol{x}^T$ is a row vector. $(x_1, \ldots, x_D)$ denotes a row vector with D elements and the corresponding column vector is denoted as $\boldsymbol{x} = (x_1, \ldots, x_D)^T$. We will use $\boldsymbol{x} \in \mathbb{R}^D$ to denote the D dimensional vector $\boldsymbol{x}$ whose elements are real numbers. Matrices are denoted with upper case bold Roman letters such as $\boldsymbol{A}$ with $A_{i,j}$ denoting the element at the i^{th} row and j^{th} column. If we have N values $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ of a D dimensional vector $\boldsymbol{x} = (x_1, \ldots, x_D)^T$ we combine them in a $D \times N$ matrix $\boldsymbol{X}$ with the n^{th} column being the n^{th} value, $\boldsymbol{x}_n$, of the vector $\boldsymbol{x}$. Often $\boldsymbol{x} \in \boldsymbol{X}$ will be used to denote the vector which takes values from the column vectors of the matrix $\boldsymbol{X}$ and $|\boldsymbol{X}|$ to denote the number of columns in the data matrix $\boldsymbol{X}$. $\mathrm{tr}(\boldsymbol{A})$ denotes the trace of the matrix $\boldsymbol{A}$, i.e. the sum of its diagonal elements $A_{i,j}, \quad i = j$ and $\det(\boldsymbol{A})$ denotes the matrix determinant.

Probability densities and distributions will be both denoted using $p(\cdot)$ and the meaning will be clear from the context. Conditional probability will be denoted using $p(\cdot|\cdot)$, e.g. $p(x|z)$ denotes the distribution of the values of the variable x conditioned on the value of the variable z. The expectation of a function $f(x, y)$ with respect to a random variable x will be denoted with $\mathbb{E}_x[f(x, y)]$. For matrices with columns being observations of a random vector such as $\boldsymbol{X}$ we will use $p(\boldsymbol{X})$ to denote the joint probability $p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N)$.

$\sum_{\{i:0<i<N+1\}} x_i$ denotes the summation of variables x_i where the index i takes values in the set appearing below the summation operator, i.e. $x_1 + \cdots + x_N$. The symbol $\propto_q$ will be used to denote rank equivalence for a query q. For example $f(x, q) \propto_q \log f(x, q)$ signifies that the log of the function $f(x, q)$ produces the same rank of objects x as the function $f(x, q)$. $\nabla_{\boldsymbol{x}} f(\boldsymbol{x})$ denotes the gradient of the function $f(\boldsymbol{x})$ with respect to $\boldsymbol{x}$, that is the vector

with values

$$\left(\frac{\partial f(\boldsymbol{x})}{\partial x_1}, \ldots, \frac{\partial f(\boldsymbol{x})}{\partial x_D}\right)^T.$$

$\arg\max_x f(x)$ denotes the value of the variable x such that $f(x)$ attains its maximum value.

$\Gamma(\cdot)$ denotes the Gamma function while $\Gamma_D(\cdot)$ denotes the D-variate Gamma function defined as $\Gamma_D(n) = \pi^{D(D-1)/4} \prod_{j=1}^{D} \Gamma\left[n + (1-j)/2\right]$. $B(\cdot)$ denotes the Beta function defined as

$$B(x, y) = \frac{\Gamma(x)\Gamma(y)}{\Gamma(x+y)}.$$

For a random vector $\boldsymbol{x} \in \mathbb{N}^D$, $\mathcal{M}(\boldsymbol{x}|n, \boldsymbol{\theta})$ denotes the Multinomial distribution with probability mass function

$$\mathcal{M}(\boldsymbol{x}|n, \boldsymbol{\theta}) = \frac{(\sum_{i=1}^{D} x_i)!}{\prod_{i=1}^{D} x_i!} \prod_{i=1}^{D} \theta_i^{x_i},$$

where $n \in \mathbb{N}$ such that $x_1 + \cdots + x_D = n$ and $0 < \theta_i < 1$ and $\theta_1 + \cdots + \theta_D = 1$. $\mathcal{D}(\boldsymbol{\theta}|\boldsymbol{\alpha})$ denotes the Dirichlet distribution with probability density function

$$\mathcal{D}(\boldsymbol{\theta}|\boldsymbol{\alpha}) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{i=1}^{D} \theta_i^{a_i - 1},$$

where $\alpha_i \in \mathbb{R}^+$.

For a random variable $y \in \mathbb{R}^+$, $\mathcal{G}(y|\alpha, \beta)$ denotes the Gamma distribution with probability density function

$$\mathcal{G}(y|\alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} y^{\alpha-1} \exp(-\beta y),$$

where $\alpha, \beta \in \mathbb{R}^+$.

For a random vector $\boldsymbol{x} \in \mathbb{R}^D$, $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the D-variate Gaussian distribution with probability density function

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-D/2} \det(\boldsymbol{\Sigma})^{-1/2} \exp\left((\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu})\right),$$

where $\boldsymbol{\mu} \in \mathbb{R}^D$ and $\boldsymbol{\Sigma}$ a positive definite $D \times D$ matrix. $\mathrm{St}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, v)$ denotes the D-variate Student-t distribution with probability density function

$$\mathrm{St}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}, v) = \frac{\Gamma(v/2 + D/2)\det(\boldsymbol{\Sigma})^{1/2}}{\Gamma(v/2)(v\pi)^{D/2}} \left(1 + \frac{1}{v}(\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}(\boldsymbol{x} - \boldsymbol{\mu})\right)^{-v/2-D/2},$$

where $v \in \mathbb{R}^+$.

Finally, for a positive definite $D \times D$ matrix $\boldsymbol{\Sigma}$, $\mathcal{IW}(\boldsymbol{\Sigma}|\boldsymbol{W}, v)$ denotes the Inverse Wishart distribution with probability density function

$$\mathcal{IW}(\boldsymbol{\Sigma}|\boldsymbol{W}, v) = \frac{\det(\boldsymbol{W})^{\frac{v}{2}}\det(\boldsymbol{\Sigma})^{\frac{-(v+D+1)}{2}}}{2^{vD/2}\Gamma_D(v/2)} \exp\left(-\frac{1}{2}\mathrm{tr}(\boldsymbol{W}\boldsymbol{\Sigma}^{-1})\right),$$

where $v \in \{r : r > D - 1, r \in \mathbb{R}^+\}$.

Chapter 1

Introduction

Searching for information is a recurring problem that almost everyone has faced at some point. Being in a library looking for a book, searching through newspapers and magazines for an old article or searching through emails for an old conversation with a colleague are some examples of the searching activity. These are some of the many situations where someone; the "user"; has some vague idea of the information he is looking for; an "information need"; and is searching through a large number of documents, emails or articles; "information items"; to find the most "relevant" item for his purpose. Information Retrieval is the academic field which studies all aspects of the information retrieval process: from the structure, analysis and indexing of information items, to the modelling of the users' information need and the presentation of relevant information. The goal is to design effective information systems to assist users in searching large collections of information items.

Although every aspect of the information retrieval process is important, all rely on a fundamental task that every information retrieval system should fulfil, namely *ad-hoc retrieval*. Ad-hoc retrieval is the task where the information retrieval system is given a representation of the user's information need, a user "query", and searches through a static collection of information items and returns the most relevant ones. The system does not have any prior information on which documents might be relevant and the only information available is the user query and the static collection. The system must also return information items ordered such that items assumed to be more relevant to the user query are ordered first.

Image retrieval studies the problems arising in information retrieval when users are searching for images in large image archives. Image retrieval systems have many applications such as medical image retrieval, image and video copyright infringement, surveillance from CCTV cameras and personal image collections, to name a few. Moreover, image retrieval systems are often used as subcomponents for larger multimedia retrieval systems, e.g. video retrieval systems. Images pose several questions about the retrieval process such as how the user's information need should be represented and how images in the collection should be processed, indexed and retrieved. In this thesis we study the problem of *ad-hoc image retrieval* and we look into two particular ways of representing the user's information need, namely *query by example* and *semantic retrieval*. In the former case we assume that users express their information need with images and they look for "similar" images in a static collection. In the latter case we assume that users express their information

needs using semantic features such as semantic categories and keywords, e.g. "tiger", "sunset" etc. and search for images depicting the concepts or objects described by those semantic features.

1.1 Motivation

Models of the Information Retrieval Process

As in any scientific field, information retrieval also relies on models. Models allow researchers to express their assumptions and hypotheses about the observations in a concise mathematical way and provide guidance on designing experiments to validate these assumptions. Models in ad-hoc information retrieval are used to make hypothesis and express the assumptions of how users searching through a collection of information items are categorising them into relevant and non relevant and how information items are ranked in order of relevance. Competing hypotheses of different models can be evaluated in a controlled environment and the most successful can be used for designing information retrieval systems.

The information retrieval process is inherently uncertain since the users' perception of relevance is not a deterministic process. Information items can be relevant with respect to a user's information need at different degrees and this might change over time. A natural way to handle such uncertainties is to consider retrieval as a stochastic process and make probabilistic statements about the relevance of information items. Probabilistic models have played

a fundamental role in allowing researchers to gain insights in the retrieval process and develop practical and effective ranking and retrieval systems (Robertson & Zaragoza 2009).

One family of probabilistic models which has been very successful in information retrieval are *Language Models* (Ponte & Croft 1998, Hiemstra 2001). More detailed discussion on language models is given later in this thesis however for now we briefly mention the main model assumptions and characteristics. Language models assume that each information item is generated by a stochastic process which specifies the probability of specific *information bearing features* to be present in the information item. The reason for using this definition will become apparent later but for making the definition more clear we mention that in the case of text retrieval the information item is a document and *information bearing features* are words or some canonical representation derived from words. In this model, a user expresses his information need with the same *information bearing features* (words) and the relevance of information items (documents) is modelled by the likelihood that the query is generated by the same generative process as the information item.

In contrast to what the name suggests, the assumptions made by language models are very generic and can in fact be applied to other types of information items. Notice that on purpose we gave a definition of the main assumptions which is independent of the specific type of information. For image retrieval information items and queries can be images. The information bearing features are then features describing the appearance and characteris-

tics of images. These topics will be discussed in more detail in the Chapter 2. Several researchers have considered probabilistic models and especially language models for image retrieval, see Westerveld et al. (2003) and Vasconcelos & Lippman (2000) for examples. Moreover, audio and music retrieval can be similarly expressed in the same probabilistic framework (Turnbull et al. 2008). In this thesis the main focus is the study of probabilistic models based on the language modelling framework for image retrieval.

Probabilistic models as well as other types of models in general, are subject to *unknown* model parameters. This is not a drawback of the probabilistic modelling framework but a modelling decision in order to allow flexibility and interpretation. For example in language models for text retrieval the same stochastic process is assumed for all documents. However, the model parameters allow different instantiations of the generative process where each of the model parameters corresponds to the probability of a word in each document.

Uncertainty in model parameters

Unknown model parameters have to be estimated using the available information. For probabilistic models, parameter estimation is a well studied problem in the statistics literature. The parameters are set such that the likelihood of the observed data under the chosen model is maximised. This is referred to as "model fitting" or Maximum Likelihood estimation. For ad-hoc retrieval the only available information is the user query and the static collection and in the language modelling framework there is a model for each

document in the collection. Hence, the document model parameters can be estimated by maximising the likelihood of the observed document's word frequencies.

However, here lies another source of uncertainty since there will be many different parameter values that will fit the observed data with different likelihood. Using only parameter values which result in the maximum likelihood implicitly assumes that there is a "true" model. However, models are only used as simplifying surrogates of more complicated processes. Despite the philosophical aspects of the argument, in information retrieval this has practical implications. For example, what is the probability of a word that has zero frequency in a document? Maximum likelihood suggests that is also zero which implies that documents are not relevant with respect to user queries containing at least one word with zero document frequency. In information retrieval this problem has been handled by *smoothing* maximum likelihood estimates by introducing small "pseudo-frequencies" for terms with zero document frequency.

The theoretical consequence of smoothing is that *a-priori* it is assumed that all terms have equal non-zero probability in a document. The observed document frequencies are then used to update these prior beliefs resulting in a *posterior distribution*. The relationships between smoothing and prior distributions over model parameters have been a well studied topic in information retrieval, see Zhai & Lafferty (2001) and MacKay & Bauman Peto (1994) for examples. Smoothing however is a less studied topic for the models used for image retrieval. For example, Vasconcelos & Lippman (2000),

Westerveld et al. (2003) Fergus et al. (2003) and Carneiro et al. (2007) use maximum likelihood estimates for their probabilistic models. In this thesis we study the problem of specifying prior distributions for generative probabilistic models used in image retrieval and propose to go a step further from simply using smoothed parameter estimates.

To handle the uncertainty in model parameters we have to also allow for probabilistic statements about the parameters to be made. That is, we need to obtain a posterior distribution over parameters that quantifies the conditional probability of parameter values given the observed data. Then when making predictions, such as calculating the probability of a query being generated from the same process as an information item in the collection, we want to *marginalise* this uncertainty by taking the average of all possible model parameters weighted by their posterior probability. Zaragoza et al. (2003) show that this can be easily done for language models for text retrieval since the *marginal predictive density* has a simple analytic form. In this thesis we show how the results in Zaragoza et al. (2003) can be directly applied for image retrieval models based on the *Bag of Terms* (Zhu et al. 2002) representation.

For the image retrieval models of Vasconcelos & Lippman (2000) and Westerveld et al. (2003) obtaining the posterior directly is not trivial. The reason is that the generative probabilistic model for images is defined as a mixture of Gaussian distributions (McLachlan & Peel 2000). The posterior for mixtures does not have a closed form and thus one has to resort to numerical estimation or approximations. In this thesis we study these problems

in more detail. In particular we study how we can approximate the posterior distribution and how we can obtain samples using Markov Chain Monte Carlo (Robert & Casella 2005) which we can use to numerically estimate the marginal predictive density. Then we apply these methods and develop image retrieval systems for the query by example and semantic retrieval paradigms which we evaluate in a controlled environment using real data.

1.2 Thesis Statement

In this work we study the problem of ad-hoc image retrieval using generative probabilistic models. We consider two main paradigms for image retrieval, query by example and query by semantic features. We propose to handle the uncertainty in model parameters by ranking images using the *marginal predictive density* instead of the query likelihood and demonstrate the methodology using Bag of Terms (Zhu et al. 2002) and Gaussian mixture models (Vasconcelos & Lippman 2000, Westerveld et al. 2003). For semantic retrieval, we also propose the use of the *marginal predictive densities* of Gaussian mixture models for developing Support Vector Machine and Naive Bayes classifiers. Our focus is to develop algorithms that have the same computational complexity as previous approaches. In order to achieve this we consider suitable approximations when needed.

Our main claim is that handling uncertainty in this way leads to improved retrieval performance while the resulting algorithms remain within the same order of complexity as maximum likelihood approaches. We evaluate our

claims using real image data and compare with previous approaches. The methodology advocated here is very general and can be applied to any generative probabilistic model provided that an approximation or samples from the posterior of model parameters are available.

The research outcomes of this thesis can be summarised as follows:

- We provide an extensive study of Bayesian inference methods for mixtures of Gaussian distributions and show how these methods can be applied for image retrieval systems.
- We evaluate the ranking function of Zaragoza et al. (2003) for image retrieval with the Bag of Terms representation.
- We generalise the models of Westerveld et al. (2003) and Vasconcelos & Lippman (2000) for ranking images using the predictive densities. Our proposed approach has the same order of complexity whilst achieving significantly better performance.
- A query modelling approach using Gaussian mixture models and ranking using a kernel function between the predictive densities of query and collection images is also studied. This methodology also allows to construct Support Vector Machine classifiers for semantic image classification.
- We develop a hierarchical Variational EM algorithm for efficiently estimating the approximate posteriors of Gaussian mixtures modelling images in the same semantic category.

- We generalise the methodology of Carneiro et al. (2007) for Naive Bayes classification using the predictive class conditional densities.

1.3 Thesis Outline

The remainder of the thesis is organised as follows:

- In Chapter 2 we discuss probabilistic models of information retrieval with emphasis on language models. We also discuss image retrieval models for query by example and semantic features and close the chapter with a discussion on the evaluation of information retrieval systems.
- In Chapter 3 we discuss in detail Gaussian mixture models and parameter estimation methods using MCMC and variational approximation. We empirically evaluate novel MCMC samplers in terms of efficiency and computational complexity. The discussion and results from this chapter have also been presented in Stathopoulos & Girolami (2010) and Stathopoulos & Girolami (2011).
- In Chapter 4 we develop and evaluate image retrieval systems for query by example using Bag of Terms and Gaussian mixture models. Part of this chapter has been presented in Stathopoulos & Jose (2011).
- In Chapter 5 we develop and evaluate semantic image retrieval systems with Naive Bayes and Support Vector Machines classifiers. Part of this chapter has been presented in Stathopoulos & Jose (2009)

- Finally Chapter 6 concludes this thesis and discuss future work.

Chapter 2

Background and Related Work

2.1 Information Retrieval

2.1.1 Vector Space Models

One of the first models for information retrieval was based on the notion of similarity (Luhn 1957). Luhn (1957) proposed that documents and queries should be represented in the same way and then a similarity function for this representation should be used to rank documents in decreasing order of similarity with respect to a query. The simplest instantiation of this model represents documents and queries as vectors $\boldsymbol{d} = (d_1, \ldots, d_T)^T$ and $\boldsymbol{q} = (q_1, \ldots, q_T)^T$ respectively, where each of the d_i and q_i elements are associated with the i^{th} index term. Index terms can be words, keywords or some canonical form of words obtained by stemming (Manning et al. 2008, Chap. 2) for example. Using this representation a simple similarity measure is the

vector inner product

$$\text{sim}(\boldsymbol{q}, \boldsymbol{d}) = \sum_{i=1}^{T} q_i \times d_i.$$

The elements of the vectors can be binary indicating the presence or absence of a term in a document or query, in which case the inner product results in the number of common terms. Alternatively, the vector elements can be some real valued weight indicating the importance of each term for documents and queries and thus the inner product will favour more significant terms. In either case, the inner product similarity function tends to favour larger documents since they will simply have more terms entering the sum. The solution is to normalise the similarity function with the document and query lengths respectively to give

$$\text{sim}(\boldsymbol{q}, \boldsymbol{d}) = \frac{\sum_{i=1}^{T} q_i \times d_i}{\sqrt{\sum_{i=1}^{T} q_i^2} \times \sqrt{\sum_{i=1}^{T} d_i^2}},$$

in which case the similarity function returns the cosine of the angle between the document and query vectors (van Rijsbergen 1979).

The main question is then how to obtain weights for documents and queries indicating the importance of terms. This question has been the core one addressed by the Information Retrieval community for several decades and there have been many different weighting algorithms. A complete account of this research is out of the scope of this thesis and thus we refer the reader to Salton & Buckley (1988) and references therein for a more thorough review. We describe however one of the most fundamental weighting algorithms for the vector space model that has also been used for modern

image retrieval systems.

The TF-IDF weighting algorithm is based on the observation that terms which appear more frequently in a document will be more significant in determining a document's relevance with respect to a query with these terms (Salton & Buckley 1988). Its name stems from its two components, namely Term Frequency and Inverse Document Frequency. On the other hand, terms appearing in almost all documents in the collection will not be discriminating between different documents (Sparck-Jones 1972). The TF-IDF weight balances these two effects by combining the frequency of a term in a document (TF) with the frequency of the term in the collection (IDF). The weights for documents and queries using the TF-IDF weighing scheme are

$$d_i = n_{d,i} \log \frac{N}{df_i}, \quad q_i = n_{q,i} \log \frac{N}{df_i},$$

where $n_{d,i}$ denotes the frequency of the i^{th} term in document d, df_i denotes the number of documents in the collection with at least one occurrence of the i^{th} term and N is the number of documents in the collection.

Despite the simplicity of the vector space model, it has been proven very successful in practice and particularly suitable for large scale collections. The inner product similarity function depends only on the terms matching between the query and the document and thus it naturally exploits the sparse nature of the representation. However it does not provide any theoretical guidance on how to estimate the term weights.

2.1.2 Probabilistic Retrieval Models

Instead of relying on ad-hoc methods and empirical observations in order to develop weighting functions for the vector space model, probabilistic retrieval models treat the information retrieval problem as a special case of a classification problem. For a particular query, the collection is divided in two parts, relevant and non relevant documents. Ranking documents by their probability of being relevant guarantees that the ranking will be optimal. Estimating this probability is not trivial however and several different models have been proposed.

The probabilistic models of Robertson & Jones (1976) and several versions of it, as well as the BM25 of Robertson & Walker (1994) and its predecessor, the 2-Poisson model (Harter 1975) can be seen as instantiations of a more general probabilistic framework that models relevance with a binary variable (Robertson & Zaragoza 2009). We will not describe these models in full detail as they are not used further in this thesis. However we will briefly discuss the general framework with its main assumptions in order to highlight the difficulties in generalising it for different types of media.

Encoding relevance of documents with respect to a particular query q by a binary variable $r_q \in \{0, 1\}$ with 1 denoting relevance and 0 non relevance, the quantity of interest is

$$p(r_q = 1|\boldsymbol{d}) \propto_q \frac{p(r_q = 1|\boldsymbol{d})}{p(r_q = 0|\boldsymbol{d})} = \frac{p(\boldsymbol{d}|r_q = 1)p(r_q = 1)}{p(\boldsymbol{d}|r_q = 0)p(r_q = 0)} \propto_q \frac{p(\boldsymbol{d}|r_q = 1)}{p(\boldsymbol{d}|r_q = 0)}, \quad (2.1)$$

where $\propto_q$ is used to denote that two functions are rank equivalent with re-

spect to the query q, i.e. they produce the same ranking of documents. Assuming that documents are represented by *unordered sets of features* $\boldsymbol{d} = \{d_1, \ldots, d_T\}$ and that individual features d_i are conditionally independent such that $p(\boldsymbol{d}|r_q) = \prod_{i=1}^{T} p(d_i|r_q)$, Equation (2.1) can be written as:

$$\frac{p(\boldsymbol{d}|r_q = 1)}{p(\boldsymbol{d}|r_q = 0)} \approx \prod_{i=1}^{T} \frac{p(d_i|r_q = 1)}{p(d_i|r_q = 0)} \propto_q \sum_{i=1}^{T} \log \frac{p(d_i|r_q = 1)}{p(d_i|r_q = 0)}. \tag{2.2}$$

From Equation 2.2, we can see that the ranking function is the sum of the log odds of document features being relevant. So far the only assumption restricting the nature of the document representation is that documents are unordered sets of features with individual features being conditionally independent. In the context of image retrieval, one can imagine that the d_i features are multidimensional vectors describing local patches of an image. The independence assumption might be more unrealistic in the domain of images, however it is commonly used in the Computer Vision literature to obtain tractable models for object recognition.

Once the document representation is known then one has to specify the distributional assumptions for the conditionals $p(d_i|r_q = 1)$ and $p(d_i|r_q = 0)$. The main problem that arises from this formulation is that the conditionals will be modelled by some parametric distribution or density. For example Bernouli for the model of Robertson & Jones (1976) and a mixture of two Poisson distributions for the BM25 (Robertson & Walker 1994) and 2-Poisson models (Harter 1975). For estimating the parameters however we cannot employ the standard maximum likelihood estimators since we are in the peculiar situation where we have no observed data. The only observed information is

that provided by the query which for ad-hoc retrieval is usually very limited for maximum likelihood estimation purposes. The common approach is to make approximations and further simplifying but reasonable assumptions. We will discuss the model of Robertson & Jones (1976) in order to clarify this point however for the BM25 model we refer the reader to Robertson & Walker (1994) and Robertson & Zaragoza (2009).

Binary Independence Model

Robertson & Jones (1976) assume that document features d_i are binary, modelling the absence or presence of the i^{th} index term in a document. They then employ a Bernoulli model for the conditionals $p(d_i|r_q = 1)$ and $p(d_i|r_q = 0)$ with parameters $\pi_{i,r_q=1}$ and $\pi_{i,r_q=0}$ respectively. The first assumption that is common in the Robertson-SparkJones and BM25 models is that terms which are not in the query contribute a constant term in Equation (2.2) and thus they can be neglected. This assumption is usually made in order to obtain a ranking function that can be efficiently implemented with an inverted index data structure. By separating terms not in the query, rearranging and removing any remaining constant terms, Equation (2.2) can be written

$$\begin{aligned}\sum_{i=1}^{T} \log \frac{p(d_i|r_q = 1)}{p(d_i|r_q = 0)} &\approx \sum_{\{i:d_i>0 \wedge q_i>0\}} \log \frac{p(d_i|r_q = 1)p(0|r_q = 0)}{p(d_i|r_q = 0)p(0|r_q = 1)} \\ &= \sum_{\{i:d_i>0 \wedge q_i>0\}} \log \frac{\pi_{i,r_q=1}(1 - \pi_{i,r_q=0})}{\pi_{i,r_q=0}(1 - \pi_{i,r_q=1})},\end{aligned}$$

where in the last equation we have introduced the parameters for the Bernoulli distributions.

Given a random sample of fully judged documents with respect to the query then the standard maximum likelihood estimate for the two Bernoulli parameters can be used. To avoid problems related to unobserved terms a smoothed estimate is used where the pseudo frequency term of 0.5 is introduced. The final Robertson-Sparck Jones function is then

$$\sum_{\{i:d_i>0\wedge q_i>0\}} \log \frac{(a_i+0.5)(N-R-n_i+a_i+0.5)}{(n_i-a_i+0.5)(R-a_i+0.5)}, \tag{2.3}$$

where N is the size of the judged document sample, n_i is the number of judged documents with term i, R is the number of relevant judged documents and a_i is the number of judged relevant documents with term i. When there are relevant judgments these quantities can be estimated directly. However for ad-hoc retrieval where there is no relevance information and further assumptions have to be made. For large document collections most of the documents will not be relevant with respect to the query. One limiting case is to consider that all documents will not be relevant thus N will be the size of the collection and n_i the number of documents in the collection with term i. Similarly, very few documents will be relevant and the limiting case is when R and a_i are both 0. The resulting formula is then

$$\sum_{\{i:d_i>0\wedge q_i>0\}} \log \frac{N-n_i+0.5}{n_i+0.5}, \tag{2.4}$$

which resembles the IDF term used in the TF-IDF weighting scheme.

2.1.3 Unigram Language Models

Language models are an alternative probabilistic framework for information retrieval. Instead of treating the retrieval process as a classification problem a generative model for documents is employed. The documents can then be ranked using the *query likelihood*, that is the probability that the query has been generated by the same language model as the document (Ponte & Croft 1998).

Each document in the collection is assumed to be a random sample from a stochastic process. The modelling assumptions of the generative process are specified by a generative probabilistic model $p(\boldsymbol{d}|\boldsymbol{\theta}_d)$ governed by unknown parameters $\boldsymbol{\theta}_d$. Notice that parameters are indexed by the document in order to make explicit the fact that each document is generated by a different process. The simplest form of this model is to assume that queries and documents are *unordered sets of features* represented by vectors, $\boldsymbol{q} = (q_1, \ldots, q_T)^T$ and $\boldsymbol{d} = (d_1 \ldots, d_T)^T$ respectively, and that features are conditionally independent such that $p(\boldsymbol{d}|\boldsymbol{\theta}_d) = \prod_{i=1}^{T} p(d_i|\boldsymbol{\theta}_d)$. This is also referred as a *unigram* language model. The unknown parameters can then be estimated by maximising the likelihood (ML) using the observed document term frequencies

$$\hat{\boldsymbol{\theta}}_d^{(ML)} = \arg\max_{\boldsymbol{\theta}_d} p(\boldsymbol{d}|\boldsymbol{\theta}_d)$$

or by the maximising the posterior (MAP) assuming a suitable prior distribution $p(\boldsymbol{\theta}_d)$.

$$\hat{\boldsymbol{\theta}}_d^{(MAP)} = \arg\max_{\boldsymbol{\theta}_d} p(\boldsymbol{d}|\boldsymbol{\theta}_d)p(\boldsymbol{\theta}_d) \tag{2.5}$$

Many of the smoothing procedures developed for the unigram language models in the information retrieval literature are equivalent to a MAP estimate, e.g. Zaragoza et al. (2003), Zhai & Lafferty (2001).

Notice that for the unigram model ordering is not important since document features are considered conditionally independent. Moreover, this formulation is very general and it does not make any particular assumptions about the nature of the document's features or the actual generative process for the documents apart from the assumption that features are conditionally independent. For example, in the image retrieval context the document features can be multidimensional vectors and a suitable generative process for such data can be specified.

Once the unknown parameters have been estimated, documents can be ranked with respect to a user query using the following general ranking function

$$p(\boldsymbol{d}|\boldsymbol{q}) = \frac{p(\boldsymbol{q}|\boldsymbol{d})p(\boldsymbol{d})}{p(\boldsymbol{q})} \propto_q p(\boldsymbol{q}|\boldsymbol{d})p(\boldsymbol{d})$$

where $p(\boldsymbol{d})$ is a document prior. In this thesis we assume documents have equal prior probabilities but Miller et al. (1999) considers several alternatives. The quantity $p(\boldsymbol{q}|\boldsymbol{d})$ is the probability of the query conditioned on the the observed document and its actual form is:

$$p(\boldsymbol{q}|\boldsymbol{d}) = \int p(\boldsymbol{q}|\boldsymbol{\theta}_d)p(\boldsymbol{\theta}_d|\boldsymbol{d})d\boldsymbol{\theta}_d, \tag{2.6}$$

where the model parameters are marginalised using the posterior $p(\boldsymbol{\theta}_d|\boldsymbol{d})$. Equation (2.6) is also known as the *predictive distribution*.

The integral is commonly approximated by using a smoothed estimate of the model parameters such that $p(\boldsymbol{q}|\boldsymbol{d}) \approx p(\boldsymbol{q}|\hat{\boldsymbol{\theta}}_d)$. This is the approach followed by many instantiations of the language modelling framework for document retrieval such as (Ponte & Croft 1998, Hiemstra 2001, Berger & Lafferty 1999, Miller et al. 1999). In some cases (Zaragoza et al. 2003, Zhai & Lafferty 2001) the smoothing estimation procedure is equivalent to a MAP estimate and thus the approximation can be justified by the asymptotic properties of the posterior.

When there is a large number of data, the likelihood term in Equation (2.5) will overwhelm the prior and thus the posterior will be sharply peaked with all its mass concentrated in a small region around the maximum. Therefore a MAP estimate is a reasonable approximation of the integral above. However, in document retrieval the number of parameters is usually so large (equal to the terms in the vocabulary) that any reasonable sized document will not contain enough observations. The problem is further amplified when document features are high dimensional vectors as in the image retrieval context.

Zaragoza et al. (2003) show that for unigram models of term frequencies both the posterior and the predictive distribution can be obtained analytically and therefore there is no need for the MAP estimate approximation. Moreover, the predictive density is analytic for many popular smoothing methods which can be expressed using the Dirichlet distribution. Since most of this thesis is concerned with the application of this framework in the context of image retrieval we describe these models in more detail.

Language models for ad-hoc text retrieval

For text documents, the document features d_i correspond to the frequency of the i^{th} index term in the document (Ponte & Croft 1998). For this representation the most common model in the literature is that of a Multinomial[1] with distribution:

$$\mathcal{M}(\boldsymbol{d}|\boldsymbol{\theta}_d) = \frac{(\sum_{i=1}^{T} d_i)!}{\prod_{i=1}^{T} d_i!} \prod_{i=1}^{T} \theta_{d,i}^{d_i},$$

where each $\theta_{d,i}$ parameter models the probability of the i^{th} index term in the document. The Maximum Likelihood estimate is $\hat{\theta}_{d,i}^{(ML)} = d_i / \sum_{i'} d_{i'}$ (Bishop 2006, Chap. 2). The parametric form of the prior distribution is usually chosen such that it simplifies the derivations although it should be flexible enough in order not to introduce bias. For the Multinomial models such a prior is the Dirichlet with distribution:

$$\mathcal{D}(\boldsymbol{\theta}_d|\boldsymbol{a}) = \frac{\Gamma(\sum_{i=1}^{T} a_i)}{\prod_{i=1}^{T} \Gamma(a_i)} \prod_{i=1}^{T} \theta_{d,i}^{a_{i-1}},$$

where a_i are the prior hyper-parameters (Gelman et al. 2003) and $\Gamma(\cdot)$ is the Gamma function. Different smoothing methods can be recovered by setting these parameters appropriately. For example setting $a_i = 2$ the Laplace smoothing is recovered while setting a_i to the average frequency of the i^{th} index term in the collection the MAP estimate is equivalent to a Bayes-smoothed estimate. See Zaragoza et al. (2003) and Zhai & Lafferty (2001) for a discussion and empirical evaluation of different smoothing methods. Finally, the general form of the MAP estimate is $\hat{\theta}_{d,i}^{(MAP)} =$

[1]The Multinomial model also relaxes the conditional independence assumption with a weaker assumption, that of exchangeability.

$(d_i + a_i - 1)/\sum_{i'=1}^{T}(d_{i'} + a_{i'} - 1)$ and the analytical form of the posterior is also a Dirichelt $\mathcal{D}(\boldsymbol{\theta}_d|\boldsymbol{d} + \boldsymbol{a})$ (Gelman et al. 2003).

Using a MAP approximation for the predictive distribution the resulting ranking function is:

$$\begin{aligned} \log p(\boldsymbol{q}|\hat{\boldsymbol{\theta}}_d^{(MAP)}) \quad &\propto_q \quad \sum_{\{i:q_i>0 \wedge d_i>0\}} q_i \log\left(\frac{d_i}{a_i - 1} + 1\right) \\ &- \quad \log\left(\sum_{i'} d_{i'} + a_{i'} - 1\right) \sum_{\{i:q_i>0\}} q_i. \end{aligned} \tag{2.7}$$

Notice that the ranking function depends only on terms common in the document and the query and thus it can be efficiently implemented. Moreover, the term $(\sum_{i'} d_{i'} + a_{i'} - 1)$ is a document dependent constant and can be calculated during indexing. In contrast to the Binary Independence model discussed in the previous section there is no need to make additional approximations in order to arrive in this convenient form.

Finally, using the explicit form of the posterior the predictive distribution can be analytically evaluated (Zaragoza et al. 2003) to give:

$$p(\boldsymbol{q}|\boldsymbol{d}) = \frac{(\sum_i q_i)!}{\prod_i q_i!} \frac{\Gamma\left(\sum_i d_i + a_i\right)}{\Gamma\left(\sum_i q_i + d_i + a_i\right)} \prod_i \frac{\Gamma\left(q_i + d_i + a_i\right)}{\Gamma\left(d_i + a_i\right)}, \tag{2.8}$$

which can again be simplified to a form that is convenient for implementation

using an inverted index

$$\begin{aligned}\log p(\boldsymbol{q}|\boldsymbol{d}) \quad &\propto_q \quad \sum_{\{i:q_i>0 \wedge d_i>0\}} \sum_{g=1}^{q_i} \log\left(1+\frac{d_i}{a_i+g-1}\right) \\ &- \quad \sum_{j=1}^{\sum_{i'} q_{i'}} \log\left(\sum_{i'} d_{i'}+a_{i'}+j-1\right). \end{aligned} \tag{2.9}$$

Equations (2.7) and (2.9) will be re-introduced in Chapter 4 where they will be applied for image retrieval using the Bag-of-Terms representation.

2.2 Image Retrieval

2.2.1 Similarity Models

Image retrieval models have also been based on the notion of similarity or distance and they have many similarities with the vector space model. One of the main difficulties however is how to represent user queries. For text documents the user query can be naturally expressed with keywords and thus a document and query representation with features derived directly from words or terms seems natural. For images, such representation is not possible unless the image collection is manually curated and indexed. One way to resolve this problem is to allow the user to express queries using images. This is often referred to as *query by example* in the content based image retrieval literature.

Images are represented by vectors in a high dimensional feature space and a similarity or distance function is used for ranking and retrieval. The

elements of an image vector correspond to features describing visual and structural characteristics of an image such as colour, texture and shape. A large number of research papers has focussed on developing and evaluating different feature types and similarity functions and an extensive review can be found in Smeulders et al. (2000) and Deselaers et al. (2008). Moreover the Multimedia Content Descriptor Interface, MPEG-7, (Salembier & Sikora 2002) is an ISO standard for describing the content of multimedia objects such as videos and images. It specifies several different image descriptors and corresponding similarity or distance functions. Although the standard does not specify how content based image retrieval should be carried out, it serves as the basic building block for multimedia retrieval systems (Lux 2009).

The main assumption of these approaches is that the similarity function and the image features correspond to the user's perception of similarity for a particular task. For example Oliva & Torralba (2001) develop a feature representation which projects image vectors close together in a high dimensional space if they share membership in semantic categories. However, many authors have indicated that no single type of feature is suitable for all retrieval tasks. The solution adopted in practice is to either allow the user to specify the type of features and similarity function (Lux 2009) or implement algorithms to automatically combine information from different types of features (Donald & Smeaton 2005).

Although the are clear similarities between early content based image retrieval systems and the vector space model, their implementation is radically different. The reason for this is that unlike documents which have a natural

sparse representation, vectors representing images are composed by dense continuous features. Data structures such as K-D trees and R* trees are not appropriate for very high dimensional data such as those often encountered in image retrieval as their performance deteriorates for data with more than 10 dimensions Weber et al. (1998). Indexing methods relying on hashing (Andoni & Indyk 2006) are more efficient for such data although they are approximate in nature.

2.2.2 Bag of Terms Models

Bag of terms models are considered the state of the art approach for large scale image retrieval systems (Zhu et al. 2002). The key to their success is that they rely on a sparse representation which can be efficiently implemented with an inverted index data structure and is similar to the bag of words representation used in text information retrieval. Moreover, bag of terms models allow any of the classical information retrieval algorithms discussed in the previous section to be applied directly on content based image retrieval. For example Philbin et al. (2007) and Sivic & Zisserman (2003) use the TF-IDF weighting algorithm and the Euclidean distance to rank image vectors.

The indexing process for bag of terms models involves four distinct stages which are briefly discussed in the next paragraphs. Namely, region detection, visual description, code block generation and quantisation.

Region detection

In the region detection stage images are subdivided into local regions where each region depicts parts of the main concepts or objects present. A robust detection algorithm should satisfy two properties, accuracy and repeatability, i.e. given two images depicting the same scene under different viewing conditions the algorithm should detect the same regions at the corresponding image locations (Mikolajczyk, Tuytelaars, Schmid, Zisserman, Matas, Schaffalitzky, Kadir & Van Gool 2005). This implies that the detection algorithm should be invariant to affine transformations and to scale changes, where by scale changes we refer to the appearance of objects at different distances from the camera.

The two most frequently used approaches for region detection in bag of terms models are image segmentation algorithms such as Normalised Cuts (Shi & Malik 1997) and affine region detectors (Mikolajczyk, Tuytelaars, Schmid, Zisserman, Matas, Schaffalitzky, Kadir & Van Gool 2005). Affine region detectors are mostly based on edge and corner detection and they extract regions around points in the image where there is a significant change of the signal in orthogonal directions (Harris & Stephens 1988). Therefore, although these detectors can be used to very accurately model shape and texture, they cannot detect regions of homogenous appearance. On the other hand, segmentation algorithms segment images into homogenous regions but they are very sensitive to scale, illumination and perspective changes.

A simple yet very effective alternative which has been shown to perform better than affine region detectors (Nowak et al. 2006, Tuytelaars 2010) is to

segment images using a regular grid. In that way both object parts and homogenous regions can be captured. This is also the approach that we follow in Chapters 4 and 5. In order to achieve scale invariance several grids of different resolution can be used. For example, in Koniusz & Mikolajczyk (2010) images at different scales are used in order to introduce scale invariance using a segmentation algorithm.

Visual description

Once local regions have been extracted the next stage of the indexing process is to compute a feature descriptor from the pixel intensities within the image regions. For this purpose several types of features can be used such as colour or edge histograms or even the raw pixel values (Nowak et al. 2006). The Scale Invariant Feature Transform (SIFT) descriptor extracts a histogram with 128 bins of edge orientations and sifts the histogram such that the most dominant orientation is at the first bin in order to achieve rotation invariance (Lowe 2004). The standard SIFT descriptors cannot model the appearance of homogeneous regions since they do not capture colour information and rely on the presence of strong edges. Several extensions to the SIFT descriptors have been proposed and evaluated in (van de Sande et al. 2010).

Obdrzálek & Matas (2003) propose to exploit the compression properties of the Discrete Cosine Transform (DCT) in order to obtain a low dimensional representation of image regions. The DCT is even more effective when applied on image regions in YUV colour space since higher compression rates can be obtained for the chrominance channels, (Hamilton 1992). DCT

coefficients have also been used for probabilistic image retrieval models by Vasconcelos & Lippman (2000) and Westerveld et al. (2003) as well for image classification in Carneiro et al. (2007). We also chose to use this feature representation for our experiments in Chapter 4 and 5.

Code block generation

In the code block generation stage the feature descriptors obtained from all images in the collection are clustered using the K-means algorithm or other variants into a large number of clusters. The cluster means are assigned a unique identifier and are treated similarly to terms in the bag of words model for information retrieval. The set of all cluster means with their unique identifiers is usually referred as the *visual vocabulary* (Zhu et al. 2002). The code block generation stage is the most computationally demanding task of the indexing process. The reason is that a large number of feature descriptors, usually in the order of millions, need to be clustered into a large number of clusters, usually in the order of thousands. At each iteration of the K-Means algorithm each feature descriptor has to be compared with all current cluster means. As discussed previously, data structures such K-D trees cannot be used in this scenario since their performance can be even worst than a linear scan (Weber et al. 1998).

A solution that has been proposed by several authors (Nister & Stewenius 2006, Philbin et al. 2007) is to apply K-Means in a hierarchical fashion. That is, the K-means algorithm is applied initially to all feature descriptors using a small number of clusters, e.g. 2. Then a separate K-Means process can

be applied to the items of each of the clusters obtained at the previous step. The process continues recursively until the desired number of clusters has been reached or when a cluster contains a predefined number of items. This not only allows for an efficient parallel implementation but also significantly simplifies the quantisation stage. The problem with this approach however is that quantisation errors accumulate at each step and thus the clustering obtained is an approximation.

Quantisation

Images are quantised by finding for each image feature descriptor its closest, in the Euclidean sense, cluster mean from the visual vocabulary. Then the feature descriptor can be replaced by the unique identifier corresponding to that cluster mean. Thus images have the same representation as text documents with the unique visual term identifiers playing the role of words. If the hierarchical K-Means approach has been used for the code block generation then the quantisation can be efficiently implemented using a tree data structure (Philbin et al. 2007). For a tree with a branching factor K and depth L only $K \times L$ distance calculations are required for finding the closest cluster mean for an image feature descriptor. In comparison, a simple linear scan would require K^L distance calculations. This is important not only because it allows the algorithms to scale to larger collections but also because the quantisation process has to be applied for the user's query image.

2.2.3 Probabilistic Models

Bag of terms models do not provide any guidance on designing weighting algorithms or similarity functions. The weighting algorithms developed by the information retrieval community make assumptions which are tailored for text documents and thus they are not necessarily valid for bag of terms models. For example the TF-IDF weighting algorithm assumes that terms in the collection follow a power law distribution (Manning et al. 2008, Chap. 5). However, the K-means algorithm clusters feature descriptors such that clusters have a uniform distribution over the entire collection. Moreover, the quantisation errors introduced by the K-means clustering can significantly degrade the discriminative power of feature descriptors (Boiman et al. 2008) and thus negatively affect retrieval performance.

Vasconcelos & Lippman (2000) and Westerveld et al. (2003) have proposed a generative probabilistic model for image retrieval that has many similarities with the language modelling framework. However, instead of relying on a quantisation of the feature space in order to generate discrete image features they directly model the density of feature descriptors in each image. Following the notation introduced in Section 2.1.3 the generative process for images is modelled using a finite Gaussian mixture model:

$$p(\boldsymbol{I}|\boldsymbol{\theta}_I) = \prod_{\boldsymbol{x}\in\boldsymbol{I}} \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{2.10}$$

where an image is assumed to be an *unordered set of feature vectors* represented by a matrix $\boldsymbol{I}$ with columns the vectors $\boldsymbol{x}_1, \ldots \boldsymbol{x}_M$ where each vector

is a feature descriptor extracted from a local image region with $\boldsymbol{x} \in \mathbb{R}^D$ and D the dimensionality of the feature descriptor. $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. The parameters $\boldsymbol{\theta}_I = \{\pi_k, \boldsymbol{\mu_k}, \boldsymbol{\Sigma}_k : k \in \{1, \ldots, K\}\}$ are estimated by maximising the likelihood (2.10) using the Expectation Maximisation (EM) algorithm which we discuss in detail in Chapter 3. Given a new query image $\boldsymbol{Q}$, images in the collection can then be ranked using the query likelihood $p(\boldsymbol{Q}|\hat{\boldsymbol{\theta}}_I^{(ML)})$, i.e. the probability that the query image has been generated by the same process as the image $\boldsymbol{I}$.

This approach does not involve smoothing the model parameters and thus it is subject to over fitting. In information retrieval smoothing parameter estimates by encoding collection statistics using a prior distribution has played a core role in the development and theoretical understanding of the retrieval process. This can be easily achieved for this model by adopting a conjugate prior and use a variation of the EM algorithm in order to obtain a smoothed MAP estimate.

In Section 2.1.3 we have seen that ML and MAP estimates give us only an approximation of the ranking function since the quantity that we want to estimate is the *predictive density* (Zaragoza et al. 2003):

$$p(\boldsymbol{Q}|\boldsymbol{I}) = \int p(\boldsymbol{Q}|\boldsymbol{\theta}) p(\boldsymbol{\theta}|\boldsymbol{I}) d\boldsymbol{\theta}.$$

From the discussion in Section 2.1.3 we can see that the predictive density can be easily obtained for bag of terms models where the image descriptors have been discretised. In Chapter 4 we give a practical implementation

and evaluate this approach. However, for the generative model in Equation 2.10 the posterior $p(\boldsymbol{\theta}|\boldsymbol{I})$ is not analytically tractable and thus neither is the predictive density. In Chapter 3 we study two approaches for this problem. One is to obtain samples from the posterior using Markov Chain Monte Carlo (MCMC) and use them in order to numerically estimate the integral for the predictive density. The second approach is to approximate the posterior in such a way that analytical results are possible. Finally, in Chapter 4 we discuss how these methods can be applied for image retrieval and present a practical implementation which we evaluate on a real image collection.

2.2.4 Semantic Image Retrieval Models

The practical applicability of the query by example paradigm for ad-hoc interactive retrieval is limited due to the fact that users often find it difficult to express their information need using image queries. A more natural way for users is to use keywords or terms which convey directly the semantics of their information need. However, this requires that the image collection is indexed using semantic features such as keywords. In order to automate the process of indexing, classification models have been applied in order to categorise images into semantic categories which can then be used for *semantic retrieval*.

The representation induced by the bag of terms models have inspired researchers to view the image classification problem as a cross language retrieval problem. That is the user express his query using one language, keywords, and the system retrieves images which are described by visual terms. Duygulu et al. (2002) have applied algorithms from machine transla-

tion in order to estimate the joint probability of keywords and visual terms. Jeon et al. (2003) has applied the cross lingual language model of Lavrenko et al. (2002) while Lavrenko et al. (2003) has generalised this methodology to directly model high dimensional continuous feature descriptors without resorting to quantisation.

Classification models such as those applied for document categorisation (Sebastiani 2002) have also been applied for image classification. Csurka et al. (2004) have experimented with Naive Bayes and Support Vector Machines (SVM) classifiers using the bag of terms representation. In order to further motivate our research we have to discuss the Naive Bayes classifier in more detail.

Naive Bayes classifiers

The Naive Bayes classifier is similar to the probabilistic retrieval model discussed in Section 2.1.2 in that class membership is encoded with a binary variable. Using a random variable $w_c \in \{0, 1\}$ to denote membership in class w_c, the Naive Bayes classifier is obtained by the application of Bayes rule for inverting the probability of class membership conditioned on an observed query image Q as

$$p(w_c|Q) = \frac{p(Q|w_c)p(w_c)}{\sum_{w_i} p(Q|w_c)p(w_c)}, \tag{2.11}$$

where $p(w_c)$ is the prior distribution of classes which for now we assume to be uniform. The main focus is then sifted to the class conditional density $p(Q|w_c)$ for which we must specify a generative probabilistic model for images of the class w_c. This is the reason why the Naive Bayes classifier is also

referred as a *generative classifier* (Ng & Jordan 2001). Again the exact form of the class conditional density can be written as

$$p(Q|w_c) = \int p(Q|\boldsymbol{\theta})p(\boldsymbol{\theta}|w_c)d\boldsymbol{\theta},$$

where the posterior $p(\boldsymbol{\theta}|w_c)$ is in fact conditioned on images in a training collection with class w_c. Similar to the discussion in Section 2.1.3 approximations of the integral using point estimates, such as ML and MAP, can be justified by the asymptotic properties of the posterior. Thus the conditional density can be rewritten as $p(Q|\hat{\boldsymbol{\theta}}_{w_c}^{(MAP)})$ where $\hat{\boldsymbol{\theta}}_{w_c}^{(MAP)}$ is a MAP estimate of the generative model parameters obtained by maximisation of the joint likelihood using a training set of images with class w_c.

Csurka et al. (2004) use the bag of terms representation and the class conditional densities are modelled by a Multinomial distribution. To obtain a smoothed estimate they use a Laplace smoothing which corresponds to a Dirichlet prior with parameters set to 2. Fergus et al. (2003) use a generative model for jointly modelling the appearance, location and scale of SIFT local feature descriptors. Their model also includes latent parameters thus the EM algorithm is employed to obtain a ML estimate. Carneiro et al. (2007) use a Gaussian mixture model for the class conditional densities and apply the hierarchical EM algorithm of Vasconcelos & Lippman (1998) to efficiently estimate the model parameters.

Again we can see that ML and MAP estimates provide only an approximation. In Chapter 5 we show how we can estimate the *predictive class conditional* densities for bag of terms and Gaussian mixture models using

the methods discussed in Chapter 3. Moreover, we generalise the hierarchical estimation algorithm of Vasconcelos & Lippman (1998) in order to allow for an efficient parallel implementation.

2.3 Other approaches

In this section we briefly discuss other approaches for the image retrieval problem which are not directly related with the methodology presented in this thesis but have been influential in the image retrieval literature.

Yavlinsky et al. (2005) also approaches image retrieval as a classification problem and indexes images by calculating the conditional probabilities of keywords given a query image $p(w_c|Q)$. By using Bayes' theorem the conditional probability is again written as in Eq. (2.11) resulting in a Naive Bayes classifier. For the class conditional probabilities $p(Q|w_c)$ a non-parametric kernel density estimator is employed using a vector representation of images. That is, an image is represented as a high dimensional feature vector by either concatenating region feature descriptors or by extracting a global feature descriptor, see Yavlinsky et al. (2005) for more details. The kernel density estimator takes the form

$$p(Q|w_c) = \frac{1}{|\mathcal{T}_c|} \sum_{\boldsymbol{I}_c \in \mathcal{T}_c} \mathcal{N}(\boldsymbol{Q}|\boldsymbol{I}_c, \sigma \boldsymbol{I}),$$

where $\mathcal{T}_c$ is the set of all images in the training collection depicting the keyword w_c, $\boldsymbol{I}$ is the identity matrix and σ is a scale parameter. For large

training data the kernel density estimator becomes very expensive to compute and we can see the approach proposed in Chapter 5 as a simplification where the class conditional densities are estimated using a semi-parametric Gaussian mixture model.

The methodology proposed by Magalhães & Rüger (2006) and Magalhães & Rüger (2010) is more similar to the bag of terms models where instead of a K-means quantisation a "soft clustering" of feature descriptors is performed using Gaussian mixture models to create the visual vocabulary. For parameter estimation a regularisation term based on the Minimum Description Length (MD) criterion is introduced in the EM algorithm that penalises models with many parameters, i.e. many mixture components. In contrast to the Naive Bayes approach followed by Csurka et al. (2004) and Yavlinsky et al. (2005) a discriminative logistic regression classifier is employed to directly model the probability of class membership as

$$p(w_c|\boldsymbol{Q}) = \frac{1}{1+e^{-z}}, \quad z = \sum_{i=1}^{|\boldsymbol{Q}|}\sum_{k=1}^{K} \beta_{c,i,k} q_k(\boldsymbol{x}_i) + \beta_{c,0},$$

where $\beta_{c,i,k}$ are class specific regression parameters for keyword w_c estimated by maximum likelihood approaches using the Limited BFGS method and $q_k(\boldsymbol{x}_i)$ is the probability of the k^{th} term in the visual vocabulary given a corresponding feature descriptor $\boldsymbol{x}_i$ of the query image. This approach also diverges from the classical bag of terms methodology since a distinct visual vocabulary is estimated for different image regions and type of feature while it also assumes that the image representation results in the same number of feature descriptors for each image. In Chapter 5 we also use a discrim-

inative classifier based on Support Vector Machines but instead of using a visual vocabulary we utilise a kernel function between densities of feature descriptors in images. In that way we allow images to have different number of feature descriptors and thus our methodology is also applicable to SIFT type descriptors.

An alternative approach which has been proposed by many authors (Blei & Jordan 2003, Jeon et al. 2003, Lavrenko et al. 2003, Feng et al. 2004) is to treat the semantic retrieval problem as a cross lingual retrieval problem and directly model the joint probability of keywords and image features. Jeon et al. (2003) use a bag of terms representation and model the joint distribution between keywords and visual terms by introducing latent variables z indexing images in the training set and marginalising to get

$$P(\boldsymbol{w}_q, \mathbf{Q}) = \sum_{z=1}^{N} P(z) \prod_{w \in \boldsymbol{w}_q} P(w|z) \prod_{x \in \boldsymbol{Q}} P(x|z), \tag{2.12}$$

where N is the number of images in the collection and $P(z)$ is the probability of the image indexed by z in the collection, which is assumed to be uniform. $P(w|z)$ and $P(x|z)$ are the probabilities of the word w and the visual term x to appear in the image indexed by z respectively. In Jeon et al. (2003) both terms are assumed to be independent Bernoulli distributions and a smoothed estimate is obtained by interpolating between a maximum likelihood and a collection statistic, e.g. $P(x|z) = (1-b_z)\frac{count(x,z)}{|z|} + b_z\frac{count(x,T)}{|T|}$. $|T|$ is the total number visual terms in the collection while $|z|$ is the number of visual terms in the image indexed by z. $count(x, z)$ and $count(x, T)$ denote the frequency of visual term x in the image indexed by z and in the collection respectively.

The interpolation parameters b_z control the degree of smoothing and in Jeon et al. (2003) are estimated using cross validation.

Lavrenko et al. (2003) proposes the Continuous Relevance Model (CRM) where the feature space is not quantised into visual terms, rather continuous variables are used to represent the visual features of images. The difference of the CRM with the model introduced in Jeon et al. (2003) lies in the estimation of the $P(\mathbf{x}|z)$ by a kernel density estimator of the form $P(\mathbf{x}|z) = \frac{1}{|\boldsymbol{I}_z|} \sum_{\boldsymbol{x}_z \in \boldsymbol{I}_z} \mathcal{N}(\mathbf{x}|\mathbf{x}_z, \boldsymbol{\Sigma})$, and that $P(w|z)$ is assumed to be a Multinomial distribution where a MAP estimate is obtained using a Dirichlet prior. Finally, Feng et al. (2004) uses the same kernel density estimator for visual features but replaces the Multinomial distribution with a Bernoulli where MAP estimates are obtained by using a Beta prior.

In the models of Jeon et al. (2003) ,Lavrenko et al. (2003) and Feng et al. (2004) the latent variables z are simply indexing images in the collection and thus any estimates are based on averages over the whole image collection. An other popular alternative in probabilistic retrieval models is to assume a latent factor, or topic, representation and model each document or image as a mixture of these factors (Hofmann 1999). An important latent space model, developed for discrete data collections like documents and applied in Information Retrieval, is Latent Dirichlet Allocation (LDA), introduced in Blei et al. (2003). Similar to Probabilistic Latent Semantic Analysis (PLSA) (Hofmann 1999), LDA considers images, or documents, as mixtures of topics where topics are distributions over the visual term (or words) vocabulary. However, instead of introducing a latent variable for each observation it treats

the mixture probabilities of each image as a K-dimensional latent variable drawn from a parametric distribution whose parameters have to be estimated. In this way, the number of parameters that have to be estimated is reduced thus avoiding the over-fitting problems associated with PLSA.

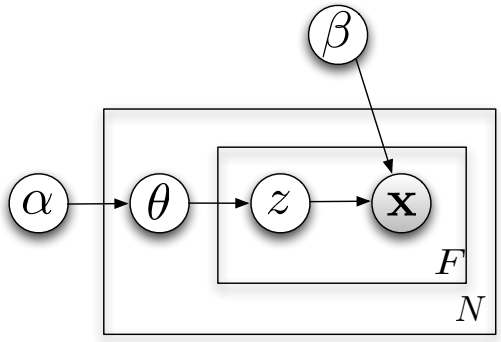

Figure 2.1: The graphical model corresponding to Latent Dirichlet Allocation. Shaded nodes correspond to observed variables and plates represent dimension

The graphical model of LDA is shown in Figure 2.1 and the joint distribution factorizes as:

$$p(\theta, \mathbf{z}, \mathbf{I}|\alpha, \beta) = p(\theta|\alpha) \prod_{f=1}^{F} p(\mathbf{z}_f|\theta) p(\mathbf{x}_f|\mathbf{z}_f, \beta), \tag{2.13}$$

where $p(\theta|\alpha)$ is a K-dimensional Dirichlet distribution with parameters α, $p(\mathbf{z}_f|\theta)$ is a K-dimensional Multinomial distribution and $p(\mathbf{x}_f|\mathbf{z}_f, \beta)$ is a Multinomial distribution with parameters β and conditioned on $\mathbf{z}_n$ which gives

$$p(\mathbf{x}_f|\mathbf{z}_f, \beta) = \prod_{k=1}^{K} Mult(\mathbf{x}_f|\beta)^{\delta(\mathbf{z}_f, 1)} \tag{2.14}$$

$\delta(\mathbf{z}_f, 1)$ an indicator function that returns 1 if $z_{f,k} = 1$ and 0 otherwise. β,

is a $K \times F$ matrix whose elements $\beta(k, f) = p(\mathbf{x}_f|z_k = 1)$ are the probabilities of visual terms under each topic, and α, is a K-dimensional vector parameterising the Dirichelt distribution. α and β are considered as collection dependent parameters which will be estimated by a training set. Given particular values for α and β the probability of an image, or document, with $\{\mathbf{x}_1, \ldots, \mathbf{x}_F\}$ visual terms is obtained by marginalising the parameters $\mathbf{z}$ and θ which gives

$$p(\mathbf{I}|\alpha, \beta) = p(\mathbf{x}_1, \ldots, \mathbf{x}_F|\alpha, \beta) = \int p(\theta|\alpha) \left(\prod_{f=1}^{F} \sum_{\mathbf{z}_f} p(\mathbf{z}_f|\theta) p(\mathbf{x}_f|\mathbf{z}_f, \beta) \right) d\theta. \tag{2.15}$$

In Blei & Jordan (2003) the LDA model has been extended to handle high dimensional continuous features $\boldsymbol{x}_f$ by exchanging the Multinomial distribution in Eq. (2.14) with multivariate Gaussian distributions with a topic specific mean and covariance parameters. The methodology presented in Chapters 4 and 5 although it has many similarities with the approach of Blei & Jordan (2003) is radically different in many respects. In Chapter 4 we do not assume a latent topic (factor) representation. Rather we estimate the feature density for each image using a Gaussian mixture model. In other words, in LDA there are K multivariate Gaussian mixture components for the whole image collection, while in Chapter 4 we have a K' component mixture model for each image. In Chapter 5 we employ Gaussian mixture models to model the density of features for images of a particular class and thus our approach can be seen as a latent topic representation for the images of that particular class. However, the estimation procedure is radically different. While in the

LDA model we need to estimate the parameters using the visual descriptors from all images, in Chapter 5 we derive a hierarchical Variational EM algorithm which only utilises the means and covariance matrices of the Gaussian mixture models estimated for each image individually.

2.4 Evaluation Methodology

For evaluating the performance of information retrieval systems in a controlled environment a *test collection* is needed. The test collection comprises a set of documents, a set of topics, which are expressed by queries, and a set of relevant judgements. In this thesis we use the Corel5K collection which is described in detail in Chapter 4. This collection has been used in several papers for evaluating content based query by example retrieval systems (Westerveld et al. 2003) and semantic retrieval (Jeon et al. 2003, Carneiro et al. 2007).

Given a query, the retrieval system returns a ranking of documents in the collection and then the relevance judgements are used to evaluate performance measures that quantify the system's performance. Table 2.1 shows the four possible states of a document given the output of the retrieval system and relevance judgments. For example a document in the ranking list returned by the retrieval system that has also been judged as relevant is a True Positive. Similarly a document in the ranking list judged as non relevant is a False Positive.

The *recall* of a retrieval system measures the percentage of relevant doc-

System / Judgments	Relevant	Non Relevant
Relevant	True Positive (TP)	False Positive (FP)
Non Relevant	False Negative (FN)	True Negative (TN)

Table 2.1: The four possible states of a document given the relevant judgments and the output of a retrieval system.

uments retrieved while *precision* measures the percentage of relevant documents in the retrieved set i.e.

$$recall = \frac{TP}{TP + FN}, \quad precision = \frac{TP}{TP + FP}.$$

A system that returns a ranking of the entire collection has $recall = 1$ and in general as *recall* increases *precision* decreases. The two measures can be combined using their harmonic mean which is usually referred as the F measure.

$$F = \frac{2 \times precision \times recall}{precision + recall} = \frac{2 \times TP}{TP + FP + TP + FN}.$$

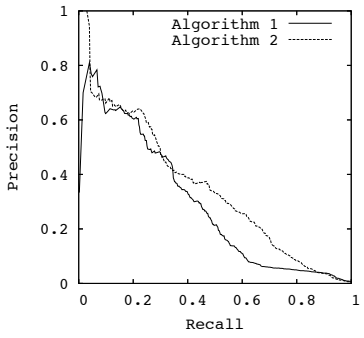

Figure 2.2: Example of a precision recall graph comparing two ranking algorithms.

Precision and recall are set measures evaluating the overall system performance and they do not take into account the order in which relevant documents appear in the ranking list. Two systems can have the same precision and recall but the system that ranks relevant documents higher is preferred. To take into account the order of relevant documents precision, can be calculated at several cutoff points of the ranking list. For example, P@10 is the precision calculated using the first 10 documents in the ranking list. Moreover, precision can be expressed as a function of recall, e.g. P(0.1) is the precision after ten percent of the relevant documents have been retrieved. The interpolated precision recall graph (Manning et al. 2008, Chap. 8) plots precision as a function of recall and can be used to assess the overall performance of the ranking algorithm. An example of a precision recall graph is shown in Figure 2.2.

Average Precision (AP) is a single measure that takes into account precision, recall and the ordering of documents in the ranking list by averaging precision in the recall interval $[0, 1]$. That is:

$$AP = \int_0^1 P(r)dr.$$

Finally the APs for every query in the test collection are averaged to result in the Mean Average Precision (MAP) which quantifies the overall performance of the ranking algorithm for the test collection.

Chapter 3

Mixture Models and Bayesian Inference

Mixture models, and in particular Gaussian mixture models, play a core role in the development of this thesis as they are used for modelling densities of information items, such as images, or groups of multimedia objects in a hierarchical manner. In this chapter we discuss mixture models in detail and study efficient algorithms for performing inference over their parameters. The focus of this chapter is on Bayesian inference where instead of point estimates we seek posterior distributions which can be used to obtain regularised estimates of the predictive densities where the uncertainty around the parameters has been marginalised.

A standard approach in density estimation using mixture models is maximum likelihood estimation through the Expectation Maximisation (EM) algorithm. Fully Bayesian approaches, although have been previously suc-

cessfully applied they are often considered inefficient and computationally expensive for practical applications like image retrieval. Recent developments though in the field of computational statistics, and in particular Markov Chain Monte Carlo (MCMC) methods (Roberts & Stramer 2003, Girolami & Calderhead 2011), have identified a new class of algorithms that promise significant improvements in efficiency and computational savings. In this chapter we develop the methodology necessary for applying this new class of MCMC algorithms on mixtures of Gaussian distributions and study their efficiency in terms of Effective Sample Size (ESS), i.e. the number of independent samples obtained by a Markov Chain, and computational complexity.

Despite the computational savings and improved sampling efficiency compered to more traditional MCMC algorithms, there are still several practical difficulties in applying MCMC algorithms for information retrieval systems. The most important ones being the requirement to run multiple chains for monitoring convergence and the need to store several posterior samples which have to be used to numerically integrate the query likelihood. Thus in the last section of this chapter we discuss approximate methods, namely Variational inference (Jordan et al. 1998), for estimating the posterior distribution of mixture model parameters. Variational inference provides analytical approximations of the posterior distributions while the computational complexity of the resulting algorithms is similar to the traditional EM algorithm. The standard variational methodology for mixture models is further developed in Chapter 5 where it is applied to hierarchies of Gaussian mixture models.

3.1 Gaussian Mixture Models

Mixture models are useful in describing a wide variety of random phenomena because of their inherent flexibility (Titterington et al. 1985). As such, they are used in many fields of science to model complex processes and systems. Examples of applications include clustering (McLachlan & Baek 2010), density estimation (Escobar & West 1994) and classification (Celeux 2006).

For observations $\boldsymbol{x} \in \mathbb{R}^D$, where D is the dimensionality of the feature space, a Gaussian mixture model with K components has the form:

$$p(\boldsymbol{x}|\boldsymbol{\Theta}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{3.1}$$

where $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the Gaussian density with mean parameter $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ which has the form:

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}\det(\boldsymbol{\Sigma})^{1/2}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})},$$

with $|\boldsymbol{A}|$ denoting the determinant of $\boldsymbol{A}$, π_k are the mixing coefficients such that $\sum_{k=1}^{K} \pi_k = 1$ and $\boldsymbol{\Theta} = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k : k \in \{1, \ldots, K\}\}$ is the set of all parameters.

For a dataset $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ of N observations the likelihood is:

$$p(\boldsymbol{X}|\boldsymbol{\Theta}) = \prod_{n=1}^{N} p(\boldsymbol{x}_n|\boldsymbol{\Theta}) = \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k). \tag{3.2}$$

An alternative and often useful formulation of mixture models can also

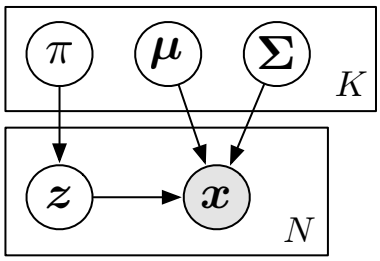

Figure 3.1: Plate diagram for a Gaussian mixture model. Shaded nodes represent observed variables and arrows conditional dependence. The plates represent multiple copies of the enclosed variables.

be obtained by introducing latent variables $\boldsymbol{Z} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n\}$, where $z_{i,k} \in \{0, 1\}$ and $\boldsymbol{z}_i \sim \mathcal{M}(1, \pi_1, \ldots, \pi_K)$ such that if $z_{i,k} = 1$ then the observation x_i is allocated to the kth mixture component. $\mathcal{M}(1, p_1, \ldots, p_K)$ is a Multinomial distribution for K mutually exclusive events with probabilities $p_1, \ldots, p_K$. The graphical model of this representation is shown in Figure 3.1 and the likelihood takes the form

$$p(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\Theta}) = \prod_{n=1}^{N} \prod_{k=1}^{K} \pi_k^{z_{n,k}} \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_{n,k}} \tag{3.3}$$

3.1.1 Maximum Likelihood Estimation

For a given set of observations, modelling its density using a Gaussian mixture model suffices in finding the parameters that maximise the log of the likelihood in Equation (3.2). The Expectation Maximasation (EM) algorithm, given in Algorithm 1, exploits the latent variable representation introduced in Equation (3.3) and successively optimises the parameters until convergence of the log likelihood.

Algorithm 1: EM-GMM

Inititialise $\boldsymbol{\Theta}$
repeat
  {E-Step}
  $\mathbb{E}\left[z_{n,k}\right] = \dfrac{\pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$
  {M-Step}
  $\pi_k = \dfrac{1}{N} \sum_{n=1}^{N} \mathbb{E}\left[z_{n,k}\right]$
  $\boldsymbol{\mu}_k = \dfrac{1}{\sum_{n=1}^{N} \mathbb{E}\left[z_{n,k}\right]} \sum_{n=1}^{N} \mathbb{E}\left[z_{n,k}\right] \boldsymbol{x}_n$
  $\boldsymbol{\Sigma}_k = \dfrac{1}{\sum_{n=1}^{N} \mathbb{E}\left[z_{n,k}\right]} \sum_{n=1}^{N} \mathbb{E}\left[z_{n,k}\right] \left(\boldsymbol{x}_n - \boldsymbol{\mu}_k\right) \left(\boldsymbol{x}_n - \boldsymbol{\mu}_k\right)^T$
until convergence

The EM algorithm is sensitive to initialisation conditions and it always converges to a local maximum of the log likelihood. To alleviate this problem, in practice the EM algorithm is repeated for several random initialisations of the parameters $\boldsymbol{\Theta}$ and the estimate with the largest log likelihood is kept.

A parameter that is often neglected in maximum likelihood estimation is the number of components K which is generally unknown a-priori. Unlike the local maxima problem, the log likelihood is monotonically increasing with the number of components and approaches infinity as the number of components approaches the number of observations, thus allowing for a single component for each observation. Practical solutions to the problem involve a penalisation term with respect to the number of parameters. An example of such penalisation is the Bayesian Information Criterion (BIC) defined in Equation (3.4). In the setting of mixture models, performing model selec-

tion involves obtaining maximum likelihood estimates for models of different complexity and selecting the model which maximises the BIC:

$$\text{BIC} = 2\log(p(\boldsymbol{X}|\boldsymbol{\Theta})) - (K(D^2 + D + 1))\log(N). \tag{3.4}$$

3.2 Bayesian Inference

In Bayesian inference one is not interested in obtaining parameter estimates that maximise the likelihood, $p(\boldsymbol{X}|\boldsymbol{\Theta})$, but instead obtain a posterior distribution over the parameters, $p(\boldsymbol{\Theta}|\boldsymbol{X})$, by combining prior beliefs and evidence from the observed data through the use of Bayes' theorem (Equation 3.5). The benefits of Bayesian inference is that one can naturally encode prior information about the possible solutions and also obtain regularised estimates of the predictive distribution for new observations $\boldsymbol{x}^*$ by marginalising the uncertainty about parameters, $p(\boldsymbol{x}^*|\boldsymbol{X}) = \int p(\boldsymbol{x}^*|\boldsymbol{\Theta})p(\boldsymbol{\Theta}|\boldsymbol{X})d\boldsymbol{\Theta}$. Moreover, the posterior distribution over the parameters can be used as an importance sampling distribution for calculating the model evidence $\int p(\boldsymbol{X}|\boldsymbol{\Theta})p(\boldsymbol{\Theta})d\boldsymbol{\Theta}$ required for estimating Bayes' factors for model selection. As the model evidence is not a function of the parameters and is also a regularised estimate of the data likelihood, it naturally penalises more complex models as opposed to likelihood-ratio tests.

$$p(\boldsymbol{\Theta}|\boldsymbol{X}) = \frac{\overbrace{p(\boldsymbol{X}|\boldsymbol{\Theta})}^{\text{Likelihood}}\overbrace{p(\boldsymbol{\Theta})}^{\text{Prior}}}{\underbrace{\int p(\boldsymbol{X}|\boldsymbol{\Theta})p(\boldsymbol{\Theta})d\boldsymbol{\Theta}}_{\text{Marginal likelihood}}}. \tag{3.5}$$

For most problems of interest however, the posterior $p(\boldsymbol{\Theta}|\boldsymbol{X})$ is not analytically tractable or it is not possible to directly sample from it. In the rest of this chapter we will consider approaches to obtain samples from, or approximate, the posterior distribution for mixtures of Gaussians. The first approach is Markov Chain Monte Carlo (MCMC), Section 3.4, where a Markov Chain is constructed whose stationary distribution is the distribution of interest, $p(\boldsymbol{\Theta}|\boldsymbol{X})$. After convergence to the stationary distribution samples from the chain can be used to obtain Monte Carlo estimates of the required integrals. The second approach, Section 3.7, is to consider an approximate parametric form for the target posterior and estimate its parameters by minimising the Kullback-Leibler (KL) divergence between the true posterior and the approximate.

It is worth noting here that in contrast to approximate Bayesian inference, MCMC algorithms are theoretically exact in the limit of infinite samples. In practice however, and especially when the posteriors are complex and multimodal, they tend to only approximate the posterior around a local mode. Despite that, all the algorithms considered in the next sections can be easily integrated in a population MCMC scheme that allows to better explore the parameter space.

3.3 Choice of Priors

In a Bayesian framework prior knowledge about a particular problem is encoded in the prior distribution $p(\boldsymbol{\Theta})$. When such *a-priori* knowledge is available, a prior restricting the parameter space and favouring particular solutions over others can be designed. For example, for a vector of variables $\boldsymbol{\beta} \in \mathbb{R}^D$ we can encode our preference to sparse solutions by imposing a Laplace prior (Williams 1995) for individual elements of the vector $\boldsymbol{\beta}$. Often however, there is no a-priori information about which solutions are preferred. Such lack of knowledge can be encoded by using *flat* priors, i.e priors that assign equal probability to all possible solutions, e.g the uniform distribution.

In many practical applications, another important factor on the choice of prior is conjugacy. That is, the parametric form of the prior is selected such that it is mathematically convenient to derive several expressions. For example when the likelihood is a Gaussian with an unknown mean and known variance, then a Gaussian prior over the mean parameter allow us to analytically calculate the posterior, which is also a Gaussian.

For Gaussian mixture models a common choice of conjugate priors is Dirichlet for the mixing coefficients π_k, Gaussian for the mean parameters $\boldsymbol{\mu}_k$ and inverse Wishart, or Wishart, for the covariance matrices $\boldsymbol{\Sigma}_k$, or precision matrices $\boldsymbol{\Sigma}_k^{-1}$ respectively. The joint prior as well as the forms of the individual priors are given in Equations (3.6 - 3.9) where $B(\cdot)$ is the multinomial beta function, $\Gamma_D(\cdot)$ is the multivariate Gamma function and $\mathrm{tr}(\boldsymbol{A})$ is the trace of $\boldsymbol{A}$. When conjugacy is not required or the model is re-parameterised, e.g. see Section 3.4.2, alternative priors can be assumed.

$$
\begin{aligned}
p(\boldsymbol{\Theta}) &= p(\boldsymbol{\pi}) \prod_{k=1}^{K} p(\boldsymbol{\mu}_k|\boldsymbol{\Sigma}_k) p(\boldsymbol{\Sigma}_k), && (3.6)\\
p(\boldsymbol{\pi}) &= \mathcal{D}(\boldsymbol{\pi}|\boldsymbol{a}_0), && (3.7)\\
p(\boldsymbol{\mu}_k|\boldsymbol{\Sigma}_k) &= \mathcal{N}(\boldsymbol{\mu}_k|\boldsymbol{m}_0, \beta^{-1}\boldsymbol{\Sigma}_k), && (3.8)\\
p(\boldsymbol{\Sigma}_k) &= \mathcal{IW}(\boldsymbol{\Sigma}_k|\boldsymbol{W}_0, v_0). && (3.9)\\
& && (3.10)
\end{aligned}
$$

The prior hyper-parameters $\boldsymbol{a}_0, \boldsymbol{m}_0, \boldsymbol{W}_0$ can be used to encode prior beliefs. For example, the Dirichlet prior ensures that $0 \leqslant \pi_k \leqslant 1$ and that $\sum_{k=1}^{K} \pi_k = 1$. Hyper-parameters $\boldsymbol{a}_0$ can be set to 1 corresponding to a uniform prior thus indicating a complete lack of knowledge *a-priori* about the parameters $\boldsymbol{\pi}$. Values closer to 0, on the other hand, favour sparse vectors $\boldsymbol{\pi}$ and therefore models with few mixture components are preferred. Conversely, values larger than 1 favour dense vectors $\boldsymbol{\pi}$ and thus preference is given to models where all mixture components have almost equal contribution.

The rest of the prior hyper-parameters can be chosen such that the resulting prior distribution is flat in the region where the likelihood is substantial and not much greater elsewhere (McLachlan & Peel 2000, Chap. 4). For example, the prior mean hyper-parameters $\boldsymbol{m}_0$ are usually set to the mean of the observations $\boldsymbol{X}$ since components centred far away from the observations' mean are unlikely *a-priori*. Similarly, β^{-1} parameter can be set to a large value, 5 to 10, and the positive definite matrix $\boldsymbol{W}_0$ can be set to the inverse of the covariance of the observations, $\frac{1}{N}\boldsymbol{X}^T\boldsymbol{X}$.

3.4 Markov Chain Monte Carlo

For obtaining samples from a complex posterior, MCMC algorithms construct a Markov Chain whose stationary distribution is the target posterior. The chain is constructed by defining a transition kernel $p(\boldsymbol{\theta}|\boldsymbol{\theta}^{t-1})$, where $t-1$ denotes the position of the chain at time $t-1$, and samples are drawn repeatedly such that

$$\boldsymbol{\theta}^t \sim p(\boldsymbol{\theta}|\boldsymbol{\theta}^{t-1}).$$

After convergence to the stationary distribution, the samples generated will be samples from the target posterior.

In order for a Markov Chain to have the target posterior $p(\boldsymbol{\theta})$ as its stationary distribution, the transition kernel has to leave the target posterior invariant and satisfy *detailed balance*:

$$p(\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{\theta}^*) = p(\boldsymbol{\theta}^*)p(\boldsymbol{\theta}^*|\boldsymbol{\theta}). \tag{3.11}$$

In the next sections we discuss specific MCMC algorithms and give the necessary expressions in order to draw samples from the posterior of Gaussian mixture models. We will not provide any proofs of *detailed balance* since it is out of the scope of this chapter, however the reader can refer to the seminal book of Robert & Casella (2005), Chapter 11 of Bishop (2006) and the references given in the next sections.

3.4.1 Metropolis-Hastings

For a random vector $\boldsymbol{\theta} \in \mathbb{R}^D$ with unormalised posterior density $\tilde{p}(\boldsymbol{\theta})$ the Metropolis-Hastings (MH) algorithm (Metropolis et al. 1953, Robert & Casella 2005) employs a proposal mechanism $q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{t-1})$ and proposed moves are accepted with probability

$$\min\left\{1, \frac{\tilde{p}(\boldsymbol{\theta}^*)q(\boldsymbol{\theta}^{t-1}|\boldsymbol{\theta}^*)}{\tilde{p}(\boldsymbol{\theta}^{t-1})q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{t-1})}\right\}.$$

In the context of Bayesian inference $\tilde{p}(\boldsymbol{\theta})$ corresponds to the unormalised posterior distribution of the model parameters:

$$\tilde{p}(\boldsymbol{\Theta}|\boldsymbol{X}) = p(\boldsymbol{X}|\boldsymbol{\Theta})p(\boldsymbol{\Theta}).$$

Tuning the Metropolis-Hastings algorithm involves selecting the right proposal mechanism. A common choice is to use a random walk Gaussian proposal of the form $q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{t-1}) = \mathcal{N}(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{t-1}, \boldsymbol{A})$. The general form of a Metropolis-Hastings sampler is given in Algorithm 2.

Selecting the covariance matrix however, is far from trivial in most cases since knowledge about the target density is required. Therefore a more simplified proposal mechanism is often considered where the covariance matrix is replaced with a diagonal matrix such as $\boldsymbol{A} = \epsilon\boldsymbol{I}$ where the value of the scale parameter ϵ has to be tuned in order to achieve fast convergence and good mixing. Small values of ϵ imply small transitions and result in high acceptance rates while the mixing of the Markov Chain is poor. Large values on the other hand, allow for large transitions but they result in most of the

proposals being rejected. Tuning the scale parameter becomes even more difficult in problems where the standard deviations of the marginal posteriors differ substantially, since different scales are required for each dimension, and this is exacerbated when correlations between different variables exist. Adaptive schemes for the Metropolis-Hastings algorithm have also been proposed (Haario et al. 2005) though they should be applied with care (Andrieu & Thoms 2008).

Algorithm 2: Metropolis-Hastings

1: Inititialise $\boldsymbol{\theta}^0$
2: **for** $t = 1$ to T **do**
3: $\quad \boldsymbol{\theta}^* \sim q(\boldsymbol{\theta}|\boldsymbol{\theta}^{t-1})$
4: $\quad r = \min\left\{1, \tilde{p}(\boldsymbol{\theta}^*)q(\boldsymbol{\theta}^{t-1}|\boldsymbol{\theta}^*)/\tilde{p}(\boldsymbol{\theta}^{t-1})q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{t-1})\right\}$
5: $\quad u \sim \mathcal{U}_{[0,1]}$
6: $\quad$ **if** $r > u$ **then**
7: $\quad\quad \boldsymbol{\theta}^t = \boldsymbol{\theta}^*$
8: $\quad$ **else**
9: $\quad\quad \boldsymbol{\theta}^t = \boldsymbol{\theta}^{t-1}$
10: $\quad$ **end if**
11: **end for**

In applying the MH algorithm for drawing samples from the posterior of mixture model parameters we are faced with two problems. Firstly, the parameters are not vectors in $\mathbb{R}^D$ and secondly some of the parameters, such as the mixing coefficients π_k and the precision, or covariance, matrices, are constrained, e.g. $0 \leqslant \pi_k \leqslant 1, \sum_{k=1}^{1} \pi_k = 1$ and $\boldsymbol{\Sigma}_k$ must be positive definite. The first problem can be solved by vectorising the precision, or covariance, matrices and concatenating all vectors, including the mean parameters and mixing coefficients, in a large vector in an augmented space. The second problem can be resolved by appropriately re-parameterising the model.

3.4.2 Re-parameterisation

For the mixing coefficients we can introduce parameters $\pi'_k \in \mathbb{R}_+$ such that $\pi_k = \frac{\pi'_k}{\sum_{l=1}^{K} \pi'_l}$ and use the relation of the Dirichlet with the Gamma distribution to impose independent Gamma priors on the new parameters, $p(\pi'_k) = \mathcal{G}(\pi'_k|a_k, 1)$. The interpretation of the Gamma hyper-parameters a_k remains the same as discussed in Section 3.3. The new parameters π'_k are still constrained to be positive so we can further re-paremeterise using $\pi'_k = e^{\gamma_k}, \gamma_k \in \mathbb{R}$ thus the original parameters are $\pi_k = \frac{e^{\gamma_k}}{\sum_{l=1}^{K} e^{\gamma_l}}$. Finally, the density of the independent Gamma priors have to be scaled appropriately by the Jacobian of e^{γ_k} thus $p(\gamma_k) = \mathcal{G}(e^{\gamma_k}|a_k, 1)e^{\gamma_k}$.

For the precision matrices we can re-parameterise by following Pinheiro & Bates (1996). We introduce new parameters $\boldsymbol{L}_k$ such that $\boldsymbol{\Sigma}_k^{-1} = \boldsymbol{L}_k\boldsymbol{L}_k^T$ where $\boldsymbol{L}_k$ is a lower triangular matrix. This re-parameterisation ensures that $\boldsymbol{\Sigma}_k^{-1}$ is positive definite. In order the decomposition to be unique the diagonal elements are constrained to be positive thus we can further re-parameterise such that

$$\boldsymbol{L}_k = \begin{bmatrix} e^{((\boldsymbol{B}_k)_{1,1})} & 0 & 0 & 0 \\ (\boldsymbol{B}_k)_{2,1} & \ddots & 0 & 0 \\ \vdots & \ddots & e^{((\boldsymbol{B}_k)_{d,d})} & 0 \\ (\boldsymbol{B}_k)_{D,1} & \dots & (\boldsymbol{B}_k)_{D,D-1} & e^{((\boldsymbol{B}_k)_{D,D})} \end{bmatrix}$$

For obtaining a prior over the new parameters we use the Bartlett decomposition which gives us the distributional assumptions of the elements of $\boldsymbol{L}_k$ (Kshirsagar 1959). The diagonal elements are independent Gamma variables while the off-diagonal elements are independent Gaussian. For the diagonal

term of $\boldsymbol{L}_k$ we also need to scale the Gamma density by the Jacobian of the transformation $(\boldsymbol{L}_k)_{d,d} = e^{((\boldsymbol{B}_k)_{d,d}}$.

The new set of parameters for the re-parameterised model are

$$\boldsymbol{\Theta}^r = \{\gamma_k, \boldsymbol{\mu}_k, \boldsymbol{B}_k : k \in \{1, \ldots, K\}\},$$

and the new joint prior as well as the individual prior densities are given in Equations (3.12 - 3.15) where we have dropped the dependence of the means to the covariance matrices.

$$\begin{aligned}
p(\boldsymbol{\Theta}^r) &= \prod_{k=1}^{K} p(\gamma_k)p(\boldsymbol{\mu}_k)p(\boldsymbol{B}_k), && (3.12)\\
p(\gamma_k) &= \mathcal{G}(e^{\gamma_k}|a_{0,k}, 1)e^{\gamma_k}, && (3.13)\\
p(\boldsymbol{\mu}_k) &= \mathcal{N}(\boldsymbol{\mu}_k|\boldsymbol{m}_0, \boldsymbol{W}_0), && (3.14)\\
p(\boldsymbol{B}_k) &= \prod_{d=1}^{D} \mathcal{G}(e^{(\boldsymbol{B}_k)_{d,d}}|\lambda, \psi)e^{(\boldsymbol{B}_k)_{d,d}} \prod_{d' \neq d} \mathcal{N}((\boldsymbol{B}_k)_{d',d}|0, \beta). && (3.15)
\end{aligned}$$

3.4.3 Gibbs Sampling

The Gibbs sampler (Geman & Geman 1984) can be seen as a special case of the MH algorithm where the proposal distribution at each step updates only one variable, or a set of variables, from their full conditional distributions and all samples are accepted with probability 1. Thus for a vector of random variables $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_D\}$ with posterior density $p(\boldsymbol{\theta})$ one step of the Gibbs sampler simulates a sample $\theta_d^t \sim p(\theta_d^*|\theta_1^t, \ldots \theta_{d-1}^t, \theta_{d+1}^{t-1}, \ldots, \theta_D^{t-1})$.

The Gibbs sampler requires that the full conditional distributions are explicit and easy to sample from. However, even when conjugate priors are

considered this is not always the case. For problems where the conditionals are not directly available a completion of the original distribution for which it is easier to derive the conditionals can be used (Robert & Casella 2005, Chap. 10). A completion of the original distribution is constructed by introducing new variables $\boldsymbol{z}$ and define the joint posterior $p(\boldsymbol{\theta}, \boldsymbol{z})$ such that $\int_Z p(\boldsymbol{\theta}, \boldsymbol{z}) d\boldsymbol{z} = p(\boldsymbol{\theta})$. If the conditionals of the joint posterior are easy to sample from then the Gibbs sampler can be applied to draw samples $(\boldsymbol{\theta}, \boldsymbol{z})$.

For mixtures such completion is naturally derived from the missing structure of the problem by introducing indicator variables associating observations to components as we have seen in Equation 3.3. By assuming conjugate priors as in Section 3.3, the conditional distributions which can be used in the Gibbs sampler follow as (McLachlan & Peel 2000)

$$
\begin{aligned}
p(\boldsymbol{z}_i|\boldsymbol{X}, \boldsymbol{\Theta}) &= \mathcal{M}(\boldsymbol{z_i}|1, \rho_{i,1}, \ldots, \rho_{i,K}), && (3.16)\\
p(\boldsymbol{\mu}_k|\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\Sigma}_k) &= \mathcal{N}\left(\boldsymbol{\mu}_k|\boldsymbol{m}_k, (\beta + n_k)^{-1}\boldsymbol{\Sigma}_k\right), && (3.17)\\
p(\boldsymbol{\Sigma}_k^{-1}|\boldsymbol{X}, \boldsymbol{Z}, \boldsymbol{\mu}_k) &= \mathcal{IW}\left(\boldsymbol{\Sigma}_k|\boldsymbol{W}_k, n_k + v_0\right), && (3.18)\\
p(\boldsymbol{\pi}|\boldsymbol{Z}) &= \mathcal{D}(\boldsymbol{\pi}|\boldsymbol{a}_0 + n_k), && (3.19)
\end{aligned}
$$

where we have defined

$$
\rho_{i,k} = \frac{\pi_k \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{k'=1}^{K} \pi_{k'} \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_{k'}, \boldsymbol{\Sigma}_{k'})}, \quad \boldsymbol{m}_k = \frac{\boldsymbol{m}_0 \beta + n_k \bar{\boldsymbol{x}}_k}{\beta + n_k},
$$

$$
\boldsymbol{W}_k = \boldsymbol{W}_0^{-1} + n_k \boldsymbol{V}_k + \frac{n_k v_0}{n_k + v_0} (\bar{\boldsymbol{x}} - \boldsymbol{m_0})(\bar{\boldsymbol{x}} - \boldsymbol{m_0})^T,
$$

$$\boldsymbol{V}_k = \frac{1}{n_k}\sum_{i=1}^{N} z_{i,k}(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k)(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k)^T, \quad \bar{\boldsymbol{x}}_k = \frac{1}{n_k}\sum_{i=1}^{N} z_{i,k}\boldsymbol{x}_i, \quad n_k = \sum_{i=1}^{N} z_{i,k}.$$

The Gibbs sampler for mixtures of Gaussians is given in Algorithm 3.

Algorithm 3: Gibbs sampler for Gaussian mixture models

1: Inititialise $\boldsymbol{\Theta}^0$
2: **for** $t = 1$ to T **do**
3: $\quad \boldsymbol{Z}^t \sim p(\boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{\Theta}^{t-1})$ using Eq. (3.16)
4: $\quad \boldsymbol{\mu}_k^t \sim p(\boldsymbol{\mu}_k|\boldsymbol{X}, \boldsymbol{Z}^t, \boldsymbol{\Sigma}_k^{t-1}), \quad \forall k \in \{1, \ldots, K\}$ using Eq. (3.17)
5: $\quad \boldsymbol{\Sigma}_k^t \sim p(\boldsymbol{\Sigma}_k|\boldsymbol{X}, \boldsymbol{Z}^t, \boldsymbol{\mu}_k^t), \quad \forall k \in \{1, \ldots, K\}$ using Eq. (3.18)
6: $\quad \boldsymbol{\pi}^t \sim p(\boldsymbol{\pi}|\boldsymbol{Z}^t)$ using Eq. (3.19)
7: **end for**

Although the Gibbs sampler is simple to implement and no tuning is required, simulation involves sampling the indicator variables which are in the order of the observations therefore increasing the state space considerably. Moreover, sampling the mixture parameters conditioned on indicator variables and vice versa, implies that the length of the transitions is constrained since for a particular value of the indicator variables the mixture parameters are very concentrated and therefore their new value will not allow for large moves of the indicator variables in the next iteration (Marin et al. 2005). Finally, the Gibbs sampler can be very sensitive to initialisation, especially when there are local modes in the vicinity of the starting position. An interesting illustration of these problems can be found in (Marin et al. 2005).

Methods for improving the efficiency of Gibbs sampling include the ordered overrelaxation (Neal 1999), which is applicable when the cumulative and inverse cumulative functions of the conditional distributions can be efficiently calculated.

3.4.4 Manifold Metropolis Adjusted Langevin Algorithm

Denoting the log of the target density as $\mathcal{L}(\boldsymbol{\theta}) = \log p(\boldsymbol{\theta})$, the manifold MALA (mMALA) method, (Girolami & Calderhead 2011), defines a Langevin diffusion with stationary distribution $p(\boldsymbol{\theta})$ on the Riemann manifold of density functions with metric tensor $\boldsymbol{G}(\boldsymbol{\theta})$. By employing a first order Euler integrator to solve the diffusion, a proposal mechanism with density $q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{t-1}) = \mathcal{N}(\boldsymbol{\theta}^*|\boldsymbol{\mu}(\boldsymbol{\theta}^{t-1}, \epsilon), \epsilon^2\boldsymbol{G}^{-1}(\boldsymbol{\theta}^{t-1}))$ is obtained, where ϵ is the integration step size, a parameter which needs to be tuned, and the dth component of the mean function $\boldsymbol{\mu}(\boldsymbol{\theta}, \epsilon)_d$ is

$$\boldsymbol{\mu}(\boldsymbol{\theta}, \epsilon)_d = \boldsymbol{\theta}_d + \frac{\epsilon^2}{2}\left(\boldsymbol{G}^{-1}(\boldsymbol{\theta})\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta})\right)_d - \epsilon^2 \sum_{i=1}^{D}\sum_{j=1}^{D} \boldsymbol{G}(\boldsymbol{\theta})^{-1}_{i,j}\Gamma^d_{i,j}, \quad (3.20)$$

where $\Gamma^d_{i,j}$ are the Christoffel symbols of the metric in local coordinates (Kühnel 2005).

Similarly to MALA (Roberts & Stramer 2003), due to the discretisation error introduced by the first order approximation, convergence to the stationary distribution is not guaranteed anymore and thus the Metropolis-Hastings ratio is employed to correct this bias. The mMALA algorithm can be simply stated as in Algorithm 4. Details can be found in (Girolami & Calderhead 2011).

We can interpret the proposal mechanism of mMALA as a local Gaussian approximation to the target density similar to the adaptive Metropolis-Hastings of Haario et al. (1998). In contrast to Haario et al. (1998) however,

Algorithm 4: mMALA

Inititialise $\boldsymbol{\theta}^0$
for $t = 1$ to T **do**
    $\boldsymbol{\theta}^* \sim \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}(\boldsymbol{\theta}^{t-1}, \epsilon), \epsilon^2 \boldsymbol{G}^{-1}(\boldsymbol{\theta}^{t-1}))$
    $r = \min\left\{1, p(\boldsymbol{\theta}^*)q(\boldsymbol{\theta}^{t-1}|\boldsymbol{\theta}^*)/p(\boldsymbol{\theta}^{t-1})q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{t-1})\right\}$
    $u \sim \mathcal{U}_{[0,1]}$
    if $r > u$ **then**
        $\boldsymbol{\theta}^t = \boldsymbol{\theta}^*$
    else
        $\boldsymbol{\theta}^t = \boldsymbol{\theta}^{t-1}$
    end if
end for

the effective covariance matrix in mMALA is the inverse of the metric tensor evaluated at the current position and no samples from the chain are required in order to estimate it, therefore avoiding the difficulties of adaptive MCMC discussed in (Andrieu & Thoms 2008). Furthermore a simplified version of the mMALA algorithm can also be derived by assuming a manifold with constant curvature, thus cancelling the last term in Equation (3.20) which depends on the Christoffel symbols. Finally, the mMALA algorithm can be seen as a generalisation of the original MALA (Roberts & Stramer 2003) since, if the metric tensor $\boldsymbol{G}(\boldsymbol{\theta})$ is equal to the identity matrix corresponding to an Euclidean manifold, then the original algorithm is recovered.

Similar to the MH algorithm the mMALA family of algorithms requires the parameters to be vectors in $\mathbb{R}^D$. In Section 3.4.2 we have shown how the model can be re-parameterised in order to apply the MH algorithm. For mMALA we can use the same re-parameterisation however we also need to derive the gradient $\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta})$, Metric tensor $\boldsymbol{G}(\boldsymbol{\theta})$ and the first order derivatives of $\boldsymbol{G}(\boldsymbol{\theta})$ with respect to the parameters $\boldsymbol{\theta}$.

3.4.5 Metric Tensor for Mixtures of Gaussians

For density functions the natural metric tensor is the expected Fisher Information, $\boldsymbol{I}(\boldsymbol{\theta})$, (Amari & Nagaoka 2000) which is non-analytic for mixture models. In this work an estimate of the Fisher information, the empirical Fisher information as defined in (McLachlan & Peel 2000, chap. 2) is used as the metric tensor, and its form is given in Equation (3.21) where we have defined the $N \times D$ score matrix $\boldsymbol{S}$ with elements $S_{i,d} = \frac{\partial \log p(x_i|\boldsymbol{\theta})}{\partial \theta_d}$ and $\boldsymbol{s} = \sum_{i=1}^{N} \boldsymbol{S}_{i,\cdot}^T$ which is the gradient. The derivatives of the empirical Fisher information are also easily computed as they require calculation of the second derivatives of the log likelihood and their general form is given in Equation (3.22).

$$\boldsymbol{G}(\boldsymbol{\theta}) = \boldsymbol{S}^T\boldsymbol{S} - \frac{1}{N}\boldsymbol{s}\boldsymbol{s}^T. \tag{3.21}$$

$$\frac{\partial \boldsymbol{G}(\boldsymbol{\theta})}{\partial \theta_d} = \left(\frac{\partial \boldsymbol{S}^T}{\partial \theta_d}\boldsymbol{S} + \boldsymbol{S}^T\frac{\partial \boldsymbol{S}}{\partial \theta_d}\right) - \frac{1}{N}\left(\frac{\partial \boldsymbol{s}}{\partial \theta_d}\boldsymbol{s}^T + \boldsymbol{s}\frac{\partial \boldsymbol{s}^T}{\partial \theta_d}\right). \tag{3.22}$$

The first and second order partial derivatives of the log likelihood for Gaussian mixture models with respect to the parameters as well as the corresponding partial derivatives for the prior, are more involved and are omitted from the main text.

Another alternative which is not explored further in this thesis is to approximate locally the Fisher Information metric. The metric tensor is defined by the choice of the distance function between two densities. The Fisher information can be derived by taking a first order expansion of the symmetric Kullback Liebler (KL) divergence between two densities, the Hellinger dis-

tance yields the same metric as it provides a bound on the KL divergence. The choice of distance function therefore dictates the form of the metric tensor and we can see that the analytical intractability of the expected Fisher information for mixture models is "inherited" from the Kullback Liebler divergence. However, by considering a different divergence it is possible to define a metric tensor that has an analytic form. In fact, if we consider the L_2 distance between two densities given by

$$\int_{\mathcal{X}} |p(x|\boldsymbol{\theta} + \delta\boldsymbol{\theta}) - p(x|\boldsymbol{\theta})|^2 \, dx,$$

and take a first order expansion, the metric tensor under the L_2 metric is

$$\int_{\mathcal{X}} \nabla_{\boldsymbol{\theta}} p(x|\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}}^T p(x|\boldsymbol{\theta}) dx,$$

where for mixtures of Gaussian densities the integral is analytic. The L_2 metric is a special case of the power divergence (Basu et al. 1998) and has been used in robust estimation of parametric models in (Scott 2001). Although we do not explore these ideas further in this thesis we believe that this is an interesting topic for further research.

3.4.6 Manifold Hamiltonian Monte Carlo

The Riemann manifold Hamiltonian Monte Carlo (RM-HMC) method defines a Hamiltonian on the Riemann manifold of probability density functions by introducing the auxiliary variables $\boldsymbol{p} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{G}(\boldsymbol{\theta}))$, which are interpreted as the momentum at a particular position $\boldsymbol{\theta}$ and by considering the nega-

tive log of the target density as a potential function. More formally, the Hamiltonian defined on the Riemann manifold is:

$$H(\boldsymbol{\theta}, \boldsymbol{p}) = -\mathcal{L}(\boldsymbol{\theta}) + \frac{1}{2}\log\left[2\pi\text{det}(\boldsymbol{G}(\boldsymbol{\theta}))\right] + \frac{1}{2}\boldsymbol{p}^T\boldsymbol{G}(\boldsymbol{\theta})^{-1}\boldsymbol{p}, \tag{3.23}$$

where the terms $-\mathcal{L}(\boldsymbol{\theta}) + \frac{1}{2}\log\left[2\pi\text{det}(\boldsymbol{G}(\boldsymbol{\theta}))\right]$ and $\frac{1}{2}\boldsymbol{p}^T\boldsymbol{G}(\boldsymbol{\theta})^{-1}\boldsymbol{p}$ are the potential energy and kinetic energy terms, respectively. Simulating the Hamiltonian requires a time-reversible and volume preserving numerical integrator. For this purpose the Generalised Leapfrog algorithm can be employed and provides a deterministic proposal mechanism for simulating from the conditional distribution, i.e. $\boldsymbol{\theta}^*|\boldsymbol{p} \sim p(\boldsymbol{\theta}^*|\boldsymbol{p})$. More details about the Generalised Leapfrog integrator can be found in (Girolami & Calderhead 2011). To simulate a path across the manifold, the Leapfrog integrator is iterated L times which along with the integration step size ϵ are parameters requiring tuning. Again, due to the integration errors on simulating the Hamiltonian, in order to ensure convergence to the stationary distribution the Metropolis-Hastings ratio is applied. Moreover, following the suggestion in (Neal 1993) the number of Leapfrog iterations L is randomised in order to improve mixing. The RM-HMC algorithm is given in Algorithm 5.

Similar to the mMALA algorithm, when the metric tensor $\boldsymbol{G}(\boldsymbol{\theta})$ is equal to the identity matrix corresponding to an Euclidean manifold, then RM-HMC is equivalent to the HMC algorithm of (Duane et al. 1987).

Algorithm 5: RM-HMC

Inititialise $\boldsymbol{\theta}^0$
for $t = 1$ to T **do**
  $\boldsymbol{p}_*^0 \sim \mathcal{N}(\boldsymbol{p}|\mathbf{0}, \boldsymbol{G}(\boldsymbol{\theta}^{t-1}))$
  $\boldsymbol{\theta}_*^0 = \boldsymbol{\theta}^{t-1}$
  $e \sim \mathcal{U}_{[0,1]}$
  $N = \text{ceil}(\epsilon L)$
  {Simulate the Hamiltonian using a generalised Leapfrog integrator for N steps}
  for $n = 0$ to N **do**
    solve $\boldsymbol{p}_*^{n+\frac{1}{2}} = \boldsymbol{p}_*^n - \frac{\epsilon}{2}\nabla_{\boldsymbol{\theta}} H\left(\boldsymbol{\theta}_*^n, \boldsymbol{p}_*^{n+\frac{1}{2}}\right)$
    solve $\boldsymbol{\theta}_*^{n+1} = \boldsymbol{\theta}_*^n + \frac{\epsilon}{2}\left[\nabla_{\boldsymbol{p}} H\left(\boldsymbol{\theta}_*^n, \boldsymbol{p}_*^{n+\frac{1}{2}}\right) + \nabla_{\boldsymbol{p}} H\left(\boldsymbol{\theta}_*^{n+1}, \boldsymbol{p}_*^{n+\frac{1}{2}}\right)\right]$
    $\boldsymbol{p}_*^{n+1} = \boldsymbol{p}_*^{n+\frac{1}{2}} - \frac{\epsilon}{2}\nabla_{\boldsymbol{\theta}} H\left(\boldsymbol{\theta}_*^{n+1}, \boldsymbol{p}_*^{n+\frac{1}{2}}\right)$
  end for
  $(\boldsymbol{\theta}^*, \boldsymbol{p}^*) = \left(\boldsymbol{\theta}_*^{N+1}, \boldsymbol{p}_*^{N+1}\right)$
  {Metropolis-Hastings ratio}
  $r = \min\left\{1, \exp\left(-H(\boldsymbol{\theta}^*, \boldsymbol{p}^*) + H(\boldsymbol{\theta}^{t-1}, \boldsymbol{p}^{t-1})\right)\right\}$
  $u \sim \mathcal{U}_{[0,1]}$
  if $r > u$ **then**
    $\boldsymbol{\theta}^t = \boldsymbol{\theta}^*$
  else
    $\boldsymbol{\theta}^t = \boldsymbol{\theta}^{t-1}$
  end if
end for

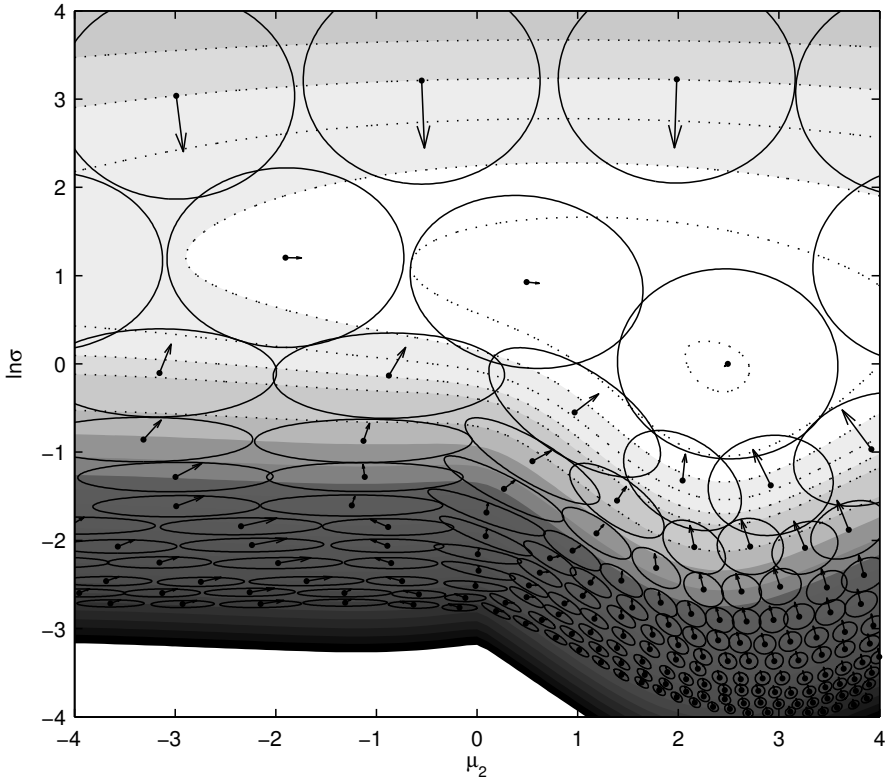

Figure 3.2: Natural gradients and the inverse metric tensor evaluated at different locations. Arrows correspond to the natural gradients and ellipses to the inverse metric tensor. Dashed lines are the isocontours of the joint log likelihood.

3.5 An Illustrative Example

To illustrate the differences between Gibbs, MALA, HMC and their Riemann manifold counterparts we use a simple example for which it is easy to visualise the gradient fields and the paths of the Markov chains. For this example we

use a mixture model of the form

$$p(x|\mu, \sigma^2) = 0.7 \times \mathcal{N}(x|0, \sigma^2) + 0.3 \times \mathcal{N}(x|\mu, \sigma^2), \tag{3.24}$$

where the variance σ^2 is shared for both components. The variance parameter has been re-parameterised as $\sigma^2 = \exp(\gamma)$ and an inverse gamma prior was used. A synthetic dataset of 500 random samples from this model with true parameters set as $\mu = 2.5$ and $\sigma^2 = 1$ was generated and 10,000 samples from the posterior using the Gibbs sampler, MALA, HMC, mMALA and RM-HMC as well as the simplified version of mMALA discussed in Section 3.4 are simulated. For all examples the same starting position was used.

In Figure 3.2 the natural gradients, i.e. the gradient scaled by the inverse metric tensor, and the metric tensor for the simple model in Equation 3.24 are illustrated. The ellipses correspond to the inverse of the metric tensor evaluated at different locations and plotted on top of the joint log likelihood surface. From Figure 3.2 we can see how the metric tensor reflects the local geometry and notice especially the ellipses on the bottom left of the figure which are elongated along the axis where the target density changes less rapidly, i.e. μ_2. For the mMALA algorithm the inverse metric tensor corresponds to the covariance matrix of the proposal mechanism and we can see how the proposal mechanism is optimally adapted allowing for large transitions in areas where the target density is flat and smaller transitions in areas where the target density is steep preventing the algorithm from 'overshooting'.

In Figure 3.3 the differences between MALA and mMALA by comparing

the first 1,000 samples from their chains is illustrated. To ensure a fair comparison the step size parameter for both algorithms was tuned such that the acceptance rate remains between 40% to 60% as discussed in (Christensen et al. 2005) and (Girolami & Calderhead 2011). For MALA achieving such an acceptance rate at stationarity was not trivial since for step sizes above $1E^{-4}$ the algorithm failed to converge in 10,000 samples by rejecting all proposed moves. In (Christensen et al. 2005) such behaviour for the transient phase of MALA is also reported and a different scaling is suggested. We have experimented with both scalings without any significant difference for this particular case. The problem lies in the large discrepancy between the gradient magnitudes at the starting position and around the mode. From Figure 3.2 we can see that at the starting position a small step size is required in order not to 'overshoot' due to the large gradient. On the contrary, around the mode the gradient magnitudes are small suggesting a larger step size in order to obtain less correlated samples.

The sample autocorrelation plots from the stationary chains, i.e. after the chains have converged to the stationary distribution, for both algorithms are also shown in Figure 3.3. We can see that the mMALA algorithm converges rapidly and once converged the correlation between samples is very low. We have seen similar behaviour in our experiments with different starting positions while retaining the same value for the integration step size parameter. Moreover, in Figure 3.3 the convergence path and autocorrelation for simplified mMALA is shown and from which we can see that despite the simplifying assumption of a constant manifold the algorithm remains efficient.

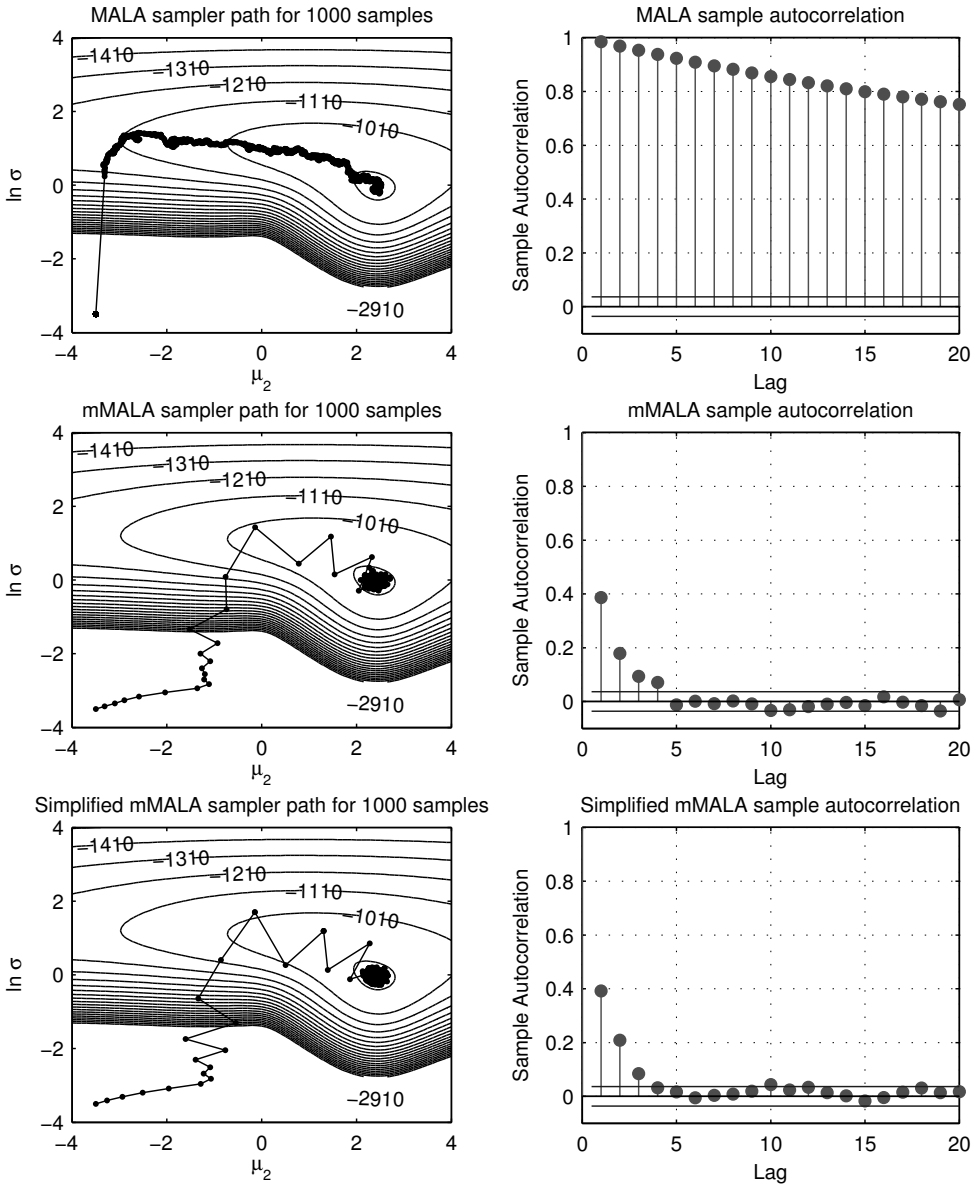

Figure 3.3: Comparison of MALA (top), mMALA (middle) and simplified mMALA (bottom) convergence paths and autocorrelation plots. Autocorrelation plots are from the stationary chains, i.e. once the chains have converged to the stationary distribution.

Similar experiments are performed with HMC and RM-HMC where the step size parameters were selected such that the acceptance rate was above 70% and the number of Leapfrog iterations was fixed at 10 for both algorithms. The results are presented in Figure (3.4). We can again see a similar behaviour as in the previous example with MALA and mMALA. HMC needs several iterations to reach the mode due to the small step size required to escape from the low potential region at the starting position while RM-HMC rapidly converges to the mode and the autocorrelation of the samples remains low. Also note that the RM-HMC algorithm seems more efficient than mMALA, something that is also reported in (Girolami & Calderhead 2011) and relates to the fact that for mMALA a single discretisation step of the diffusion is used to propose a new sample while for RM-HMC the Hamiltonian is simulated for several iterations. Finally similar plots for the Gibbs sampler are presented in Figure (3.4) where we can see that the Gibbs sampler using the full conditional distributions follows a path along the gradients although no gradient information is used explicitly. Despite its fast convergence, however, the chain mixing is not as good as for RM-HMC as indicated by the autocorrelation function.

3.6 Experiments

In this section a set of experiments designed to evaluate the MCMC algorithms in more realistic scenarios than the one presented in Section 3.5 are described. Five different simulated datasets randomly generated from the densities in Table 3.1 taken from (McLachlan & Peel 2000) are considered.

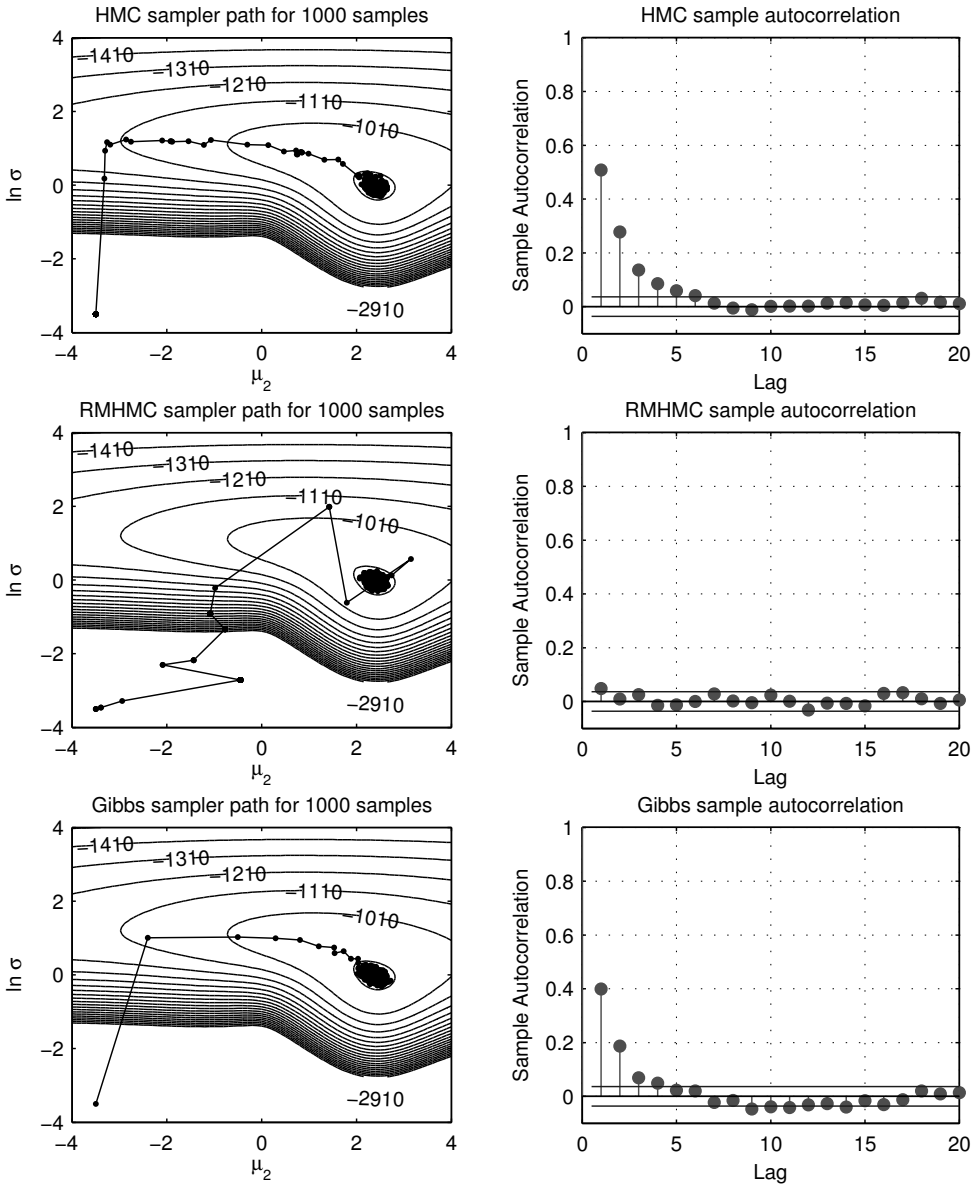

Figure 3.4: Comparison of HMC(top), RM-HMC (middle) and Gibbs (bottom) convergence paths and autocorrelation plots. Autocorrelation plots are from the stationary chains, i.e. once the chains have converged to the stationary distribution.

Dataset name	Density function	Num. of parameters
Kurtotic	$\frac{2}{3}\mathcal{N}(x\|0,1) + \frac{1}{3}\mathcal{N}\left(x\|0,\left(\frac{1}{10}\right)^2\right)$	6
Bimodal	$\frac{1}{2}\mathcal{N}\left(x\|-1,\left(\frac{2}{3}\right)^2\right) + \frac{1}{2}\mathcal{N}\left(x\|1,\left(\frac{2}{3}\right)^2\right)$	6
Skewed bimodal	$\frac{3}{4}\mathcal{N}\left(x\|0,1\right) + \frac{1}{4}\mathcal{N}\left(x\|\frac{3}{2},\left(\frac{1}{3}\right)^2\right)$	6
Trimodal	$\frac{9}{20}\mathcal{N}\left(x\|-\frac{6}{5},\left(\frac{3}{5}\right)^2\right) + \frac{9}{20}\mathcal{N}\left(x\|\frac{6}{5},\left(\frac{3}{5}\right)^2\right) + \frac{1}{10}\mathcal{N}\left(x\|0,\left(\frac{1}{4}\right)^2\right)$	9
Claw	$\frac{1}{2}\mathcal{N}\left(x\|0,1\right) + \sum_{i=0}^{4}\frac{1}{10}\mathcal{N}\left(x\|\frac{i}{2}-1,\left(\frac{1}{10}\right)^2\right)$	18

Table 3.1: Densities used for the generation of synthetic datasets.

The densities were selected in order to assess the performance of the methods under different conditions. For example, in the "bimodal" density the modes are well separated and the posterior standard deviations are of the same order. On the other hand the "claw" density has overlapping components of different variances and the posterior standard deviations have large differences. Plots of the densities are shown in Figure 3.5 while the number of parameters is also given in Table 3.1. For the datasets "kurtotic", "bimodal" and "skewed bimodal" 2,000 random samples are generated while for the datasets 'trimodal' and 'claw' we generated 4,000 and 5,000 samples, respectively. Different samples were used in order to guarantee that the different components are well represented in the dataset.

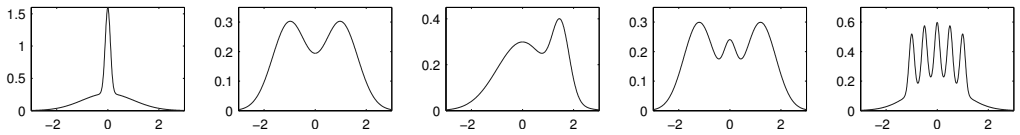

Figure 3.5: Densities used to generate synthetic datasets. From left to right the densities are in the same order as in Table 3.1. The densities are taken from (McLachlan & Peel 2000)

To ensure fair comparison, all algorithms were tuned by following the suggestions in (Roberts et al. 1997, Christensen et al. 2005) and (Neal 1993). More precisely, the Metropolis-Hastings scale parameter was tuned such that acceptance rate was between 20% and 30%. The scale parameter for MALA was set such that acceptance rate was between 40% and 60%. For HMC, the step size parameter ϵ was set at the smallest standard deviation of the marginal posterior and the number of Leapfrog iterations L is set such that ϵL was above the largest standard deviation. Of course such knowledge is not known *a-priori* and pilot runs to obtain the marginal posterior are needed when HMC is applied in practice. For the RM-HMC, mMALA and simplified mMALA, this knowledge of the target density is not required, thus the method in (Girolami & Calderhead 2011) is followed to tune the step size parameters such that the acceptance rate was above 70% and the number of leapfrog steps for RM-HMC was kept fixed to five.

It is interesting to note here that the suggestions for tuning the Metropolis-Hastings and the MALA algorithms (Roberts et al. 1997, Christensen et al. 2005) assume stationarity. Therefore tuning those algorithms requires several pilot runs where the chains are simulated until convergence and the acceptance rate is measured to ensure that it lies between the suggested values. Similarly, tuning HMC requires pilot runs in order to obtain the posterior standard deviations. In contrast, the tuning of mMALA and RM-HMC does not require such pilot runs, instead the first set of parameters that achieved an acceptance rate above 70% were used to set the step size. The acceptance rate during the transient and stationary phases of mMALA and RM-HMC

were at the same levels across all our experiments.

All algorithms run for 100,000 iterations and only the last 10,000 samples are kept as samples from the posterior, provided that the chains have converged. Convergence was assessed by inspection of the trace plots. The posterior means were also compared with the true values provided in Table 3.1 to ensure that the chains have converged to the correct mode. In the experiments presented here, we have not encountered the label switching problem in any of the algorithms or datasets. This is due to the careful tuning of samplers to achieve relatively high acceptance rates. In pilot runs, however, where larger step sizes and scales were used, we have observed components switching labels from Metropolis-Hastings, HMC and RM-HMC, although the acceptance rates were below the suggested values. The lack of label switching due to different tuning of parameters is also reported in (Marin et al. 2005). Finally, from the output of each algorithm the minimum, median and maximum Effective Sample Size (ESS) (Geyer 1992) across parameters as well as the number of samples per second are measured. The experiments where repeated 10 times and all results presented in Table 3.2 are averages over the 10 different runs.

From Table 3.2 we can immediately see that in terms of raw ESS, mMALA and simplified mMALA were always better than MALA and Metropolis-Hastings. Similarly RM-HMC was in all cases better than HMC. This highlights that by exploiting the intrinsic geometry of the model in MCMC we can achieve superior mixing. Another interesting observation is that the correlation of samples across different parameters (minESS and max ESS) is

Dataset	Algorithm	Min ESS	Med ESS	Max ESS	minESS/sec.
Kurtotic	M.-H.	37 ±10	133 ±24	1347 ±58	4.5 ±1.2
	MALA	19 ±7	78 ±10	5172 ±304	0.7 ±0.2
	mMALA	158 ±13	193 ±13	235 ±15	2.6 ±0.1
	simp. mMALA	173 ±15	206 ±16	231 ±12	4.5 ±0.4
	HMC	892 ±42	4167 ±412	10000 ±0	4.6 ±0.2
	RM-HMC	9759 ±56	9961 ±89	10000 ±0	**28.9 ±0.6**
	Gibbs	1728 ±173	3102 ±73	9733 ±230	9.0 ±0.8
Bimodal	M.-H.	83 ±13	103 ±17	119 ±23	10.2 ±1.7
	MALA	141 ±26	169 ±31	187 ±35	5.4 ±1.0
	mMALA	695 ±20	798 ±53	869 ±55	9.02 ±0.3
	simp. mMALA	774 ±24	817 ±27	937 ±86	20.4 ±0.7
	HMC	1509 ±149	1675 ±144	1747 ±142	**24.4 ±2.4**
	RM-HMC	4593 ±128	4920 ±184	5215 ±152	13.7 ±0.3
	Gibbs	473 ±97	504 ±93	574 ±97	2.5 ±0.5
Skewed	M.-H.	50 ±9	68 ±17	259 ±102	6.2 ±1.2
	MALA	77 ±9	113 ±19	312 ±51	2.9 ±0.3
	mMALA	437 ±59	511 ±63	669 ±91	5.6 ±0.7
	simp. mMALA	537 ±86	587 ±77	703 ±96	**14.2 ±2.3**
	HMC	1491 ±57	1849 ±93	3613 ±187	**14.1 ±0.5**
	RM-HMC	4793 ±639	5152 ±704	6969 ±690	**14.7 ±1.5**
	Gibbs	407 ±40	469 ±53	1032 ±83	2.1 ±0.2
Trimodal	M.-H.	10 ±3	42 ±17	136 ±30	0.6 ±0.1
	MALA	20 ±4	75 ±23	312 ±11	0.3 ±0.07
	mMALA	209 ±44	272 ±21	342 ±24	0.9 ±0.2
	simp. mMALA	224 ±13	272 ±15	319 ±37	**2.5 ±0.1**
	HMC	582 ±79	1713 ±262	6851 ±363	1.9 ±0.2
	RM-HMC	2066 ±117	2369 ±176	2622 ±175	**2.3 ±0.1**
	Gibbs	205 ±31	381 ±40	621 ±29	0.5 ±0.08
Claw	M.-H.	-	-	-	-
	MALA	-	-	-	-
	mMALA	79 ±11	145 ±15	222 ±28	0.09 ±0.01
	simp. mMALA	95 ±13	154 ±7	242 ±30	0.4 ±0.05
	HMC	923 ±215	1739 ±140	6529 ±70	**0.5 ±0.1**
	RM-HMC	1756 ±427	2075 ±503	2825 ±815	0.3 ±0.06
	Gibbs	238 ±71	966 ±51	4197 ±367	**0.5 ±0.1**

Table 3.2: Evaluation of MCMC algorithms for univariate mixtures of normals. ESS is estimated using 10,000 samples from the posterior. minESS/sec denotes the number of uncorrelated samples per second. The Metropolis-Hastings and MALA algorithms failed to converge after 100,000 samples in almost all of the 10 runs for the 'claw' dataset.

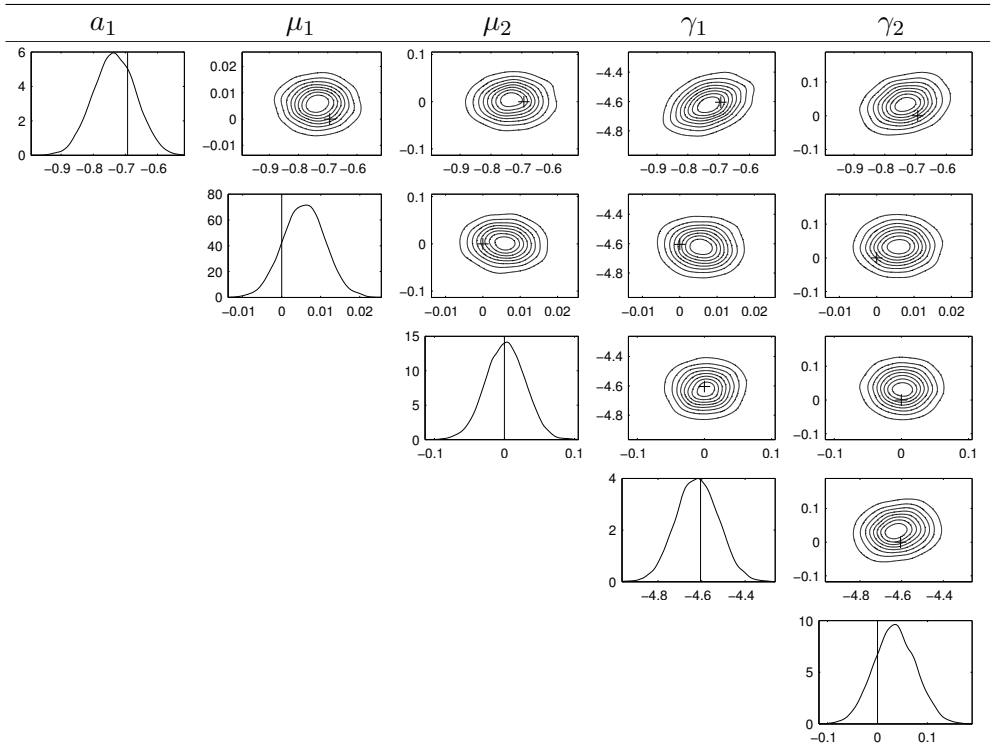

Table 3.3: Posterior distribution of the kurtotic dataset. True values are depicted by vertical lines or the plus sign.

almost the same for RM-HMC and mMALA while for all other algorithms it varies significantly. This is related to the difficulties in tuning the scale and step size parameters of MALA and HMC. When the standard deviations of the marginal posterior exhibit very large differences in scale it is very difficult to tune MALA and HMC to achieve good mixing across all parameters. For RM-HMC and mMALA the gradients are appropriately scaled by the metric tensor and therefore smaller variation across the parameters is expected making the algorithms less sensitive to tuning parameters. This argument is also supported by the results of Metropolis-Hastings, MALA and HMC for the "bimodal" dataset where the difference between minESS and maxESS is low suggesting a small difference in scale of the posteriors standard deviation.

Indeed, the minimum and maximum standard deviations of the posterior for the 'bimodal' dataset was found to be 0.0402 and 0.0822, respectively. Finally it is interesting to note that in terms of raw ESS the Gibbs sampler was only better 2 out of 5 times compared to mMALA and simplified mMALA while it was always worse than RM-HMC.

In practice, however, raw ESS is not very informative about the practical significance of an MCMC algorithm since less computationally demanding algorithms can obtain the required number of effectively independent samples faster than a more demanding but more efficient (in terms of mixing) algorithm. For that reason the number of effectively uncorrelated samples per second (minESS/sec) are also reported. The results suggest that for the most difficult examples such as the "trimodal", "claw" and "kurtotic" datasets the RM-HMC algorithm is always better while the less computationally demanding simplified mMALA performs almost equally in some cases. Interestingly, the Gibbs sampler which is the most widely used algorithm for inference in mixture models is always substantially worst when compared to some of the other MCMC methods with the exception of the claw dataset.

In Table 3.3 the posterior for the "kurtotic" dataset is shown. Notice the large difference in variance for μ_1 compared to all other parameters as well as the correlation between α_1 and γ_1,γ_2. Parameter γ_1 corresponds to the log of the variance of the second component of the 'kurtotic' density in Table 3.1, where its density is depicted in Figure 3.5, and the second component corresponds to the peak around 0. When the variance of both components increases the observations around 0 have higher likelihood under the second

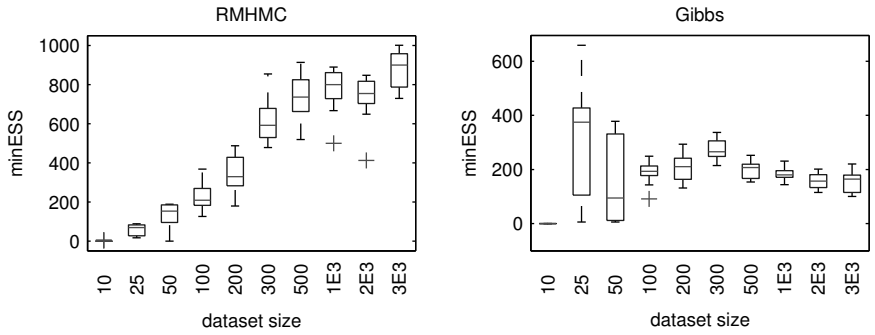

Figure 3.6: Comparison of minESS for the RM-HMC (left) and Gibbs (right) samplers for different dataset sizes. For datasets of size 10 both algorithms failed to converge after 10,000 iterations.

component thus increasing the value of its mixing coefficient and therefore explaining the correlation in Table 3.3.

The datasets used in this study have a relatively large size and since a finite sample estimate of the Fisher information is used, it is unknown if asymptotic behaviour is affecting the comparisons. For this reason further experiments were conducted using the "kurtotic" dataset where datasets of different size have been created and the minimum effective sample size of Gibbs and RM-HMC samplers is compared. For these experiments 10,000 burn-in samples are used and ESS is measured using 1,000 posterior samples provided the chains have converged. All experiments are repeated for 10 runs and results are presented in Figure 3.6. The minimum ESS for RM-HMC is increasing with the size of the dataset and after 500 observations we can see that it remains almost constant. In comparison, the Gibbs sampler is independent of the sample size as it was expected. Moreover, we can see that even for small datasets RM-HMC has superior mixing properties when

compared to the Gibbs sampler.

3.7 Approximate Bayesian Inference

In the previous sections we have seen methods that in the limit of infinite samples can produce exact results. However these methods are computationally demanding and they often require human intervention. For example tuning the algorithms requires several pilot runs and monitoring of the acceptance rate. Moreover, in order to check convergence to the stationary distribution, multiple chains have to be run and the variance within and across chains need to be monitored, see Brooks & Gelman (1998) for example. In this section we consider approximate methods for obtaining the posterior distribution and, in particular, Variational Inference (Jordan et al. 1998).

Variational inference, although has its origins in the *mean field theory* of statistical physics (Parisi 1988), has been recently popularised in the Machine Learning and Bayesian statistics communities for solving large and computationally demanding problems. In contrast to MCMC algorithms, variational inference does not produce samples from the posterior rather it provides analytical approximations by assuming that the posterior factorizes between sets of variables or has a specific parametric form.

More formally, for a target posterior $p(\boldsymbol{\Theta}|\boldsymbol{X})$ we assume that can be approximated by $q(\boldsymbol{\Theta})$ and we can rewrite the log marginal using Jensen's

inequality as

$$\log p(\boldsymbol{X}) = \mathcal{L}_b(q) + \mathrm{KL}(q, p), \tag{3.25}$$

where

$$\mathcal{L}_b(q) = \int q(\boldsymbol{\Theta}) \log \left\{ \frac{p(\boldsymbol{X}, \boldsymbol{\Theta})}{q(\boldsymbol{\Theta})} \right\} d\boldsymbol{\Theta}, \tag{3.26}$$

$$\mathrm{KL}(q, p) = -\int q(\boldsymbol{\Theta}) \log \left\{ \frac{p(\boldsymbol{\Theta}|\boldsymbol{X})}{q(\boldsymbol{\Theta})} \right\} d\boldsymbol{\Theta}. \tag{3.27}$$

Equation (3.26) is a lower bound on the marginal and Equation 3.27 is the KL divergence between the approximate and the true posterior. We can see from Equation 3.25 that by maximising the lower bound the KL divergence is minimised and when the KL divergence vanishes then the approximate posterior is equal to the true posterior distribution.

Variational inference considers posterior approximations $q(\boldsymbol{\Theta})$ that factorize between sets of variables as

$$q(\boldsymbol{\Theta}) = \prod_{i=1}^{M} q_i(\boldsymbol{\Theta}_i)$$

To obtain an analytical approximation for each factorized distribution $q_j(\boldsymbol{\Theta}_j)$ that maximise the lower bound we can substitute the above expression in Equation (3.26) and by considering all terms involving $q_i, i \neq j$ fixed we can

rearrange terms to get:

$$\begin{aligned}\mathcal{L}_b(q) &= \int \prod_i q_i(\boldsymbol{\Theta}_i) \left\{ \log p(\boldsymbol{X}, \boldsymbol{\Theta}) - \sum_i \log q_i(\boldsymbol{\Theta}_i) \right\} d\boldsymbol{\Theta} \\ &= \int q_j(\boldsymbol{\Theta}_j) \log \tilde{p}(\boldsymbol{X}, \boldsymbol{\Theta}_j) d\boldsymbol{\Theta}_j - \int q_j(\boldsymbol{\Theta}_j) \log q_j(\boldsymbol{\Theta}_j) d\boldsymbol{\Theta}_j + const \\ &= \mathrm{KL}(q_j(\boldsymbol{\Theta}_j) | \tilde{p}(\boldsymbol{X}, \boldsymbol{\Theta}_j)) + const, \end{aligned} \tag{3.28}$$

where

$$\log \tilde{p}(\boldsymbol{X}, \boldsymbol{\Theta}_j) = \int \log p(\boldsymbol{X}, \boldsymbol{\Theta}) \prod_{i \neq j} q_i(\boldsymbol{\Theta}_i) d\boldsymbol{\Theta}_i = \mathbb{E}_{i \neq j} \left[\log p(\boldsymbol{X}, \boldsymbol{\Theta}) \right], \tag{3.29}$$

is the expectation of the joint log likelihood with respect to the approximate distributions $q_i, i \neq j$. From Equation (3.28) we see that the lower bound with respect to the approximate distribution q_j is maximised when the KL divergence with respect to the expectation (3.29) is minimised.

In the variational inference framework the factorisation is constructed in such a way that the expectations in (3.29) can be evaluated analytically. Although this restricts us in considering only conjugate priors, it is analytically convenient since the optimal solutions of the approximate posteriors are then equal to the expectations in Equation (3.29) which are frequently referred to in the literature as *variational distributions*.

Finally, since each variational posterior q_j is optimised by considering all other distributions $q_i, i \neq j$ fixed, an iterative algorithm similar to EM that updates each variational posterior conditioned on the rest guarantees convergence of the lower bound to a local maximum. The general algorithm

for variational inference, Variational Expectation Maximisation (VEM) is given in Algorithm 6.

Algorithm 6: Variational Expectation Maximisation

1: Inititialise $q_j(\boldsymbol{\Theta}_j)$ for all $j \in \{1, \ldots, M\}$
2: **repeat**
3: **for** $j = 1$ to M **do**
4: $q_j(\boldsymbol{\Theta}_j) = \int \log p(\boldsymbol{X}, \boldsymbol{\Theta}) \prod_{i \neq j} q_i(\boldsymbol{\Theta}_i) d\boldsymbol{\Theta}_i$
5: **end for**
6: **until** convergence

3.7.1 Variational Inference for Gaussian Mixtures

For mixture models the integrals in Equation (3.29) are not analytically tractable even when conjugate priors are considered due to the summation over components in Equation (3.2). Here we follow the same treatment as in (Bishop 2006, Chap. 7) and Nasios & Bors (Aug. 2006) to obtain the variational posteriors for the mixture model parameters. As discussed in Section 3.4.3 a completion of the posterior $p(\boldsymbol{\Theta}|\boldsymbol{X})$ can be constructed by introducing the latent variables $\boldsymbol{Z}$ associating observations to components. Treating $\boldsymbol{Z}$ as parameters, the joint posterior $p(\boldsymbol{\Theta}, \boldsymbol{Z}|\boldsymbol{X})$ can be approximated by a variational distribution which factorizes as $q(\boldsymbol{\Theta}, \boldsymbol{Z}) = q(\boldsymbol{\Theta})q(\boldsymbol{Z})$. Therefore the only assumption made is that the posterior factorizes between latent variables and parameters.

Using the general result of equation (3.29) we can obtain the variational posteriors for the latent variables and model parameters (Bishop 2006, Chap. 7). However, due to the assumed factorisation, the conditional independence

as expressed by the graphical structure in Figure 3.1 and the prior, the variational posterior can be further factored as a product of independent variational posteriors with form:

$$q(\boldsymbol{Z}, \boldsymbol{\Theta}) = q(\boldsymbol{\pi}) \left(\prod_{n=1}^{N} q(\boldsymbol{z}_n) \right) \left(\prod_{k=1}^{K} q(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right).$$

Note that this factorisation is not another simplifying assumption rather is induced by the choice of prior and variable independence. Finally, the variational posteriors take the form of a Multinomial for the latent variables, Dirichlet for the mixing coefficients and Normal-Inverse Wishart for the means and covariance matrices. More formally

$$q(\boldsymbol{\pi}) = \mathcal{D}(\boldsymbol{\pi}|\boldsymbol{p}), \tag{3.30}$$

$$q(\boldsymbol{z}_n) = \mathcal{M}(\boldsymbol{z_n}|1, r_{n,1}, \ldots, r_{n,K}), \tag{3.31}$$

$$q(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \mathcal{N}\left(\boldsymbol{\mu}_k|\boldsymbol{m}_k, \beta_k^{-1}\boldsymbol{\Sigma}_k\right) \mathcal{IW}\left(\boldsymbol{\Sigma}_k|\boldsymbol{W}_k, v_k\right). \tag{3.32}$$

The parameters of the variational posteriors are

$$\begin{aligned} r_{n,k} &= \rho_{n,k} / \sum_{k'=1}^{K} \rho_{n,k'}, \quad p_k = a_0 + n_k, \quad v_k = v_0 + n_k, \\ \boldsymbol{m}_k &= \frac{\boldsymbol{m}_0\beta + n_k\bar{\boldsymbol{x}}_k}{\beta_k}, \quad \beta_k = \beta_0 + n_k, \\ \boldsymbol{W}_k &= \boldsymbol{W}_0^{-1} + n_k\boldsymbol{V}_k + \frac{n_k\beta_0}{n_k + \beta_0}(\bar{\boldsymbol{x}} - \boldsymbol{m_0})(\bar{\boldsymbol{x}} - \boldsymbol{m_0})^T, \end{aligned}$$

where we have defined

$$\begin{aligned}
\log \rho_{n,k} &= \mathbb{E}_{\pi_k}[\log \pi_k] - \frac{1}{2}\mathbb{E}_{\boldsymbol{\Sigma}_k}[\log(\det(\boldsymbol{\Sigma}_k))] - \frac{D}{2}\log(2\pi) \\
&- \frac{1}{2}\mathbb{E}_{\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k}\left[(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)\right], \qquad (3.33) \\
n_k &= \sum_{n=1}^{N} \mathbb{E}_{z_{n,k}}[z_{n,k}], \qquad (3.34) \\
\bar{\boldsymbol{x}}_k &= \frac{1}{n_k}\sum_{n=1}^{N} \mathbb{E}_{z_{n,k}}[z_{n,k}]\boldsymbol{x}_n, \qquad (3.35) \\
\boldsymbol{V}_k &= \frac{1}{n_k}\sum_{n=1}^{N} \mathbb{E}_{z_{n,k}}[z_{n,k}](\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k)(\boldsymbol{x}_i - \bar{\boldsymbol{x}}_k)^T. \qquad (3.36)
\end{aligned}$$

It is interesting to note here the similarity of the variational posteriors with the conditional distributions for the Gibbs sampler in Section 3.4.3. In fact the variational distributions for mixture models are the same as the full conditionals with the exception that for each variational posterior the expectations with respect to all other variables are taken. The expectations in equations (3.33-3.36) are with respect to the variational posteriors (3.30-3.32) and can be calculated analytically (Bishop 2006, Chap. 7). Thus the VEM algorithm can be seen as an approximation to the Gibbs sampler where instead of drawing samples, a point estimate of the variational parameters that maximises the lower bound is obtained. The VEM algorithm for Gaussian mixture models is given in Algorithm 7.

By adopting the variational inference framework we can obtain approximate estimates of the posterior without the need of running multiple MCMC chains and tuning parameters other than selecting appropriate prior hyperparameters. Moreover, the variational posteriors also allow us to calculate

Algorithm 7: VEM-GMM

Initialise $p_k, \beta_k, v_k, \boldsymbol{m}_k, \boldsymbol{W}_k$
repeat
  {Update $q(\boldsymbol{Z})$}
  Calculate $\log \rho_{n,k}$, equation (3.33)
  $r_{n,k} = \rho_{n,k} / \sum_{k'=1}^{K} \rho_{n,k'}$
  {Update $q(\boldsymbol{\Theta})$}
  Calculate $n_k, \bar{\boldsymbol{x}}_k, \boldsymbol{V}_k$, equations (3.34-3.36)
  $p_k = a_0 + n_k, \quad v_k = v_0 + n_k, \quad \beta_k = \beta_0 + n_k$
  $\boldsymbol{m}_k = \dfrac{\boldsymbol{m}_0 \beta + n_k \bar{\boldsymbol{x}}_k}{\beta_k}$
  $\boldsymbol{W}_k = \boldsymbol{W}_0^{-1} + n_k \boldsymbol{V}_k + \dfrac{n_k \beta_0}{n_k + \beta_0} (\bar{\boldsymbol{x}} - \boldsymbol{m_0})(\bar{\boldsymbol{x}} - \boldsymbol{m_0})^T$
until convergence

analytically the predictive density in contrast to Monte Carlo numerical estimates required when only samples from the posterior are available.

For Gaussian mixture models the predictive density for new samples $\boldsymbol{x}^*$ can be calculated by marginalising the parameters with respect to the variational posteriors giving

$$\begin{aligned} p(\boldsymbol{x}^*|\boldsymbol{X}) &= \sum_{k=1}^{K} \int\int\int \pi_k \mathcal{N}(\boldsymbol{x}^*|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) q(\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) d\boldsymbol{\pi} d\boldsymbol{\mu} d\boldsymbol{\Sigma} \\ &= \frac{1}{\hat{\boldsymbol{p}}} \sum_{k=1}^{K} p_k \mathrm{St}(\boldsymbol{x}^*|\boldsymbol{m}_k, \boldsymbol{\Lambda}_k, \nu_k + 1 - D), \end{aligned} \tag{3.37}$$

where $\hat{\boldsymbol{p}} = \sum_{k=1}^{K} p_k$, $\mathrm{St}(\cdot)$ is the multivariate Student t distribution and

$$\boldsymbol{\Lambda}_k = \frac{(\nu_k + 1 - D)\beta_k}{1 + \beta_k} \boldsymbol{W}_k^{-1}.$$

3.8 Model Selection

In this section we briefly discuss methods for Bayesian model selection in the context of mixture models, i.e. selecting the number of components K. As discussed in Section 3.2, different models can be compared based on their marginal likelihoods $\int p(\boldsymbol{X}|\boldsymbol{\Theta}p(\boldsymbol{\Theta})d\boldsymbol{\Theta}$ by calculating the Bayes factor

$$B_{1,2} = \frac{p(\boldsymbol{X}|M_1)}{p(\boldsymbol{X}|M_2)},$$

for two competing models M_1 and M_2. In the context of mixture models $p(\boldsymbol{X}|M_1)$ and $p(\boldsymbol{X}|M_1)$ correspond to the marginal likelihoods of models with different number of components. Once the Bayes factor is computed its value can be interpreted using Table 3.4 provided by Kass & Raftery (1995) which gives the evidence in favour of the first model M_1. Finally the marginal likelihoods can be obtained by utilising samples from the posterior and compute the harmonic mean of the likelihood values

$$\int p(\boldsymbol{X}|\boldsymbol{\Theta})p(\boldsymbol{\Theta})d\boldsymbol{\Theta} = \left\{ \frac{1}{S} \sum_{t=1}^{S} p(\boldsymbol{X}|\boldsymbol{\Theta_t})^{-1} \right\}^{-1},$$

where S is the number of samples from the posterior. For more details see Newton & Raftery (1994).

As it is clear from the above, the standard method for Bayesian model selection involves running several chains for different models and computing multiple Bayes factors, one for each pair of competing models. As this is a computationally demanding task an alternative proposed in Green (1995)

is to construct MCMC samplers which jump between parameter subspaces of different dimensionality thus allowing to obtain samples from effectively all possible models. In the reversible-jumbs scheme the model identifier, i.e. K in the context of mixture models, essentially becomes part of the model parameters and appropriate steps to add, remove, split or merge components are added at each iteration of the MCMC sampler. Implementing such steps however is far from trivial when multivariate components are considered since all steps have to leave the overall dispersion constant and the covariance, or precision matrices, also need to remain positive definite. Dellaportas & Papageorgiou (2006) suggest constructing such steps using random permutations of the covariance matrices' eigen-vectors. The computation of eigen-vectors for each covariance matrix at each MCMC iteration restricts however the applicability of their method to problems with a relatively small number of dimensions due to the computational overhead.

Similarly, in the Variational approximation framework the VEM algorithm has to be run for several models with different number of components and compare the lower bound provided that a term of $\log K!$ is added to account for the K factorial equivalent modes due to the weak identifiability of mixture models (Jasra et al. 2005).

In this Thesis we consider an alternative approach which does not require multiple runs of the VEM algorithm. Initially the algorithm is started using a large number of components, for example the number of observations. The prior over the mixing coefficients $\boldsymbol{\pi}$ is then set such that a-priori sparse solutions are favoured. This can be achieved by setting the α hyper-parameter

for the Dirichlet prior to a small value such as 10^{-3} as discussed in Section 3.3. After convergence of the VEM algorithm we can can use the variational posterior $q(\boldsymbol{\pi})$ in Equation (3.30) for interpretation. For a mixing coefficient π_k the variational posterior mean and variance are:

$$\mathbb{E}[\pi_k] = \frac{p_k}{\sum_{k=1}^{K} p_k}, \quad \text{Var}[\pi_k] = \frac{p_k(\hat{\boldsymbol{p}} - p_k)}{(\hat{\boldsymbol{p}})^2 (\hat{\boldsymbol{p}} + 1)}$$

where $p_k = \alpha_0 + n_k$ and $\hat{\boldsymbol{p}} = \sum_{k=1}^{K} p_k$. When the posterior mean is close to 0 and the variance is very small then we can say that we are almost certain that the component is not contributing to the model and can therefore be eliminated. Also notice that when $n_k = 0$ then the posterior mean is almost 0 and the variance is very small since $a_0 = 10^{-3}$. n_k corresponds to the expected number of observations associated with the k^{th} component and we can see that components which do not "explain" any data are effectively eliminated. Finally from Equation (3.37) we can also see that components with $n_k = 0$ have negligible effect on the predictive density.

The methodology discussed in the previous paragraph is similar to the method proposed by Bishop & Corduneanu (2001) which also requires a single run of the VEM algorithm to determine the number of components. However, in Bishop & Corduneanu (2001) the authors do not impose a prior distribution over mixing coefficient π_k and they treat them as free parameters which are optimised using type II maximum likelihood. The equations presented in Section 3.7.1 remain unchanged with the only difference that there is no variational posterior for $\boldsymbol{\pi}$ while all other variational posteriors are conditioned on $\boldsymbol{\pi}$. Thus the variational lower bound now approximates the

$B_{1,2}$	evidence against M_2
1 - 3	Not worth more than a bare mention
3 - 10	Substantial
10 - 100	Strong
> 100	Decisive

Table 3.4: Interpretation of Bayes factor

marginal conditional on $\boldsymbol{\pi}$, i.e. $p(\boldsymbol{X}|\boldsymbol{\pi})$. The VEM algorithm also remains largely unchanged since the only modification is that in step 8 of Algorithm 7 instead of updating the parameters p_k for the variational posterior over $\boldsymbol{\pi}$, the free parameters π_k are updated using $\pi_k = \frac{1}{N}\sum_{n=1}^{N} r_{n,k}$. At every iteration of the algorithm components with π_k bellow a predefined threshold are eliminated.

3.9 Discussion

In this chapter we have discussed several methods for performing inference over parameters of Gaussian mixture models and developed the methodology for applying a recently proposed family of MCMC algorithms, namely Riemann manifold Langevin and Hamiltonian Monte Carlo, on such models. The experiments in this chapter suggest that Riemann manifold MCMC algorithms can be more efficient than the standard Gibbs sampler and require almost no tuning.

Despite these promising findings however, MCMC algorithms still pose several practical difficulties when they are considered for information retrieval models. Firstly, monitoring convergence to the stationary distribution re-

quires several chains to be run in parallel and measure within and across chains variance to compute convergence statistics (Brooks & Gelman 1998). Secondly, after convergence, several samples from the posterior are required in order to obtain reliable estimates of the necessary expectations such as predictive densities. Therefore their application remains limited to problems with limited data where scalability is not an issue. Moreover, numerical integration should be performed for each new observation for which we need to calculate its density given the observed training data, and therefore their application in situations when such quantities need to be estimated in almost real time is also limited. Informational retrieval systems operate in almost real time when responding to user queries while they also need to scale well with the size of the underlying collection.

In the last sections we also discussed a family of methods which approximate the true posterior and provide analytical solutions when conjugate priors are assumed. The Variational EM algorithm for mixture of Gaussians is very simple to implement and its computational complexity is almost the same as the Expectation Maximisation algorithm for maximum likelihood estimation. Moreover, the number of mixture components can be identified without requiring running the algorithm for several different models. Therefore variational approximations will be considered for the rest of this thesis when inference of mixture model parameters is required.

Chapter 4

Probabilistic Content Based Image Retrieval

Probabilistic models for information retrieval are based on decision and probability theory and thus they provide guidance on how to optimally rank documents with respect to user queries (Robertson & Zaragoza 2009). Despite their successful applications on web and text document retrieval, their application on retrieving multimedia documents such as images and videos with no associated text information or meta-data has not been widely explored until recently. Early content based image retrieval systems where based on similarity and distance functions designed specifically for the underlying image representation and feature extraction method (Smeulders et al. 2000).

Recently, Chum et al. (2008) proposed a methodology to represent images as unordered sets of discrete descriptive salient features which are analogous to terms for text documents and thus indexing and ranking models for in-

formation retrieval can be directly applied. The earlier methods proposed by Vasconcelos (2001), Westerveld et al. (2003) can be seen as a generalisation of the methodology proposed in Chum et al. (2008) where instead they create a representation of images similar to text documents by employing generative probabilistic models to directly model the density of continuous features. This methodology has been shown to be very general and has also been applied to audio retrieval (Turnbull et al. 2008).

In this chapter we firstly show how the models of Chum et al. (2008) and Vasconcelos & Lippman (2000) can be formally derived as special cases of a more general methodology employing the *predictive densities* of image models (Zaragoza et al. 2003). This will allow us to make more clear the relationships and differences of the two approaches. The methodology will also allow us to derive new ranking functions for both approaches which are shown to be theoretically and experimentally superior while maintaining the same computational complexity. Finally, we show how the computational complexity associated with the semi-parametric density models can be reduced by also modelling the query image density and rank images based on the divergence between the predictive densities of collection and query images.

4.1 Probabilistic Image Retrieval

In this thesis, we follow a generative approach, in analogy to the one used in the language modelling framework for Information Retrieval (Ponte & Croft

1998), to derive probabilistic ranking functions for Content Based Image Retrieval (CBIR). In this framework, a generative probabilistic model $p(\boldsymbol{x}|\boldsymbol{\theta}_I)$ with image specific parameters $\boldsymbol{\theta}_I$ is defined for the features $\boldsymbol{x}$ of each image I in an image collection. Inference over parameters is performed by fitting each model using the likelihood $p(I|\boldsymbol{\theta}_I)$ during indexing. Then for a previously unseen query image Q, images in the collection are ranked based on the likelihood that the query image is generated from the corresponding generative process $p(Q|\boldsymbol{\theta}_I)$. The image specific parameters $\boldsymbol{\theta}_I$ can be estimated during the indexing of the collection using a Maximum Likelihood (ML) or Maximum A Posteriori (MAP) procedure

$$\hat{\boldsymbol{\theta}}_I^{(ML)} = \arg\max_{\boldsymbol{\theta}_I} p(I|\boldsymbol{\theta}_I), \quad \hat{\boldsymbol{\theta}}_I^{(MAP)} = \arg\max_{\boldsymbol{\theta}_I} p(I|\boldsymbol{\theta}_I)p(\boldsymbol{\theta}_I)$$

where $p(\boldsymbol{\theta}_I)$ is a prior distribution encoding *a-priori* knowledge about the parameters.

4.1.1 The Multinomial Dirichlet Model

A popular methodology for image retrieval is to create a representation of images that is similar to that of text documents and then apply directly information retrieval ranking models. For example, Chum et al. (2008) extract local SIFT features (Lowe 2004) from a collection of images and quantise them using K-means to form a visual vocabulary. SIFT features from an image are mapped to their closest visual term from the vocabulary and an image is then represented as an unordered set of *visual terms*. In their work,

images in the collection and queries are defined as sparse vectors counting the frequency of visual terms, i.e. $I = \{n_{1,I}, \ldots, n_{T,I}\}$ and $Q = \{n_{1,Q}, \ldots, n_{T,Q}\}$ respectively, where $n_{t,I}$ is the frequency of the t^{th} term in image I. For retrieving images they use a *TF-IDF* weighting scheme.

In this chapter we will cast their model in a probabilistic framework in order to highlight the similarities and differences in the underlying modelling assumptions. Although, the vector space model with *TF-IDF* weighting scheme might initially seem unrelated to the generative model discussed here, there are interesting connections between the two as the term frequency of images is encoded by the likelihood function and the inverse document frequency can be encoded by the prior. These connections are well known for ad-hoc information retrieval, see Zhai & Lafferty (2001) for a discussion.

The distribution of terms in an image under this representation is modelled as a multinomial distribution $\mathcal{M}(x|\boldsymbol{\theta}_I)$ and the ML estimate of the parameters is $\hat{\theta}_{t,I}^{(ML)} = n_{t,I} / \sum_{t'} n_{t',I}$. Due to the sparse nature of this representation the parameters are usually over-fitted which results in several numerical difficulties. For this reason smoothing is employed in the form of a prior distribution over the model parameters. A Maximum A-Posteriori (MAP) estimate can be obtained by assuming a Dirichlet prior distribution over the parameters and results in $\hat{\theta}_{t,I}^{(MAP)} = (n_{t,I} + \alpha_t - 1) / \sum_{t'} (n_{t',I} + \alpha_{t'} - 1)$. The prior hyper-parameters α_t can be set to reflect *a-priori* knowledge before an image I is observed. One approach is to set them to the average term frequencies in the collection, i.e. $\alpha_t = \sum_I n_{t,I} / \sum_{t',I'} n_{t',I'}$ resulting in the *Bayes' Smoothing* estimate, however other options are equally valid, see

(Zaragoza et al. 2003) and (Zhai & Lafferty 2001) for a discussion. The query likelihood for this model is then:

$$p(Q|\hat{\boldsymbol{\theta}}_I^{(MAP)}) = \frac{(\sum_t n_{t,Q})!}{\prod_t n_{t,Q}!} \prod_t \hat{\theta}_{t,I}^{(MAP)\ n_{t,Q}}.$$

The above expression can be further simplified by taking its logarithm; since it is a convex function and it doesn't affect the ranking; splitting terms for which $n_{t,I} = 0$, and by omitting any terms that depend only on Q; since they will be constant for all images in the collection. The final ranking function is then

$$\begin{aligned} \log p(Q|\hat{\boldsymbol{\theta}}_I^{(MAP)}) \quad &\propto \quad \sum_{\{t:n_{t,Q}>0 \wedge n_{t,I}>0\}} n_{t,Q} \log\left(\frac{n_{t,I}}{\alpha_t - 1} + 1\right) \\ &- \quad \log\left(\sum_{t'} n_{t,I} + \alpha_t - 1\right) \sum_{\{t:n_{t,Q}>0\}} n_{t,Q}, \end{aligned} \tag{4.1}$$

which can be efficiently implemented using an inverted index data structure since it relies on terms which are common between the collection image I and the query Q.

4.1.2 The Continuous Mixture of Gaussians Model

The method presented by Westerveld et al. (2003) and Vasconcelos (2001) can be seen as a generalisation of the method of Chum et al. (2008) that avoids quantisation errors by using a semi-parametric model to model directly the density of continuous image features. For each image a finite multivariate

Gaussian mixture model of the form

$$p(\boldsymbol{x}|\boldsymbol{\theta}_I) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

is employed to model the density of Discrete Cosine Transform (DCT) coefficients extracted from a uniform grid over an image. In this setting an image I is represented by an unordered set of vectors in $\mathbb{R}^D$ with D the number of DCT coefficients and the parameters are $\boldsymbol{\theta}_I = \{\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k : k \in \{1, \ldots, K\}\}$. In their study Maximum Likelihood estimates are obtained by using the EM algorithm for finite mixture models, Algorithm 1 introduced in Chapter 3, during indexing of the collection.

In Westerveld et al. (2003) smoothing with a background model is also discussed as an alternative in order to obtain a regularised estimate for the parameters. In analogy with the *Bayes' Smoothing* estimate discussed in the previous section, a MAP estimate can also be obtained by using a conjugate prior, see Section 3.3, with minor modifications to the EM algorithm.

The log of the query likelihood under this model is

$$\log p(Q|\hat{\boldsymbol{\theta}}_I) = \sum_{\boldsymbol{x} \in Q} \log \left(\sum_{k=1}^{K} \hat{\pi}_k \mathcal{N}(\boldsymbol{x}|\hat{\boldsymbol{\mu}}_k, \hat{\boldsymbol{\Sigma}}_k) \right) \tag{4.2}$$

where $\hat{\boldsymbol{\theta}}_I$ can be an ML or MAP estimate obtained by the EM algorithm. Unlike the *Bag-of-Terms* approach the scoring function (4.2) cannot be simplified further.

Moreover, the number of mixture components must be set in advance and is constant for all images in the collection. Westerveld et al. (2003)

conducted experiments with different settings for the number of components on the Corel 5K collection and found that 8 components are optimal for this dataset while results are not very sensitive for values around 8. We argue that this is the case because the number of components is kept constant for all images in the collection. Images with complex structure will require more components to accurately describe their density, while images with less complex structure must be modelled with fewer components in order to avoid over-fitting. If the number of components for each image in the collection is known in advance, setting the constant K to their mean is a good strategy to balance between over and under fitting. However, this is unlikely in real scenarios and this constant has to be optimised with an external procedure.

4.2 Predictive Densities for Image Ranking

The parameters $\boldsymbol{\theta}_I$ in the previous section have to be estimated by fitting the model to the observed data of each image I and taking into account any prior information. Even for very simple models where the maximum can be uniquely identified there will be an associated variance around this estimate quantifying the uncertainty due to model miss-specification, noisy observations and finite data samples. For making predictions, i.e. calculating the likelihood that the query image is generated by the same generative process as an image in the collection, this uncertainty has to be taken into account. The uncertainty for the parameters is encoded in their posterior distribution and it can be marginalised to give the *predictive density*.

The ranking functions using the query likelihood based on ML or MAP point estimates are in fact approximations to a ranking function employing the *predictive densities* of image models. In particular we can write $p(\boldsymbol{x}|I)$ as

$$p(\boldsymbol{x}|I) = \int_{\boldsymbol{\theta}_I} p(\boldsymbol{x}|\boldsymbol{\theta}_I)p(\boldsymbol{\theta}_I|I)d\boldsymbol{\theta}_I, \tag{4.3}$$

where $p(\boldsymbol{\theta}_I|I)$ is the posterior of the model parameters obtained by Bayes' theorem $p(\boldsymbol{\theta}_I|I) = p(I|\boldsymbol{\theta}_I)p(\boldsymbol{\theta}_I)/p(I)$. We can then use $p(Q|I)$, i.e. the *predictive likelihood* of the query conditioned on the image observations alone to rank images in the collection. In cases where the posterior is sharply peaked around some value $\hat{\boldsymbol{\theta}}_I$ then $p(\boldsymbol{x}|I) \approx p(\boldsymbol{x}|\hat{\boldsymbol{\theta}}_I)$ and thus it is equivalent to the ML or MAP ranking functions. However, when the model is miss-specified or data are noisy or scarce, the posterior is broad and the uncertainty is taken into account providing regularised estimates of relevance. Moreover, the ranking functions obtained by the predictive densities in (4.3) are no longer sensitive to parameter estimates as they are not dependent on $\boldsymbol{\theta}_I$. However, they rely on the ability to accurately estimate the integral in (4.3) and the posteriors $p(\boldsymbol{\theta}_I|I)$ for all images in the collection.

Zaragoza et al. (2003) have used the framework of *predictive densities* to derive new ranking functions for ad-hoc retrieval based on unigram language models and showed that a significant improvement over MAP estimates and smoothing models can be achieved. The *Bag-of-Terms* approach discussed in the previous section is equivalent to a unigram language model thus its extension to the *predictive densities* framework is trivial. However, this is not the case for the semi parametric mixture model and thus we will have

to resort to the methods described in Chapter 3 to obtain the *predictive densities*.

4.2.1 Predictive likelihood for the Multinomial Dirichlet model

For the *Bag of Terns* model discussed in the previous section, the posterior and predictive densities can be easily calculated in closed form provided that a Dirichlet prior is specified. In particular, the posterior is also a Dirichlet of the form $\mathcal{D}(\boldsymbol{\theta}_I|\boldsymbol{n}_{\cdot,I}+\boldsymbol{\alpha})$, where $\boldsymbol{n}_{\cdot,I}$ is the vector of term frequencies in image I, and the predictive density for a query image Q is

$$p(Q|I)=\frac{(\sum_t n_{t,Q})!}{\prod_t n_{t,Q}!}\frac{\Gamma\left(\sum_t n_{t,I}+\alpha_t\right)}{\Gamma\left(\sum_t n_{t,Q}+n_{t,I}+\alpha_t\right)}\prod_t\frac{\Gamma\left(n_{t,Q}+n_{t,I}+\alpha_t\right)}{\Gamma\left(n_{t,I}+\alpha_t\right)}. \quad (4.4)$$

Equation (4.4) can be simplified by calculating its log, as it is a convex function and thus it does not affect ranking; removing terms which depend only on $n_{t,Q}$, as they are constant for all images in the collection; and finally using the fact that $\Gamma(n)=(n-1)!$ for all positive integers n to give the following ranking function

$$\begin{aligned}\log p(Q|I) \;\propto\; & \sum_{\{t:n_{t,Q}>0\wedge n_{t,I}>0\}}\sum_{g}^{n_{t,Q}}\log\left(1+\frac{n_{t,I}}{a_t+g-1}\right)\\ & -\sum_{j=1}^{\sum_{t'}n_{t',Q}}\log\left(\sum_{t'}n_{t',I}+a_{t'}+j-1\right). \quad (4.5)\end{aligned}$$

We can see from (4.5) that the function can be efficiently implemented

using a inverted index data structure although its computational complexity scales with the number of terms in the query as opposed to (4.1) which scales with the number of terms common between the query and the image in the collection.

4.2.2 Predictive likelihood for the Gaussian mixture model

Unfortunately the posterior and the predictive density do not have a closed form for mixture models and thus the above methodology cannot be applied directly to the model of Westerveld et al. (2003). However we can use the methods described in Chapter 3 to either obtain a Monte Carlo numerical estimate or analytically approximate the integral in (4.3). We discuss both options here but we experiment only with the latter as it requires less tuning and is less computationally demanding.

We start by imposing a conjugate prior over the model parameters $\boldsymbol{\theta}_I = \{\pi_{k,I}, \boldsymbol{\mu}_{k,I}, \boldsymbol{\Sigma}_{k,I} : k \in \{1, \ldots, K\}\}$ as discussed in Section 3.3. Note the similarity of this prior with the prior used for the Multinomial-Dirichlet model in the sense that it takes into account the collection statistics and the average image in the collection. We can then setup a Markov Chain for each image in the collection in order to draw independent samples from the posterior distributions $p(\boldsymbol{\theta}_I|I)$. For this we can use any of the MCMC algorithms of Chapter 3 but we should stress that more efficient Riemann manifold MCMC algorithms will require fewer samples to converge and also fewer samples will

have to be thrown away in order to get independent samples due to the smaller autocorrelation of the chains.

We denote a sample from the posterior of image I as $S_I = \{\hat{\boldsymbol{\theta}}_{1,I}, \dots \hat{\boldsymbol{\theta}}_{N,I}\}$. The samples can be stored in a database along with their corresponding image identifier in order to be used later for retrieval. The Monte Carlo estimate of the query *predictive likelihood* for image I is then

$$\begin{aligned} \log p(Q|I) &\approx \sum_{\boldsymbol{x}\in Q} \log \frac{1}{N} \sum_{i=1}^{N} p(\boldsymbol{x}|\hat{\boldsymbol{\theta}}_{i,I}) \\ &= \sum_{\boldsymbol{x}\in Q} \log \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K_{i,I}} \hat{\pi}_{k,I}^{(i)} \mathcal{N}(\boldsymbol{x}|\hat{\boldsymbol{\mu}}_{k,I}^{(i)}, \hat{\boldsymbol{\Sigma}}_{k,I}^{(i)}). \end{aligned} \tag{4.6}$$

We can see that all the information about the posterior is contained in the posterior samples S_I. Moreover, the corresponding Monte Carlo error decreases as $1/\sqrt{(N)}$, i.e. to reduce the error by half we have to increase the sample size by 4. For this and due to the sum over all posterior samples, obtaining the query likelihood is a computationally demanding task. Finally notice that the number of components is image dependent and posterior samples do not have to be of the same dimension.

An alternative to Monte Carlo integration is to approximate the posterior distribution $p(\boldsymbol{\theta}_I|I)$ by resorting to the framework of variational inference (Attias 2000) discussed in Section 3.7. For this we have to augment the model with the latent indicator variables $\boldsymbol{Z}_I$ associating image observation to mixture components. The parameters are then $\boldsymbol{\Theta}_I = \{\boldsymbol{\theta}_I, \boldsymbol{Z}_I\}$ and the

augmented likelihood function is

$$p(I|\boldsymbol{\Theta}_I) = \prod_{\boldsymbol{x}\in I}\prod_{k=1}^{K}\left[\pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\right]^{z_{i,k}}.$$

The approximate posteriors obtained by the variational treatment of Gaussian mixture models are

$$q(\boldsymbol{z}_i) = \mathcal{M}(\boldsymbol{z}_i|1, \rho_{i,1}, \ldots, \rho_{i,K}), \qquad q(\boldsymbol{\mu}_k) = \mathcal{N}\left(\boldsymbol{\mu}_k|\boldsymbol{m}_k, (\beta + n_k)^{-1}\boldsymbol{\Sigma}_k\right),$$
$$q(\boldsymbol{\Sigma}_k) = \mathcal{IW}\left(\boldsymbol{\Sigma}_k|\boldsymbol{W}_k, n_k + v_0\right), \qquad q(\boldsymbol{\pi}) = \mathcal{D}(\boldsymbol{\pi}|\boldsymbol{a}_0 + n_k),$$

and $\boldsymbol{\rho}, \boldsymbol{m}, \boldsymbol{W}, n_k$ are the parameters of the approximate posteriors which are optimised by the *variational* EM algorithm, Algorithm 6. These parameters are stored in a database along with the corresponding image identifier in order to be used later for retrieval.

Finally, substituting the above expressions into (4.3) we can analytically evaluate the integral and obtain the predictive density which takes the form of a mixture of Student-t densities. The ranking function for a query image is then

$$\log p(Q|I) = \sum_{x\in Q} \log \frac{1}{\sum_{k=1}^{K} p_k} \sum_{k=1}^{K} p_k \mathrm{St}(\boldsymbol{x}|\boldsymbol{m}_k, \boldsymbol{\Lambda}_k, v_k + 1 - D), \tag{4.7}$$

where $p_k = a_0 + n_k$, $v_k = v_0 + n_k$ and $\boldsymbol{\Lambda}_k = \frac{(v_k+1-D)(\beta+n_k)}{1+\beta+n_k}\boldsymbol{W}_k^{-1}$ and $\mathrm{St}(\cdot)$ is the Student-t density.

Moreover, we can now estimate the number of mixture components K with a single run of the the VEM, Algorithm 7. The method is based on

initially over-estimating the number of components and setting K to a large value. Selecting a Dirichlet prior for the mixing coefficients $\boldsymbol{\pi}$ such that sparse solutions are preferred, i.e. setting α_0 to a value close to zero, the *variational* EM algorithm converges to solutions where many of the components are identical to the prior distribution with $n_k = 0$ and thus they can be removed as they do not affect the result in (4.7). See Section 3.7.1 and Bishop & Corduneanu (2001) for more details.

4.3 Relations with other models

The Gaussian mixture model discussed in the previous sections can be seen as a generalisation of the bag-of-terms approach for content based image retrieval. In fact, the histogram used to represent images in the the bag-of-terms representation is an estimate of the distribution of local feature descriptors. A histogram is known to be a very crude representation of continuous densities however. Thus, it would appear that by improving this density representation to more accurately represent the underlying observation space, we could improve the retrieval effectiveness.

Moreover, quantising a high dimensional space using a finite set of observations often leads to bins, or clusters, which capture large regions of the feature space. During histogram generation the position of the point in each cluster region is not taken into account and thus the points near the boundaries will often contribute to the wrong histogram bins. In order to solve this problem, previous studies (Jegou et al. 2009, Philbin et al. 2008) have consid-

ered allowing one point to contribute to more than one neighbouring clusters based on its position. In the mixture model representation by definition one point contributes to multiple mixture components while the covariance of the component is also taken into account when calculating point membership.

Applying the k-means algorithm to cluster a large number of points, usually in the order of millions, into a large number of clusters, in the order of thousands, is a computationally demanding task. Also, quantisation requires comparing a local feature descriptor with all terms in the vocabulary. In order to reduce the computational complexity associated with k-means and quantisation several authors (Nister & Stewenius 2006, Philbin et al. 2007) have considered creating a vocabulary tree. K-means is initially applied to all local descriptors in the collection to cluster them into a small set of clusters and then the algorithm is applied recursively to further cluster points within these groups. However, this hierarchical quantisation is an approximation to the clustering obtained by the standard K-means algorithm and thus it can further degrade retrieval performance. In contrast by using the mixture model representation the densities for each image can be estimated in parallel.

In the studies of Westerveld et al. (2003) and Vasconcelos & Lippman (2000) no smoothing is considered and a maximum likelihood estimate of the model parameters is used. Smoothing plays an important role in text information retrieval since maximum likelihood estimates over-fit the observed data and assign very small or zero probability to new queries. In Westerveld et al. (2003) the authors argue that smoothing is not required since Gaussian

densities have infinite support. In practice however, and especially in high dimensions the density function for points far from the mean quickly becomes numerically indistinguishable from zero. This can lead to numerical difficulties such as over-flows and under-flows when calculating the query likelihood which are often easily overlooked. The generalisation into the *predictive likelihood* framework provides for two sources of smoothing. One through the use of a prior over the parameters as in the Multinomial-Dirichlet model and a second by marginalising the uncertainty encoded in the posterior.

Another related issue with the maximum likelihood estimation approach in Westerveld et al. (2003), Vasconcelos & Lippman (2000) is model selection. The number of components in these studies is empirically found using cross validation. Moreover the number of components for all image models is set to a single value for the whole collection. This can result in over-fitting images with simple structure where only a few components are necessary but under-fitting images with more complicated structure where more components are needed to represent the variability of features. In the variational EM the number of components is determined automatically for each image model by Bayesian regularisation. Therefore it is expected that the density models will represent better the true density of image features and thus better retrieval performance can be achieved.

Finally, it is important to mention that although the indexing phase of the Bag of Terms representation is computationally expensive, matching and ranking can be efficiently implemented by an inverted index data structure. On the contrary, for the mixture model representation the query likelihood

has to be evaluated for all images in the collection. Additionally, the predictive likelihood has to be evaluated for all local feature descriptors in the query image.

4.4 Modelling the Query Density

In text information retrieval an alternative method for estimating relevance in the language modelling framework is to assume that the query is also generated by an underlying query model. Therefore, the query model also represents the distribution of the query features and thus relevance estimation suffices in calculating the divergence between the query and the document distributions. The assumption of a query model has also theoretical implications since query-document matching is no more inherent in the retrieval model itself (Robertson & Zaragoza 2009). Nevertheless, the query model approach has seen many practical applications in text information retrieval and for the model presented here it can allow for more efficient relevance estimation.

In Section 4.2.2 the scoring function involves calculating the predictive density of each query feature vector for all images in the collection. Since the query is also an image, the number of feature vectors can be in the order of thousands and thus evaluation of the scoring function is computationally demanding. By adopting a query model approach the number of parameters of the query model is smaller than the number of query vectors and for some parametric densities the KL divergence can be computed analytically thus

significantly reducing the computational cost for estimating relevance.

Given a document model and a query model relevance can be estimated by the KL divergence between the two densities. That is

$$\mathrm{KL}\left[p(\boldsymbol{x}|\boldsymbol{Q}), p(\boldsymbol{x}|\boldsymbol{I})\right] = \int p(\boldsymbol{x}|\boldsymbol{Q}) \log \frac{p(\boldsymbol{x}|\boldsymbol{Q})}{p(\boldsymbol{x}|\boldsymbol{I})} d\boldsymbol{x}$$

In the retrieval model presented in this chapter $p(\boldsymbol{x}|\boldsymbol{Q})$ and $p(\boldsymbol{x}|\boldsymbol{I})$ correspond to mixtures of multivariate Student t distributions and the KL divergence does not admit a closed form expression for mixtures in general. Vasconcelos & Lippman (2000) propose an asymptotic approximation for estimating the KL divergence between mixtures of Gaussian densities. The approximation heavily relies on two assumptions. The first assumption is that mixture components of a single model do not overlap. The second assumption is that a component of the query model overlaps with only one component of the image model. Westerveld & de Vries (2003) performed a Monte Carlo study on the TrecVid collection and showed that the first assumption in practice can be satisfied due to the high dimension of the feature space. However, the second assumption is violated in most of the cases and this is also reflected in their experimental results. Moreover, several approximations to the KL divergence for mixture models are compared in Hershey & Olsen (2007).

In this section the use of a kernel function between densities, namely the Probability Product Kernel (PPK) (Jebara et al. 2004), is proposed in order to estimate relevance of an image given a query model. The reason is twofold. Firstly the probability product kernel can be calculated explicitly for mixtures with components in the exponential family of distributions. Therefore

the computational complexity is reduced since no numerical estimates are required. Secondly, the probability product kernel is a valid positive definite kernel function between probability densities (Jebara et al. 2004). This means that PPK computes a generalised inner product in the space of probability densities embedded in the Hilbert space and thus it can be considered as a similarity between two densities. Furthermore, the kernel function can be used in kernel learning machines, such as Support Vector Machines (SVM), for classification, something that is exploited in the next chapter.

The probability product kernel between query and image densities is

$$\mathrm{K}\left[p(\boldsymbol{x}|\boldsymbol{Q}), p(\boldsymbol{x}|\boldsymbol{I})\right] = \int p(\boldsymbol{x}|\boldsymbol{Q})^a p(\boldsymbol{x}|\boldsymbol{I})^a d\boldsymbol{x}.$$

Where a is a parameter typically set to 1 or $\frac{1}{2}$ corresponding to the expected likelihood kernel and the Bhattacharyya kernel respectively. The integral in the above expression can be explicitly calculated when the densities are mixtures with components in the exponential family. However that does not hold for mixtures of multivariate Student t densities and thus a further approximation is proposed.

First note the following result

$$\lim_{v\to\infty} \mathrm{St}(\boldsymbol{x}|\boldsymbol{m}, \boldsymbol{\Lambda}, v) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{m}, \boldsymbol{\Lambda}^{-1}).$$

This means that in the limit of infinite samples the Student t distribution is equal to the Gaussian. An illustration of this result is shown in Figure 4.1. In order to efficiently calculate the PPK for the query and image densities

the Student t components in the predictive densities $p(\boldsymbol{x}|\boldsymbol{Q}), p(\boldsymbol{x}|\boldsymbol{I})$ will be approximated by multivariate Gaussian components. It is important to note that this similar to a MAP estimate procedure.

The approximation can be justified as follows. With the use of a sparsifying prior on mixture weights π_k most of the components will be associated with many observations thus n_k will be large. Moreover, by removing components with $n_k = 0$ since they don't contribute to the predictive density there will be no components with degrees of freedom $v = v_0$. Therefore the Student t components in the query and image predictive densities can be reasonably approximated by Gaussian densities. The accuracy of this approximation will be reflected in the retrieval results and thus it is expected that the query likelihood approach will perform better than retrieval using a query model and PPK with the approximations discussed here. However, the computational complexity associated with evaluating the PPK for mixtures of Gaussians is significantly lower than evaluating the query likelihood directly.

For the rest of this section the approximate predictive densities for a query and an image will be denoted by $\tilde{p}(\boldsymbol{x}|\boldsymbol{Q})$ and $\tilde{p}(\boldsymbol{x}|\boldsymbol{I})$ while the mixture components approximated by Gaussian densities will be denoted by $\tilde{p}_k(\boldsymbol{x}|\boldsymbol{Q})$, $\tilde{p}_{k'}(\boldsymbol{x}|\boldsymbol{I})$. Furthermore the parameters of the predictive densities will be indexed by $\boldsymbol{Q}$ or $\boldsymbol{I}$ to distinguish between parameters for the query and the image densities respectively, eg. $p_{\boldsymbol{Q},k}$ denotes the k^{th} mixture weight of the query's predictive density.

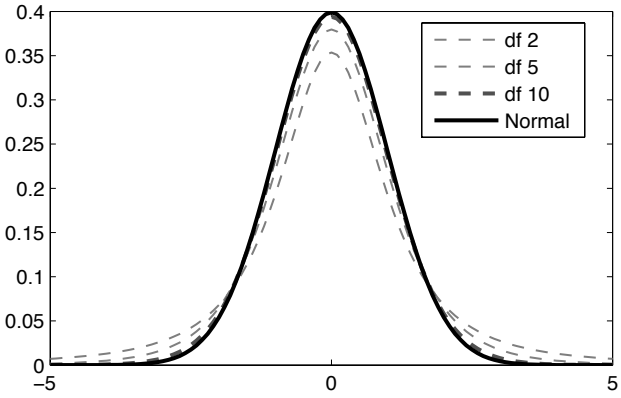

Figure 4.1: Approximating the Student-t density with a Gaussian

The PPK for the approximate predictive densities takes the form

$$\mathrm{K}\left[\tilde{p}(\boldsymbol{x}|\boldsymbol{Q}), \tilde{p}(\boldsymbol{x}|\boldsymbol{I})\right] = \sum_{k=1}^{K_{\boldsymbol{Q}}} \sum_{k'=1}^{K_{\boldsymbol{I}}} \left(p_{\boldsymbol{Q},k} p_{\boldsymbol{I},k'}\right)^a \mathrm{k}\left[\tilde{p}_k(\boldsymbol{x}|\boldsymbol{Q}), \tilde{p}_{k'}(\boldsymbol{x}|\boldsymbol{I})\right], \tag{4.8}$$

that is the sum of PPK kernels between all possible combinations of mixture components. $\mathrm{k}\left[\tilde{p}_k(\boldsymbol{x}|\boldsymbol{Q}), \tilde{p}_{k'}(\boldsymbol{x}|\boldsymbol{I})\right]$ is the PPK for Gaussian components and follows us (Jebara et al. 2004)

$$\begin{aligned}
\mathrm{k}\left[\tilde{p}_k(\boldsymbol{x}|\boldsymbol{Q}), \tilde{p}_{k'}(\boldsymbol{x}|\boldsymbol{I})\right] &= (2\pi)^{(1-2a)D/2} a^{-D/2} \det(\boldsymbol{\Lambda}^\dagger)^{1/2} \det(\boldsymbol{\Lambda}_{\boldsymbol{Q},k})^{a/2} \det(\boldsymbol{\Lambda}_{\boldsymbol{I},k'})^{a/2} \\
&\times \exp\left(\frac{a}{2}\left(\boldsymbol{m}_{\boldsymbol{Q},k}^T \boldsymbol{\Lambda}_{\boldsymbol{Q},k} \boldsymbol{m}_{\boldsymbol{Q},k} + \boldsymbol{m}_{\boldsymbol{I},k'}^T \boldsymbol{\Lambda}_{\boldsymbol{I},k'} \boldsymbol{m}_{\boldsymbol{I},k'}\right)\right) \\
&\times \exp\left(\frac{a}{2}\left(\boldsymbol{m}^{\dagger T} \boldsymbol{\Lambda}^\dagger \boldsymbol{m}^\dagger\right)\right),
\end{aligned}$$

where $\boldsymbol{\Lambda}^\dagger = \left((\boldsymbol{\Lambda}_{\boldsymbol{Q},k} + \boldsymbol{\Lambda}_{\boldsymbol{I},k'}\right)^{-1}$ and $\boldsymbol{m}^\dagger = \boldsymbol{\Lambda}_{\boldsymbol{Q},k}\boldsymbol{m}_{\boldsymbol{Q},k} + \boldsymbol{\Lambda}_{\boldsymbol{I},k'}\boldsymbol{m}_{\boldsymbol{I},k'}$. When $a = 1$ the above expression further simplifies to

$$\mathrm{k}\left[\tilde{p}_k(\boldsymbol{x}|\boldsymbol{Q}), \tilde{p}_{k'}(\boldsymbol{x}|\boldsymbol{I})\right] = \mathcal{N}\left(\boldsymbol{m}_{\boldsymbol{Q},k}|\boldsymbol{m}_{\boldsymbol{I},k'}, \boldsymbol{\Lambda}_{\boldsymbol{Q},k}^{-1} + \boldsymbol{\Lambda}_{\boldsymbol{I},k'}^{-1}\right).$$

4.5 Experiments

In this section a series of experiments designed to evaluate the models presented in this chapter are discussed. The main focus is firstly to compare the ranking functions obtained by the predictive likelihoods and also compare the Bag of Terns representation with the Gaussian mixture models. This will validate the arguments made in this chapter, namely that the predictive densities can better represent the true density of image features than the maximum likelihood solution or the histogram method and that this can result in improvements in retrieval effectiveness. Secondly, the quality of the approximations introduced for obtaining the PPK and their effect on retrieval performance will be analysed.

Table 4.1 shows the names that will be used for the rest of the chapter when referring to particular algorithms and ranking functions along with the parameters used. For the Bag of Terms models, BOT-MAP and BOT-PD, the K-means algorithm was applied with the number of clusters set to T = 2,000. The Dirichlet prior parameters where set to the average term frequency in the collection. The number of components for the EM algorithm used by GMM-PPK and GMM-QL was set to 8. For the variational EM algorithm used by PDG-PPK, PDG-QL and PD-QL the initial number of components was set to 40 and after convergence components with $n_k = 0$ were removed. The prior for the mixture models parameters was set as described in Section 3.3. Note that for PDG-PPK and PDG-QL the parameters are estimated using the variational EM algorithm and the Student-t components are approximated by Gaussians. Finally, the a parameter for the PPK kernels

Method Name	Description	Parameters	Equation
BOT-MAP	Bag of Terms model with a MAP estimate for the query likelihood	T = 2,000	(4.1)
BOT-PD	Predictive likelihood for the Bag of Terms model	T = 2,000	(4.5)
GMM-PPK	PPK with a ML estimate for the mixture model	K = 8	(4.8)
PDG-PPK	PPK where the predictive density is approximated with Gaussian components	K = 40	(4.8)
GMM-QL	Query likelihood with a ML estimate for the mixture model	K = 8	(4.2)
PDG-QL	Predictive likelihood approximated by Gaussian components	K = 40	(4.2)
PD-QL	Predictive likelihood for mixture models	K = 40	(4.7)

Table 4.1: Abbreviations and short description of methods compared in the experiments section. The parameters column gives values for any free parameters while the equation column lists the ranking functions used for each method.

was set to 1 for both GMM-PPK and PDG-PPK.

4.5.1 Evaluation Dataset and Pre-processing

The image dataset that will be used for the evaluation of the proposed model is the Corel 5K collection. The dataset consists of 5,000 high quality, professionally taken images and is divided into 50 high-level semantic categories with 100 images each. The high-level categories correspond to images with a common theme, for example general categories include images from Africa

or images from Greek isles while there are also more specific categories such as images of tigers. Moreover, each image is manually annotated with 1 to 5 tags describing objects and concepts in images. In total there are 260 unique tags in the vocabulary. The dataset is further divided into a set of 4,500 images which can be used as a training collection for optimising any model parameters and a set of 500 images which can be used for testing purposes. The 500 test images are selected uniformly from the set of 50 high level categories and there are 10 images from each category in the test set.

For the retrieval experiments presented in this section only the high level categories will be used. Each high-level category is treated as a different information need and each of the 10 images in the test set from the same high-level category correspond to query instances of the same information need. In all experiments presented here only the 4,500 training images are indexed. The 500 test images are used as user queries and for each image a ranking of the 4,500 image collection is generated.

The collection is pre-processed as follows. A schematic illustration of the process is also shown in Figure 4.2. Images are scaled in 192 pixels width and 128 pixels height. The original RGB colorspace is converted to luminance, blue and red coefficients (LCbCr) by transforming to the YUV colour space. Images are uniformly subdivided into blocks of 8x8 pixels with 4 pixels overlap in both directions resulting in 1,457 subregions. For each subregion the Discrete Cosine Transform (DCT) is calculated for each plane of the LCbCr colour space separately. The DCT coefficients of each plane are vectorised using a zig-zag scanning algorithm, Figure 4.3, and the vectors are

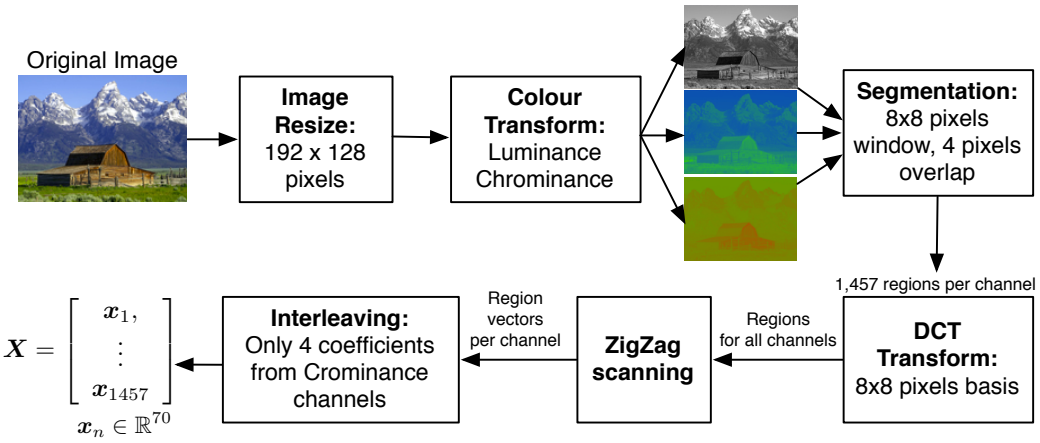

Figure 4.2: Data flow diagram of the collection pre-processing phase. The output of the pre-processing phase is a matrix of 1,457 vectors with dimension 70. See text for more details.

interleaved, Figure 4.4. Exploiting the compression properties of the DCT transform only the the first 3 coefficients from the chrominance channels are interleaved with all the luminance coefficients resulting in vectors of 70 scalar elements. The result of the process is a $70 \times 1,457$ matrix for each image where each column corresponds to one region.

4.6 Results

Table 4.2 summarises the results for the 500 queries in the test set using the standard information retrieval evaluation measures. Average Precision (AP) is the average of precisions computed at the point of each relevant image in a ranking list. Mean Average Precision (MAP) is the mean AP across all queries. R-Prec is the precision calculated at the position of the last relevant image in a ranking list and P@N is the precision calculated at the

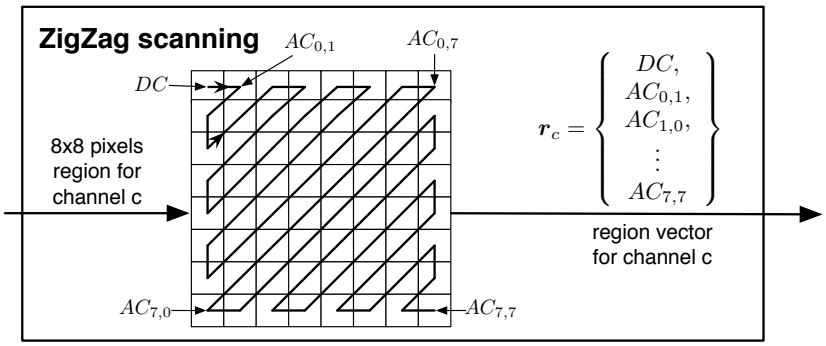

Figure 4.3: Illustration of the Zig-Zag scanning algorithm. DC refers to the first coefficient of the Discrete Cosine Transform with zero frequency on both directions while $AC_{i,j}$ denotes the coefficient of the i^{th} vertical and j^{th} horizontal frequency bands. The algorithm is applied to regions from all colour channels independently.

Nth position of the ranking list. Precision-Recall curves are also shown in Figure 4.5.

Method	**MAP**	**R-Prec.**	**P@5**	**P@10**	**P@20**
BOT-MAP	0.0333	0.0364	0.0441	0.0429	0.0383
BOT-PD	0.0341	0.0375	0.0477	0.0431	0.0387
GMM-PPK	**0.0524***	**0.0674***	**0.1238***	**0.1104***	**0.0994***
PDG-PPK	**0.0534***	**0.0681***	**0.1443***	**0.1242***	**0.1049***
GMM-QL	**0.0975***	**0.1280***	**0.3038***	**0.2599***	**0.2179***
PDG-QL	0.0999	0.1308	0.3070	0.2645	0.2210
PD-QL	**0.1165***	**0.1457***	**0.3315***	**0.2836***	**0.2370***

Table 4.2: Retrieval results for 500 query images in the test set. $*$ indicates statistical significance from the results of the previous row using a Wilcoxon rank-sum test with 1% significance level.

From 4.2 we can see that despite the efficiency and scalability of the *Bag of Terms* representation, quantisation errors can negatively impact retrieval performance. Moreover, the use of the predictive density for the Multinomial-Dirichlet model improves retrieval performance however results are not sta-

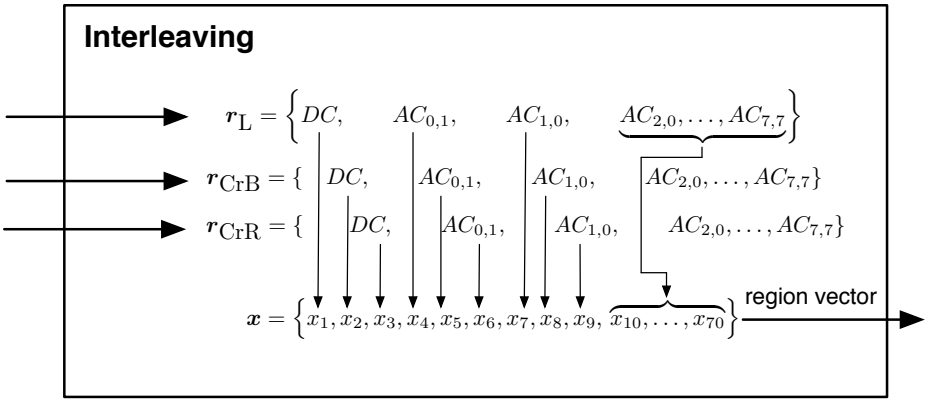

Figure 4.4: Illustration of the interleaving algorithm. Coefficients from the chrominance channels above the first frequency band are discarded. $\boldsymbol{r}_{\mathrm{L}}, \boldsymbol{r}_{\mathrm{CrB}}$ and $\boldsymbol{r}_{\mathrm{CrR}}$ denote the DCT coefficients for the luminance, blue chrominance and red chrominance channels respectively.

tistically significant.

Directly modelling the density of continuous features in images using Gaussian mixture models significantly improves retrieval performance at the cost of the additional computations for calculating the query likelihood for all images in the collection. Moreover, the approximation of the predictive density using Gaussian components instead of Student t densities, discussed above, is not severely degrading performance. It also indicates that on average the variance around the mode of the posterior is not significantly large as the PDG-QL ranking function corresponds to a Maximum A-Posteriori estimate.

Finally, As discussed in the previous sections, both methods using the Probability Product Kernel perform worst than GMM-QL and PDG-QL methods as they rely on estimate of the query image densities. However retrieval performance is still significantly better than the Multinomial-Dirichlet

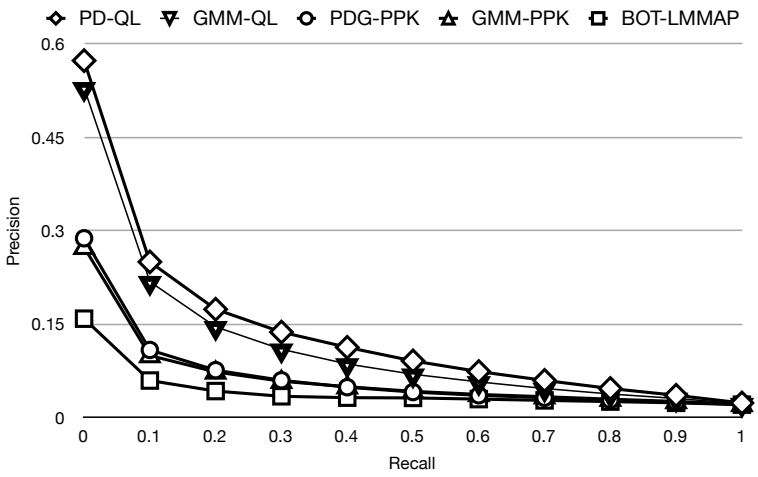

Figure 4.5: Interpolated Precision-Recall curves.

model.

The superior performance of PD-QL method can be attributed to the following two reasons. Firstly, in contrast to a MAP estimate which provides a regularised point estimate, the predictive densities provide a regularised estimate of relevance where the uncertainty associated with model parameters is marginalised. The two approaches will be equivalent if the posterior is sharply peaked around some values, but when the posterior is broad the predictive density averages all possible solutions weighted by their posterior probability. Secondly, the number of mixture components in PD-QL and PDG-QL methods is automatically determined by the output of the *variational* EM algorithm. In contrast, previous approaches (Westerveld et al. 2003, Vasconcelos 2001) set the number of components empirically to a fixed value for all images in the collection which can result in images with more complex densities to be under-fitted while others with more simple densities

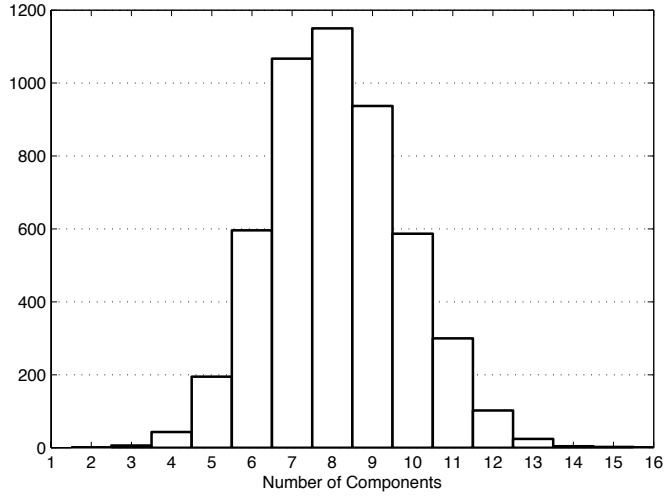

Figure 4.6: Distribution of the number of components K for images in the collection.

to be over-fitted.

In Figure 4.6 the distribution of the number of mixture components K across the Corel 5K collection is shown. The distribution is calculated by the output of the variational EM algorithm for all images. The empirical distribution can be accurately modelled by a Poisson distribution with mean 8. This confirms the findings of (Westerveld et al. 2003) claiming that 8 components is the optimal setting for the Corel 5K collection as most of the images can be modelled with 8 components. However, this is a collection dependent property and we can see how by treating K as a model parameter we can determine the number of components without resorting to cross-validation or external optimisation procedures.

4.6.1 Note on computational complexity

Bag of Terms models require the quantisation of images using a discrete vocabulary of visual terms. In the collection indexing phase the visual term vocabulary is created using the K-means algorithm to cluster feature vectors form all collection images. For the Corel5K collection and the feature extraction method discussed in Section 4.5.1 the total number of feature vectors for the training collection was $M = 6,556,500$ vectors with $D = 70$ scalar values representing the DCT coefficients. These numbers are typical for image retrieval problems even for other types of features such as SIFT Lowe (2004), which have dimension 128 and many features, in the order of thousands, are extracted from each image. Due to the high dimensionality of the data no special data structure such as K-D trees can be used since their performance can be even worst than a linear search (Weber et al. 1998). Thus each iteration of the K-means algorithm scales as $O(M \times T \times D)$.

For the Corel5K collection all feature vectors can be stored in main memory as they are approximately 3.5GB, however for larger collections specialised implementations of K-means that do not require data in main memory are needed. In cases where data can be stored in main memory parallelisation of the K-means algorithm to exploit modern multi-core architectures is relatively simple. However scaling the algorithm to large collections using a computer cluster is not straight forward and specialised programming frameworks such as Map-Reduce (Dean & Ghemawat 2008) need to be employed.

During the retrieval phase a query image has to be quantised using the

visual term vocabulary created during the collection indexing process. This requires to find the closest visual term for each image feature vector using the Euclidean distance. Again the high dimensionality of the data prevents the use of an efficient data structure and thus the process scales as $O(N_Q \times T \times D)$ where N_Q is the number of feature vectors of the query image. For the experiments presented in this chapter $N_Q = 1,457$. Finally, once the query image is quantised finding relevant images is done using the ranking functions in equations (4.1) or (4.5). Both equations depend only on the visual terms common in the query and collection images and can be both implemented using an inverted index data structure exploiting the sparsity of the visual terms representations. Therefore searching scales only with the number of documents matching visual terms in the query which is always much lower than the size of the collection.

For the Gaussian mixture models the EM and VEM algorithms can be applied for each image in the collection independently during the indexing phase. Thus the scaling and memory problems of the K-means algorithm can be avoided and the indexing algorithm can be easily scaled to accommodate large collections. Each iteration of the EM algorithm scales as $O(K \times N_I \times D + K \times D^3)$ where K is the number of components; set to 8 in our experiments, N_I is the number of feature vectors of the collection image; which for the feature extraction method used in this Thesis was 1,457, and $D = 70$ is the dimensionality of the feature vectors. The cubic scaling with respect to D is due to the inversion of the covariance matrices and can be avoided by using diagonal matrices instead. The VEM algorithm has the same complexity as

the EM algorithm with the only difference that we use a larger number of components in order to automatically perform model selection. Note that in a practical application, model selection for the EM algorithm has to be performed as well using a cross validation scheme which would require several runs and thus can be more demanding than the VEM algorithm.

For the GMM-QL, PDG-QL and PD-QL algorithms no additional computations are required for the query image. However scoring documents in the collection requires the calculation of the query likelihood with respect to all images in the collection which scales as $O(C \times N_Q \times K \times D + K \times D^3)$ where $C = 4,500$ is the size of the collection and $N_Q = 1,457$ is the number of feature vectors for the query images. The GMM-PPK and PDG-PPK methods try to remove the scaling factor N_Q by also modelling the query image density. This requires an additional EM or VEM run for the query image which scales better than the quantisation step in Bag of Terms models considering that the vocabulary size $T \gg K$. The complexity of scoring collection images then reduces to $O(C \times K \times (K-1)/2 \times D + K \times D^3)$ where $N_Q \gg (K-1)/2$.

4.7 Discussion

In this chapter we have presented a methodology for deriving retrieval functions for content based image retrieval based on the predictive density of generative models. The method does not make particular assumptions about the representation of documents in the collection but requires the specification

of a probabilistic generative model for the density of the documents' features. The methodology was applied on the Multinomial-Dirichlet model for the Bag of Terms representation and on the finite Gaussian mixture model for continuous features in R^D. In the case of Gaussian mixture models, the parameter estimation and scoring function evaluation remains in the same order of complexity with the corresponding maximum likelihood procedures while it significantly improves performance. On the other hand, for the Multinomial-Dirichlet model the predictive likelihood requires more computations than the MAP estimates although it can still implemented efficiently using an inverted index data structure.

Despite the superior retrieval performance of the mixture models compared to the Bag of Terns representation, scalability to large scale collections remains an important issue since the predictive densities for all images in the collection have to be evaluated for each query. Designing efficient indexing data structures such as the inverted index is not trivial for models such as mixtures of Gaussians.

Finally, both models used in this study assume that local image feature descriptors are conditionally independent i.e. $p(I|\boldsymbol{\theta}) = \prod_{x \in I} p(\boldsymbol{x}|\boldsymbol{\theta})$. This is an over simplifying assumption necessary to obtain tractable models and is similar to the term independence assumption in text document retrieval. It has been shown that by exploiting correlations between local descriptors retrieval performance can be improved (Philbin et al. 2007). However, for the Bag of Terms model such procedures are applied in a post processing step (Lowe 2004) and are difficult to be casted in a probabilistic framework. Fer-

gus et al. (2005) have studied a generative probabilistic model that takes into account the spatial geometry of local feature descriptors for image classification although it also relies on a Bag of Terms representation. Generalisations of this model such as those presented in the chapter is an interesting future direction.

Chapter 5

Semantic Image Retrieval

Multimedia content, such as images, video and audio, does not lend itself to traditional indexing methods like those applied to text documents. The main difficulty arises from the nature of the information encoded in these types of media. While the basic structural units of a text document are words, which directly convey the message of the document in a human understandable form, there is no corresponding analogue to words in media such as images and audio.

Content Based Image Retrieval (CBIR) techniques attempt to partially solve the problem by providing the means for comparing images. However, their application is limited mainly due to the semantic gap (Smeulders et al. 2000), i.e. the lack of correspondence between the image representation used by CBIR techniques and the semantic representation that users construct from images.

A solution towards bridging this gap between visual similarity, as expressed by similarity of low-level image features, and semantic similarity, is to directly associate to images semantic features such as keywords. Indeed, this is the approach followed by many commercial image archives such as GettyImages[1], Corbis[2] etc. Moreover, users and communities are also interested in indexing their personal collections using semantic features in order to render them accessible by others. This is evident from the success of image sharing web applications such as Flickr[3], Picasa[4] and others.

A significant amount of research has therefore been focused on developing algorithms for automatically annotating images based on classification models treating keywords as classes. Similar to content based image retrieval methods, initial approaches were based on the *Bag of Terms* representation and proposed simple Naive Bayes' and Support Vector Machine classifiers (Csurka et al. 2004). Generalisations of these approaches to continuous image features by modelling the class conditional densities using kernel density estimation (Lavrenko et al. 2003, Yavlinsky et al. 2005) and semi-parametric Gaussian mixture models (Carneiro & Vasconcelos 2005) have been also considered. In particular the algorithm of Carneiro et al. (2007) has achieved one of the best so far accuracies on the Corel 5K collection.

In this chapter we approach the problem using the principles presented in the previous chapter and propose a generalisation of the algorithm of Carneiro et al. (2007). Namely, instead of maximum likelihood estimates for

[1] www.gettyimages.com
[2] pro.corbis.com
[3] www.flickr.com
[4] picasaweb.google.com

the class conditional densities, we derive a variational approximation for *hierarchical* mixture models which can be used to obtain the *class conditional predictive densities* by marginalising the parameters using their posterior densities. Thus the algorithm of Carneiro et al. (2007) can be seen as an approximation to the algorithm presented here. In contrast to the content based image retrieval problem, where the variational EM algorithm can be directly applied to obtain approximations to the parameter posteriors, its application to *hierarchies* of mixture models is not straightforward. We therefore derive an novel variational EM algorithm for *hierarchies of mixture models*.

5.1 Generative Classifiers based on Mixture Models

For semantic image retrieval we want to rank images in the collection in response to a keyword user query based on the probability the concepts described by these keywords are present in the image. We are therefore interested in estimating $p(w_c|I)$ where w_c is a keyword from a fixed and pre-defined vocabulary and I is an image in the collection. In a generative classification scheme this quantity is inverted by Bayes' rule in order to obtain

$$p(w_c|I) = \frac{p(I|w_c)p(w_c)}{\sum_{w_i} p(I|w_i)p(w_i)},$$

where $p(I|w_c)$ is the class conditional density and $p(w_c)$ is the prior for keyword w_c. The prior can be set to be uniform across all keywords in the

vocabulary corresponding to the belief that all keywords are *a-priori* equally likely. Alternatively, assuming that the training collection of images is a representative sample $p(w_c)$ can be set to the probability of term w_c in the training collection in order to favour more frequently occurring keywords (Yavlinsky et al. 2005, Lavrenko et al. 2003, Carneiro & Vasconcelos 2005).

For the class conditional densities a parametric model $p(I|\boldsymbol{\theta}_c)$ with keyword dependent parameters $\boldsymbol{\theta}_c$ is employed and by assuming independence between image features it can be written as $\prod_{\boldsymbol{x}\in I} p(\boldsymbol{x}|\boldsymbol{\theta}_c)$. The parameters $\boldsymbol{\theta}_c$ have to be estimated by fitting the model to a collection of training images $I_c \in T_c$ depicting the concept corresponding to keyword w_c while the exact nature of the model depends on the image representation. Similar to the discussion in Chapter 4, the model parameters can be estimated using a Maximum Likelihood (ML) or a Maximum A Posteriori (MAP) procedure or they can be completely marginalised in order to obtain the *predictive class conditional density*.

5.1.1 Class conditional density models

Estimating the parameters for the class conditional density models requires a labeled training set of images for each class. Gathering such a training set for a large set of keywords is not trivial. While several approaches exist, they impose different constraints on the model assumptions that need to be made for the class conditional densities. For example, in the LabelMe collection (Russell et al. 2008) images are manually segmented and each region is annotated with a keyword. This collection imposes no restrictions

for the class conditional densities since each region depicts only the concept corresponding to the class of interest. However, obtaining such a collection requires extensive human supervision which is very expensive.

On the other hand, the CalTech256 (Griffin et al. 2007) is collected by querying a web search engine and then manually filtering images in order to ensure that the concept of interest occupies at least 50% of the image as well as to remove any false positives. Each image in this collection is associated with a single keyword but the model for the class conditional densities must take into account the fact that images for a particular class will also contain several local image descriptors from the background.

Finally, the most challenging case, from a modelling point of view, is that of the Corel5K collection (Duygulu et al. 2002) where each image is manually annotated with several keywords, depending on the concepts depicted in the image. In this collection there is no correspondence between keywords and regions in the image and thus it is relatively easier to manually annotate such a collection. These three scenarios are depicted in Figure 5.1.

The main question that arises is if we can estimate the class conditional densities for weakly labeled training collections such as the Corel5K. We argue that this is not only possible but it can also be beneficial to do so. The first part of the argument relies on *Multiple Instance* learning. In contrast to the standard supervised learning paradigm where each training example is labeled positive or negative with respect to a class, in Multiple Instance learning, training examples are bags of instances, similar to the bag of feature vectors or visual terms representation. A training example is considered to

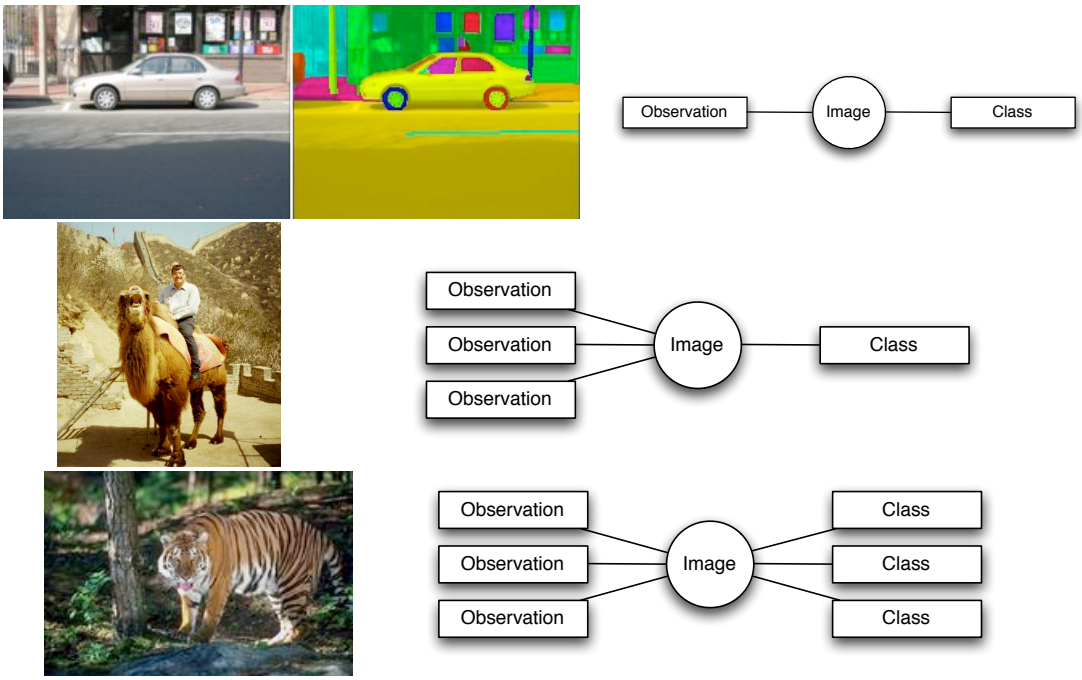

Figure 5.1: Comparison between different annotation schemes. **Top**: supervised. Each coloured region in the segmentation mask corresponds to a different class. In this case Image refers to a region based on the segmentation mask. **Middle**: Multiple-instance. One class is associated with each image. Not all regions in the image correspond to its class. **Bottom**: Multiple-instance multiple-label. Multiple classes associated with each image and there is no correspondence between classes and regions in the image.

belong to one class if and only if one of its instances is positive w.r.t that class, otherwise the training example is considered negative (Dietterich et al. 1997). The goal is therefore to estimate the density of positive instances for a collection of positive and negative training examples.

In (Dietterich et al. 1997) the density of positive instances is assumed to be uniform in an axis-aligned hyper-rectangle of the instance feature space, while in (Maron & Lozano-Pérez 1998) the axis-aligned rectangle assumption is relaxed by assuming a normal distribution with an unknown mean param-

eter. The common assumption that multiple instance algorithms share is that although the density of positive instances is not dominant within each training example, it becomes dominant when all training examples are considered since the distribution of negative instances tends to be uniform. This is better illustrated by the following simulation experiment.

5.1.2 Simulation

For the purpose of illustration we will use one dimensional instances for this simulation. We consider a training example, corresponding to an image, to be a random sample from the densities of four concepts with one of them being the concept of interest and the rest to be randomly selected without replacement from a pool of 200 concepts. All concept densities are mixtures of two univariate normals. The density of positive instances is set to be a mixture of two univariate Gaussian components with known means and standard deviations:

$$p(x|w_1) = 0.3 \times \mathcal{N}(x| - 20, 3) + 0.7 \times \mathcal{N}(x|20, 2).$$

The pool of the 200 concept densities are generated by uniformly sampling their means, standard deviations and mixing coefficients from $\mu \in [-100, 100]$, $\sigma \in [0.1, 10]$ and $\pi \in [0, 1]$ respectively.

For generating a training example we firstly simulate 100 instances from the positive density. Then 3 concept densities are randomly selected from the pool of 200 without replacement and simulate 100 points from each.

The total number of instances per training example is thus 400 with all concepts having equal probability. Finally 1,000 training examples are generated in this manner. Figure 5.2 depicts the empirical distribution of 2 training examples as well as the empirical and estimated densities for all training examples. The density estimate denoted as EM in Figure 5.2 is obtained using the EM algorithm for fitting a Gaussian mixture of 9 components to all instances from the 1,000 training examples while the density estimate denoted as HEM is obtained by using the *Hierarchical EM* algorithm discussed in the next section.

From Figure 5.2 we can see that although the concept density is not dominant in the two randomly selected training examples, it is easily identified when all training examples are considered validating the argument that we can actually estimate the class conditional densities from weakly labeled collections. In fact the distribution of all instances can be approximated by a linear combination of the target concept density and a uniform distribution.

5.1.3 Class independence

In the beginning of this chapter and in the simulation experiment described above we naively assumed that classes, or keywords, are independent. However this is a simplifying assumption as in reality several concepts will present complex interdependences manifested by keyword co-occurrence in the training examples (Nowak et al. 2010, Llorente & Rüger 2009). Moreover, if two concepts frequently co-occur in the training examples then identifying their true densities, in the same way as we did in the simulation experiment, will

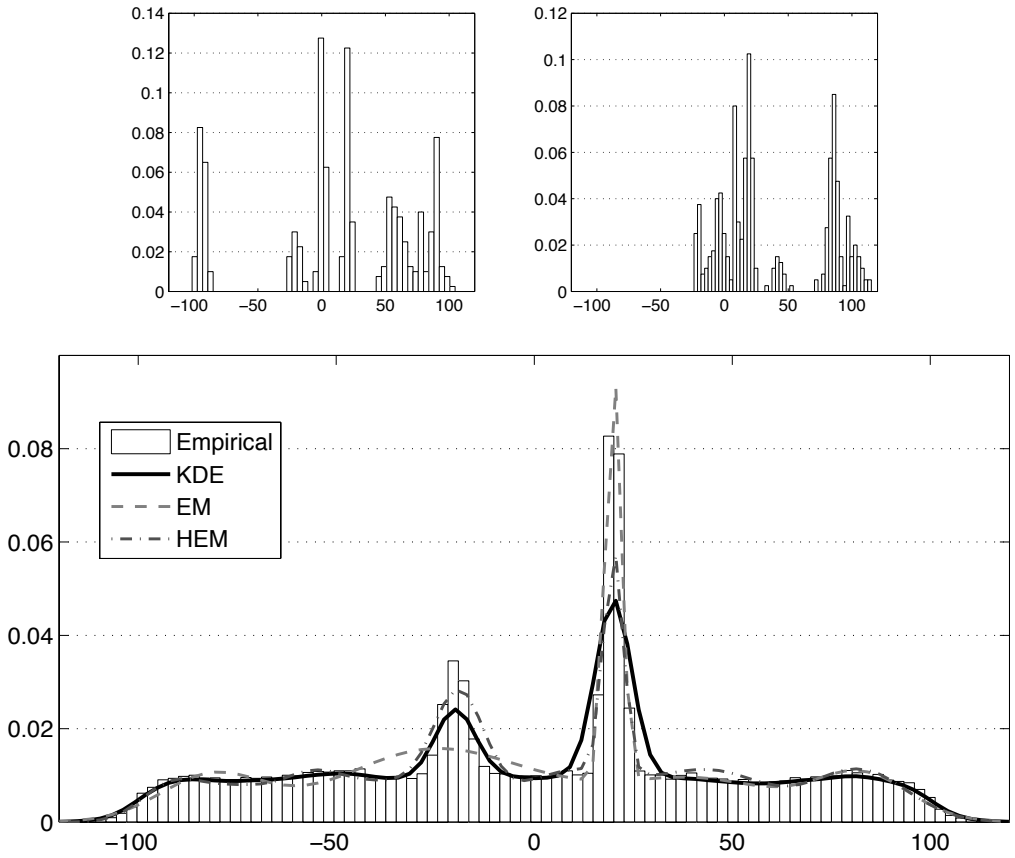

Figure 5.2: Results from the simulation. **Top row**: empirical distribution of two training examples with 400 instances. **Bottom row**: Empirical distribution of all training instances and density estimates using the Kernel Density estimation, EM and Hierarchical EM algorithms.

be more difficult. However, this is not actually a problem. In fact modelling the class conditional densities in such a way implicitly relaxes the class independence assumption as class interactions will be captured by the class conditional densities.

To better illustrate this we repeat the previous simulation experiment.

This time we introduce a second concept density:

$$p(x|w_2) = 0.6 \times \mathcal{N}(x| - 10, 1) + 0.4 \times \mathcal{N}(x|10, 2),$$

and generate synthetic data such that the two concepts co-occur in 40% of the training examples. Figure 5.3 shows the empirical distribution of all instances for concept w_1. We can see that it is a mixture of concept densities corresponding to classes w_1 and w_2 and a uniform component. Although it is not possible to distinguish the true density of w_1 from that of w_2 we can see how an unseen bag of instances generated only by the true density of w_2 will have also high likelihood under the estimated class conditional density for class w_1.

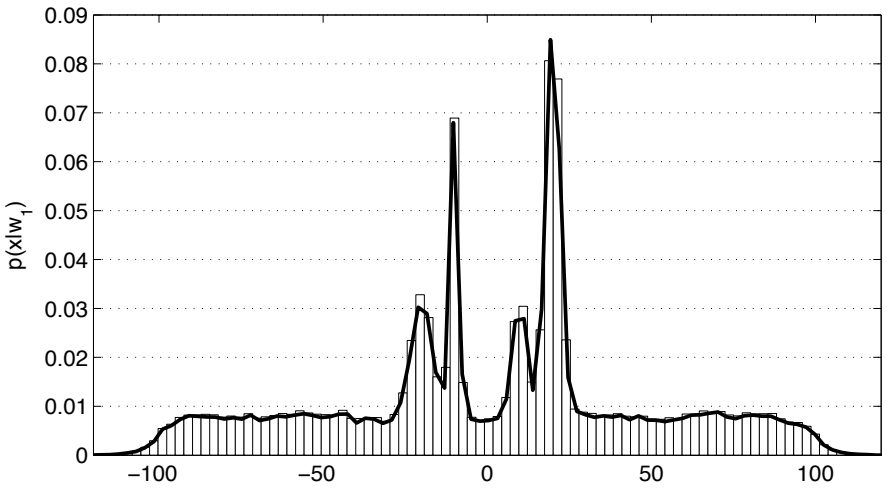

Figure 5.3: Results from the simulation experiment with two concepts co-occurring in 40% of the training examples.

5.1.4 Direct Estimation

Following the discussion in the previous sections, we can directly estimate the class conditional densities by utilising the training examples for the corresponding classes. The model for the class conditional densities however depends on the image representation.

For example in Csurka et al. (2004), for the Bag of Terms representation a Multinomial $\mathcal{M}(\boldsymbol{x}|\boldsymbol{\theta}_c)$ is used to model the distribution of visual terms for each keyword. The class dependent parameters $\boldsymbol{\theta}_c$ in their study are estimated using a MAP procedure with a Laplace smoothing prior resulting in

$$\hat{\theta}_{t,c}^{(MAP)} = \frac{1 + \sum_{I_c \in T_c} n_{t,I_c}}{\sum_{c'}(1 + \sum_{I_c \in T_{c'}} n_{t,I_c})},$$

where n_{t,I_c} is the frequency of the t^{th} visual term in training image I_c. This method can be easily generalised to the *predictive densities* framework by following the same reasoning as in Section 4.2. By assuming a Dirichelt prior over the parameters $\mathcal{D}(\boldsymbol{\theta}_c|\boldsymbol{a})$ the posterior is also a Dirichelt, $\mathcal{D}(\boldsymbol{\theta}_c|\boldsymbol{n}_{\cdot,c} + \boldsymbol{\alpha})$, where $\boldsymbol{n}_{\cdot,c}$ is the vector of visual term frequencies for the images in the training collection of keyword w_c, i.e. $n_{t,c} = \sum_{I_c \in T_c} n_{t,I_c}$. The posterior can then analytically marginalised in order to give the *class conditional predictive distribution*

$$\begin{aligned} p(I|w_c) &= \int p(I|\boldsymbol{\theta}_c) p(\boldsymbol{\theta}_c|w_c) d\boldsymbol{\theta}_c \\ &= \frac{(\sum_t n_{t,I})!}{\prod_t n_{t,I}!} \frac{\Gamma\left(\sum_t n_{t,c} + \alpha_t\right)}{\Gamma\left(\sum_t n_{t,c} + n_{t,c} + \alpha_t\right)} \prod_t \frac{\Gamma\left(n_{t,I} + n_{t,c} + \alpha_t\right)}{\Gamma\left(n_{t,c} + \alpha_t\right)} \end{aligned} \quad (5.1)$$

For modelling directly the density of continuous image features and avoid the quantisation step for the Bag of Terms representation we can again adopt a Gaussian mixture model thus the class conditional density becomes

$$p(I|\boldsymbol{\theta}_c) \;=\; \prod_{\boldsymbol{x}\in I}\sum_{k=1}^{K}\pi_{k,c}\mathcal{N}\left(\boldsymbol{x}|\boldsymbol{\mu}_{k,c},\boldsymbol{\Sigma}_{k,c}\right). \tag{5.2}$$

Maximum Likelihood estimates for the model parameters can be obtained by the EM algorithm or, as we did in Section 4.1, the *predictive class conditional density* for this model can be obtained by Monte Carlo integration or by the variational EM algorithm applied on the local feature descriptors form the training collection of class w_c. However, there are two main technical difficulties in directly modelling the class conditional densities as mixtures of Gaussians.

The first problem is computational. The number of local feature descriptors for one class can be very high since an image can have thousands or tens of thousands of local features depending on its resolution, interest point detector (Mikolajczyk, Leibe & Schiele 2005) or sampling scheme. Even with nowadays computers, it is not possible to keep all local descriptors in main memory and implementing an EM algorithm for such data is a difficult engineering task. Ideally we would like to be able to estimate the class conditional densities from the individual densities of each training image in the collection. The second problem is with the nature of the EM algorithm itself. Due to its iterative nature the EM algorithm is not guaranteed to converge to a global maximum and it can easily trapped in local modes as any deterministic optimisation algorithm.

Vasconcelos & Lippman (1998) proposed a generalisation of the EM algorithm that attempts to solve both problems and in Carneiro et al. (2007) the algorithm was applied for annotating images from the Corel5K collection achieving state of the art accuracy. In the next section we describe the algorithm of Vasconcelos & Lippman (1998) as it will be the basis for our generalisation in a Bayesian framework. In Section 5.1.6 we derive the *hierarchical variational EM* algorithm using the same principles of Vasconcelos & Lippman (1998). Both algorithms are then compared on the Corel5K collection in Section 5.4.

5.1.5 Mixture Hierarchies

Suppose we have a training collection of images $I_c \in T_c$ for keyword w_c. Each image is an unordered set of vectors $\boldsymbol{x} \in \mathbb{R}^D$ and we are interested in modelling the density of vectors from all training images using a finite Gaussian mixture model. Directly maximising the likelihood function

$$\prod_{I_c \in T_c} \prod_{\boldsymbol{x}_{I_c} \in I_c} p(\boldsymbol{x}_{I_c}|\boldsymbol{\theta}_c) = \prod_{I_c \in T_c} \prod_{\boldsymbol{x}_{I_c} \in I_c} \sum_{k=1}^{K_c} \pi_{k,c} \mathcal{N}(\boldsymbol{x}_{I_c}|\boldsymbol{\mu}_{k,c}, \boldsymbol{\Sigma}_{k,c}),$$

with the EM algorithm is problematic since all vectors must be processed at each iteration. The core idea behind the algorithm in Vasconcelos & Lippman (1998) is that we can estimate the parameters $\boldsymbol{\theta}_c$ in a bottom up approach.

Firstly, the EM algorithm, Algorithm 1, is applied to estimate the indi-

vidual densities for each training image by maximising the likelihoods

$$\prod_{\boldsymbol{x}_{I_c} \in I_c} \sum_{i=1}^{K_{I_c}} \pi_{i,I_c} \mathcal{N}(\boldsymbol{x}_{I_c} | \boldsymbol{\mu}_{i,I_c}, \boldsymbol{\Sigma}_{i,I_c}),$$

and obtain ML estimates for the image dependent parameters

$$\boldsymbol{\theta}_{I_c} = \{\pi_{i,I_c}, \boldsymbol{\mu}_{i,I_c}, \boldsymbol{\Sigma}_{i,I_c} : i \in \{1, \ldots, K_{I_c}\}\}.$$

This process can be easily performed in parallel since the EM algorithm is independent for each image I_c. Notice also that the number of components for the individual image densities is different from that of the class conditional density. Then the hierarchical EM algorithm of Vasconcelos & Lippman (1998) given in Algorithm 8 can be applied to obtain ML estimates for the class dependent parameters $\boldsymbol{\theta}_c = \{\pi_{k,c}, \boldsymbol{\mu}_{k,c}, \boldsymbol{\Sigma}_{k,c} : k \in \{1, \ldots, K_c\}\}$ using the parameters of the individual image densities. The process is schematically illustrated in Figure 5.4.

In Algorithm 8 only the model parameters for the individual image densities are used and thus the algorithm can be implemented efficiently since the number of parameters for each image density model is significantly lower than the number of its local feature descriptors. Moreover, the Expectation step is similar to that of a *deterministic annealing* EM algorithm (Ueda & Nakano 1998). The term $M\pi_{i,I_c}$ in Algorithm 8 can be seen as an inverse temperature parameter. The higher the temperature the smoother the estimate of the expectation which results in associating one observation to many mixture components and thus allowing the algorithm to escape local maxima.

Algorithm 8: HEM-GMM

Inititialise $\boldsymbol{\theta}_c$

repeat

{E-Step}

$$\mathbb{E}\left[z_{i,I_c,k}\right] = \frac{\pi_{k,c}\left[\mathcal{N}(\boldsymbol{\mu}_{i,I_c}|\boldsymbol{\mu}_{k,c},\boldsymbol{\Sigma}_{k,c})\exp\left(-\frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Sigma}_{k,c}^{-1}\boldsymbol{\Sigma}_{i,I_c}\right)\right)\right]^{M\pi_{i,I_c}}}{\sum_{j=1}^{K_c}\pi_{j,c}\left[\mathcal{N}(\boldsymbol{\mu}_{i,I_c}|\boldsymbol{\mu}_{j,c},\boldsymbol{\Sigma}_{j,c})\exp\left(-\frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Sigma}_{j,c}^{-1}\boldsymbol{\Sigma}_{i,I_c}\right)\right)\right]^{M\pi_{i,I_c}}}$$

{M-Step}

$$\pi_{k,c} = \frac{1}{\sum_{I_c\in T_c} K_{I_c}}\sum_{I_c\in T_c}\sum_{i=1}^{K_{I_c}}\mathbb{E}\left[z_{i,I_c,k}\right]$$

$$\boldsymbol{\mu}_{k,c} = \frac{\sum_{I_c\in T_c}\sum_{i=1}^{K_{I_c}}\mathbb{E}\left[z_{i,I_c,k}\right]M\pi_{i,I_c}\boldsymbol{\mu}_{i,I_c}}{\sum_{I_c\in T_c}\sum_{i=1}^{K_{I_c}}\mathbb{E}\left[z_{i,I_c,k}\right]M\pi_{i,I_c}}$$

$$\boldsymbol{\Sigma}_{k,c} = \frac{\sum_{I_c\in T_c}\sum_{i=1}^{K_{I_c}}\mathbb{E}\left[z_{i,I_c,k}\right]M\pi_{i,I_c}\left(\boldsymbol{\Sigma}_{i,I_c} + \left(\boldsymbol{\mu}_{i,I_c} - \boldsymbol{\mu}_{k,c}\right)\left(\boldsymbol{\mu}_{i,I_c} - \boldsymbol{\mu}_{k,c}\right)^T\right)}{\sum_{I_c\in T_c}\sum_{i=1}^{K_{I_c}}\mathbb{E}\left[z_{i,I_c,k}\right]M\pi_{i,I_c}}$$

until convergence

Finally, as the inverse temperature is proportional to the mixing coefficient π_{i,I_c}, components from the image densities that don't have a significant mass are more likely to be merged together by the hierarchical EM algorithm. These properties of the hierarchical EM algorithm are also evident in the simulation results in Section 5.1.2. In Figure 5.2 the hierarchical EM provides a better density estimate despite the fact that we have initialised the

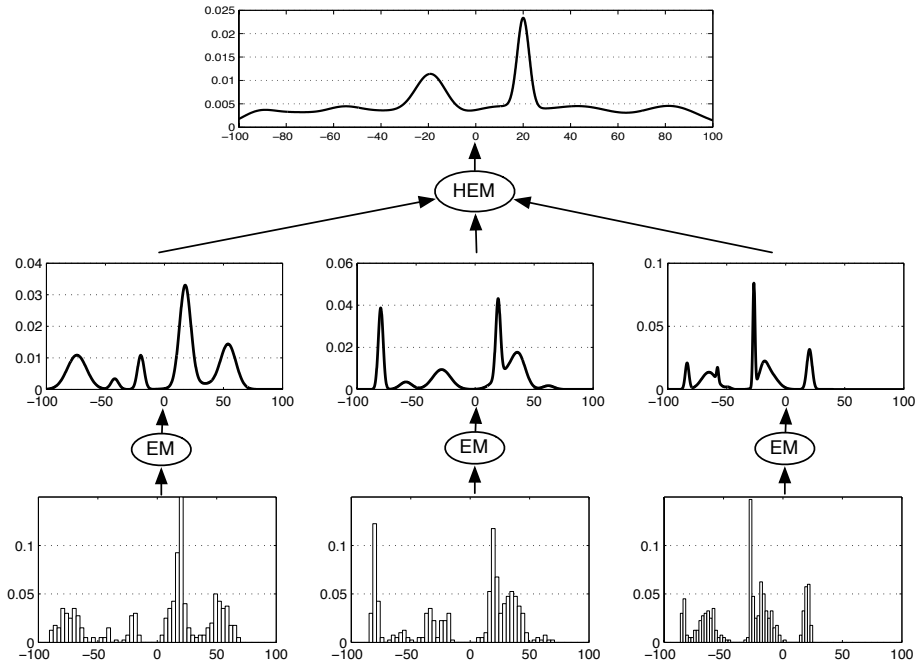

Figure 5.4: Schematic illustration of the hierarchical EM algorithm. Results are obtained by the simulation experiment in Section 5.1.2. The EM algorithm is applied to all training examples individually, estimated parameters are then used by the hierarchical EM algorithm to estimate the class conditional density.

standard EM several times and kept the best solution.

5.1.6 Bayesian Mixture Hierarchies

For generalising the *hierarchical EM* algorithm in a Bayesian framework we will follow the idea of Vasconcelos (2000) and use a *virtual sample*, i.e. a random sample from the densities of all images. As discussed in Chapter 4 we can estimate the predictive densities for individual images using the variational EM algorithm. The predictive densities have an analytic parametric

form that is a mixture of Student-t components with form

$$p(\boldsymbol{x}|I_c) = \sum_{i=1}^{K_{I_c}} p_i \mathrm{St}(\boldsymbol{x}|\boldsymbol{m}_{i,I_c}, \boldsymbol{\Lambda}_{k_{I_c}}, v_{i,I_c}), \quad \sum_{i=1}^{K_{I_c}} p_i = 1.$$

The *virtual sample* is constructed by drawing a random sample of size M from each image density. Note that each image density is a mixture of K_{I_c} Student-t components and thus the *virtual sample* will contain samples from all the K_{I_c} components each with size Mp_i. Using C to denote the number of total mixture components form all image densities, i.e. $C = \sum_{I_c \in T_c} K_{I_c}$, and using the subscript $j \in \{1, \ldots, C\}$ to index them, we can see that the *virtual sample* $\boldsymbol{V}$ can be partitioned into C subsets $\boldsymbol{X}_j \subseteq \boldsymbol{V}$ where each subset is a random sample from the j^{th} component, $\boldsymbol{x}_{j,n} \sim \mathrm{St}(\boldsymbol{x}|\boldsymbol{m}_j, \boldsymbol{\Lambda}_j, v_j)$, with size $N_j = Mp_j$. The likelihood function of the class conditional density for the *virtual sample* is then

$$p(\boldsymbol{V}|\boldsymbol{Z}, \boldsymbol{\theta}_c) = \prod_{j=1}^{C} \prod_{k=1}^{K_c} \left[\pi_{k,c} \prod_{n=1}^{N_j} \mathcal{N}(\boldsymbol{x}_{j,n}|\boldsymbol{\mu}_{k,c}, \boldsymbol{\Sigma}_{k,c}) \right]^{z_{j,k}}, \tag{5.3}$$

where we have introduced the latent indicator variables $\boldsymbol{Z}$ associating subsets $\boldsymbol{X}_j$ of the virtual sample to mixture components. This decision is made in order to prevent the hierarchical EM from splitting sample subsets into different components. This also enforces a hierarchical structure since a latent variable $z_{j,k}$ constraints all samples from a subset $\boldsymbol{X}_j$ to be associated with a single component in the mixture model for the class conditional mixture model. This is also illustrated by the diagram in Figure 5.5.

We can now introduce the following lemma:

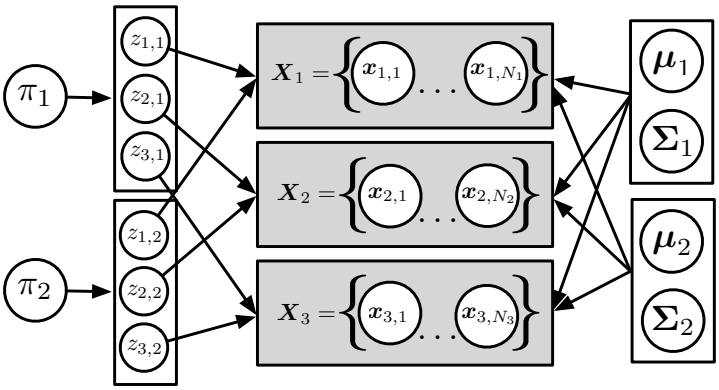

Figure 5.5: Schematic illustration of the dependencies between variables for the hierarchical mixture model using a *virtual sample*. The example considers 3 components with corresponding *virtual sample* blocks $\boldsymbol{X}_1, \boldsymbol{X}_2$ and $\boldsymbol{X}_3$ and 2 mixture components for the class conditional mixture model. Notice that blocks can be due to different images, for example $\boldsymbol{X}_1, \boldsymbol{X}_2$ can be a virtual sample from a 2 component mixture for one image while $\boldsymbol{X}_3$ can be a virtual sample from a single component mixture for a second image.

Lemma 5.1.1 *If* $\mathbf{x_n}$, $n \in \{1, \ldots, N\}$, *are an i.i.d. sample from an unknown distribution with sample mean* $\bar{\boldsymbol{x}} = \frac{1}{N}\sum_n \boldsymbol{x}_n$ *and covariance* $Cov(\boldsymbol{x}) = \frac{1}{N}\sum_n (\boldsymbol{x}_n - \bar{\boldsymbol{x}})(\boldsymbol{x}_n - \bar{\boldsymbol{x}})^T$ *then*

$$\prod_{n=1}^{N} \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \left[\mathcal{N}(\bar{\boldsymbol{x}}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \exp\left(-\frac{1}{2} tr\left(Cov(\boldsymbol{x})\boldsymbol{\Sigma}^{-1}\right)\right)\right]^N.$$

Proof

$$\prod_{n=1}^{N} \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{-\frac{DN}{2}} \det(\boldsymbol{\Sigma})^{-\frac{N}{2}} \exp\left(-\frac{1}{2}\sum_{n=1}^{N}(\boldsymbol{x}_n - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu})\right) \quad (5.4)$$

Introducing $\bar{\boldsymbol{x}} - \bar{\boldsymbol{x}}$ in the exponent and multiplying by $\frac{N}{N}$ we get

$$-\frac{N}{2}\frac{1}{N}\sum_{n=1}^{N}(\boldsymbol{x}_n - \boldsymbol{\mu} + \bar{\boldsymbol{x}} - \bar{\boldsymbol{x}})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_n - \boldsymbol{\mu} + \bar{\boldsymbol{x}} - \bar{\boldsymbol{x}}).$$

Expanding and rearranging terms gives

$$\begin{aligned} -\frac{N}{2}\frac{1}{N}\sum_{n=1}^{N}(\boldsymbol{x}_n-\bar{\boldsymbol{x}})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_n-\bar{\boldsymbol{x}}) \\ -\frac{N}{2}\frac{2}{N}\sum_{n=1}^{N}(\boldsymbol{x}_n-\bar{\boldsymbol{x}})^T\boldsymbol{\Sigma}^{-1}(\bar{\boldsymbol{x}}-\boldsymbol{\mu}) \\ -\frac{N}{2}(\bar{\boldsymbol{x}}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\bar{\boldsymbol{x}}-\boldsymbol{\mu}). \end{aligned} \tag{5.5}$$

The second term in (5.5) collapses to 0. The first term can be written as

$$-\frac{N}{2}\frac{1}{N}\sum_{n=1}^{N}tr\left((\boldsymbol{x}_n-\bar{\boldsymbol{x}})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}_n-\bar{\boldsymbol{x}})\right) \;=\; -\frac{N}{2}tr\left(Cov(\boldsymbol{x})\boldsymbol{\Sigma}^{-1}\right). \tag{5.6}$$

Replacing the exponent in (5.4) with the results (5.6) and (5.5) we get

$$(2\pi)^{-\frac{DN}{2}}\det(\boldsymbol{\Sigma})^{-\frac{N}{2}}\exp\left(-\frac{N}{2}(\bar{\boldsymbol{x}}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\bar{\boldsymbol{x}}-\boldsymbol{\mu})-\frac{N}{2}tr\left(Cov(\boldsymbol{x})\boldsymbol{\Sigma}^{-1}\right)\right),$$

which completes the proof. ∎

Lemma 5.1.1 plays a core role in the derivation of the *hierarchical Variational EM* algorithm as it allows to express the likelihood in terms of the image predictive density parameters $p_j, \boldsymbol{m}_j, \boldsymbol{\Lambda}_j$ and v_j. Since subset $\boldsymbol{X}_j$ is sampled i.i.d. from a multivariate Student-t density with mean $\boldsymbol{m}_j$, precision $\boldsymbol{\Lambda}_j$ and degrees of freedom v_j the sample mean and covariance are $\bar{\boldsymbol{X}}_j = \boldsymbol{m}_j$ and $Cov(\boldsymbol{X}_j) = \frac{v_j}{v_j-2}\boldsymbol{\Lambda}_j^{-1}$ respectively. The likelihood for the class

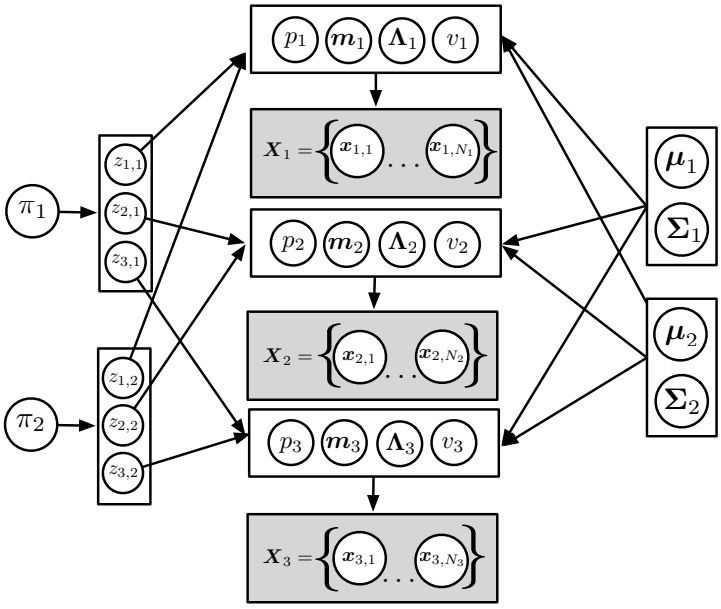

Figure 5.6: Schematic illustration of the dependencies between variables for the hierarchical mixture model using a *virtual sample* and Lemma 5.1.1 for the same example as in Figure 5.5. Notice that *virtual sample* blocks are conditionally independent from any parameters of the class conditional mixture model given the parameters of image densities.

conditional densities (5.3) therefore becomes

$$p(\boldsymbol{V}|\boldsymbol{Z},\boldsymbol{\theta}_c) = \prod_{j=1}^{C}\prod_{k=1}^{K_c} \pi_{k,c}^{z_{j,k}} \left[\mathcal{N}(\boldsymbol{m}_j|\boldsymbol{\mu}_{k,c},\boldsymbol{\Sigma}_{k,c}) \exp\left(-\frac{1}{2} tr\left(\frac{v_j}{v_j-2}\boldsymbol{\Lambda}_j^{-1}\boldsymbol{\Sigma}_{k,c}^{-1}\right)\right)\right]^{z_{j,k}N_j}. \tag{5.7}$$

Figure 5.6 depicts the dependencies between variables for equation (5.7).

From equation (5.7) we can derive the *variational hierarchical EM* algorithm by following the same reasoning as in Section 3.7.1. The posterior of the parameters and latent variables $p(\boldsymbol{Z},\boldsymbol{\theta}_c|\boldsymbol{V})$ will be approximated by a variational posterior which is assumed to factorise as $q(\boldsymbol{Z},\boldsymbol{\theta}_c) = q(\boldsymbol{Z})q(\boldsymbol{\theta}_c)$. Using the prior specified in Section 3.3 and applying equation (3.29) for the

log of the variational posterior for the latent variables we obtain

$$\begin{aligned}\log q(\boldsymbol{Z}) &= \mathbb{E}_{\boldsymbol{\pi}}[\log p(\boldsymbol{Z}|\boldsymbol{\pi})] + \mathbb{E}_{\boldsymbol{\mu},\boldsymbol{\Sigma}}[\log p(\boldsymbol{V}|\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{\Sigma})] + const \\ &= \sum_{j=1}^{C}\sum_{k=1}^{K_c} z_{j,k}\log\rho_{j,k} + const, \end{aligned} \tag{5.8}$$

where

$$\begin{aligned}\log\rho_{j,k} &= \mathbb{E}_{\pi_{k,c}}[\log\pi_k] - \frac{N_j}{2}\mathbb{E}_{\boldsymbol{\Sigma}_{k,c}}[\log(\det(\boldsymbol{\Sigma}_{k,c}))] - \frac{N_j D}{2}\log(2\pi) \\ &- \frac{N_j}{2}\mathbb{E}_{\boldsymbol{\mu}_{k,c},\boldsymbol{\Sigma}_{k,c}}\left[(\boldsymbol{m}_j - \boldsymbol{\mu}_{k,c})^T\boldsymbol{\Sigma}_{k,c}^{-1}(\boldsymbol{m}_j - \boldsymbol{\mu}_{k,c})\right] \\ &- \frac{N_j}{2}\mathbb{E}_{\boldsymbol{\Sigma}_{k,c}}\left[tr\left(\frac{v_j}{v_j-2}\boldsymbol{\Lambda}_j^{-1}\boldsymbol{\Sigma}_{k,c}^{-1}\right)\right] + const. \end{aligned} \tag{5.9}$$

Notice that this is similar to equation (3.33) for the standard variational EM. The terms dependent on $\boldsymbol{\mu}_{k,c}$ and $\boldsymbol{\Sigma}_{k,c}$ are scaled by a factor of N_j and there is an additional term involving the trace of the two covariance matrices. Exponentiating and re-normalising (5.9) to recover the constant term the variational posteriors for the the latent variables take the form of a Multinomial distribution with parameters $r_{n,k} = \rho_{n,k}/\sum_{k'=1}^{K_c}\rho_{n,k'}$.

$$q(\boldsymbol{z}_n) = \prod_{k=1}^{K_c} r_{n,k}^{z_{n,k}} = \mathcal{M}(\boldsymbol{z_n}|1, r_{n,1}, \ldots, r_{n,K}). \tag{5.10}$$

Similarly, applying (3.29) for the log of the variational posterior of the

parameters we get

$$\begin{aligned}
\log q(\boldsymbol{\theta}_c) &= \mathbb{E}_{\boldsymbol{Z}}[\log p(\boldsymbol{Z}|\boldsymbol{\pi})] + \mathbb{E}_{\boldsymbol{Z}}[\log p(\boldsymbol{V}|\boldsymbol{Z},\boldsymbol{\mu},\boldsymbol{\Sigma})] + \log p(\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}) + const \\
&= \sum_{j=1}^{C}\sum_{k=1}^{K_c} \mathbb{E}_{z_{j,k}}[z_{j,k}] \log \pi_{k,c} + \sum_{j=1}^{C}\sum_{k=1}^{K_c} \mathbb{E}_{z_{j,k}}[z_{j,k}] N_j \log \mathcal{N}(\boldsymbol{m}_j|\boldsymbol{\mu}_{k,c},\boldsymbol{\Sigma}_{k,c}) \\
&- \frac{1}{2}\sum_{j=1}^{C}\sum_{k=1}^{K_c} \mathbb{E}_{z_{j,k}}[z_{j,k}] N_j tr\left(\frac{v_j}{v_j-2}\boldsymbol{\Lambda}_j^{-1}\boldsymbol{\Sigma}_{k,c}^{-1}\right) \\
&+ \log p(\boldsymbol{\pi}) + \sum_{k=1}^{K_c} \log p(\boldsymbol{\mu}_{k,c},\boldsymbol{\Sigma}_{k,c}) + const. \qquad (5.11)
\end{aligned}$$

From which we can see that due to the prior and conditional independence between variables the variational posterior $q(\boldsymbol{\theta}_c)$ can be farther factored as a product of independent variational posteriors of the form

$$q(\boldsymbol{\theta}_c) = q(\boldsymbol{\pi}) \prod_{k=1}^{K_c} q(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k).$$

Using equation (5.11) and keeping only terms dependent on $\boldsymbol{\pi}$ we can get the variational posterior for the mixing coefficients which takes the form of a Dirichlet

$$q(\boldsymbol{\pi}) = \frac{\Gamma(\sum_{k=1}^{K_c} a_k)}{\prod_{k=1}^{K_c}\Gamma(a_k)} \prod_{k=1}^{K_c} \pi_k^{a_k-1} = \mathcal{D}(\boldsymbol{\pi}|\boldsymbol{a}), \qquad (5.12)$$

with parameters

$$a_k = a_0 + n_k, \qquad \text{where } n_k = \sum_{j=1}^{C} \mathbb{E}_{z_{j,k}}[z_{j,k}].$$

Finally, form (5.11) and keeping only terms dependent on $\boldsymbol{\mu}_{k,c}$ and $\boldsymbol{\Sigma}_{k,c}$ we can get that the variational posterior for the means and covariances is a

Normal-Inverse Wishart

$$q(\boldsymbol{\mu}_{k,c}, \boldsymbol{\Sigma}_{k,c}) = \mathcal{N}\left(\boldsymbol{\mu}_k | \boldsymbol{m}_k, \beta_k^{-1}\boldsymbol{\Sigma}_k\right) \mathcal{IW}\left(\boldsymbol{\Sigma}_k | \boldsymbol{W}_k, v_k\right), \tag{5.13}$$

with parameters

$$\begin{aligned}
\beta_k &= \beta_0 + \tilde{n}_k, \\
v_k &= v_0 + \tilde{n}_k, \\
\boldsymbol{m}_k &= \frac{\boldsymbol{m}_0\beta_0 + \tilde{n}_k\bar{\boldsymbol{x}}_k}{\beta_k}, \\
\boldsymbol{W}_k &= \boldsymbol{W}_0^{-1} + \tilde{n}_k\boldsymbol{V}_k + \frac{\tilde{n}_k\beta_0}{\tilde{n}_k + \beta_0}(\bar{\boldsymbol{x}} - \boldsymbol{m_0})(\bar{\boldsymbol{x}} - \boldsymbol{m_0})^T,
\end{aligned}$$

where

$$\tilde{n}_k = \sum_{j=1}^{C} \mathbb{E}_{z_{j,k}}[z_{j,k}]N_j, \quad \bar{\boldsymbol{x}}_k = \frac{1}{\tilde{n}_k}\sum_{j=1}^{C} \mathbb{E}_{z_{j,k}}[z_{j,k}]N_j\boldsymbol{m}_j,$$

$$\boldsymbol{V}_k = \frac{1}{\tilde{n}_k}\sum_{j=1}^{C} \mathbb{E}_{z_{j,k}}[z_{j,k}]N_j\left(\frac{v_j}{v_j - 2}\boldsymbol{\Lambda}_j^{-1} + (\boldsymbol{m}_j - \bar{\boldsymbol{x}}_k)(\boldsymbol{m}_j - \bar{\boldsymbol{x}}_k)^T\right).$$

The posterior parameters have the same expression as the parameters for the standard variational EM in Section 3.7.1. However the expressions for $\tilde{n}_k, \bar{\boldsymbol{x}}_k$ and $\boldsymbol{V}_k$ involve the additional terms N_j and $\frac{v_j}{v_j-2}\boldsymbol{\Lambda}_j^{-1}$ corresponding to the sample size and the sample covariance of subset $\boldsymbol{X}_j$ respectively. Therefore the *variational hierarchical EM* algorithm can be easily implemented as it requires very few changes to the Algorithm 7. As we did in the previous chapter the number of mixture components can be estimated by initially over-specifying the model and choosing a suitable prior for the mixing coefficients $\boldsymbol{\pi}$ such that sparse solutions are preferred and thus components with

negligible π_k can be safely removed.

5.1.7 Deterministic Annealing Variational EM

The variational EM iteratively updates the variational posteriors such that the lower bound (3.26) is maximised. The algorithm however is not guaranteed to converge to a global maximum solution and depends upon initialisation conditions. The reader should not be confused here with the local maxima problem in the standard EM. Although the reasons behind both problems are the same, the variational EM optimises a *functional* and provides a locally optimal approximation to the posterior. On the other hand the standard EM optimises a *function* and provides a locally optimum point estimate.

By inspecting equations (5.10) and (5.9) we can see that $\rho_{j,k}$ are up to a normalisation term equal to the expectation of the latent variables $z_{j,k}$. By exponentiating equation (5.9) we observe that the term N_j acts on the expectation of the latent variables in a similar way as in the *hierarchical EM* and thus we can interpret it as a temperature parameter in a deterministic annealing procedure. Therefore, the *variational hierarchical EM* algorithm not only allows to estimate the predictive density from a large number of observations in a hierarchical manner but it is also able to escape local minima that will arise during the optimisation.

5.2 Discriminative Kernel Classifiers

In a discriminative approach, we are no longer interest in modelling the class conditional or joint distributions of images and labels. Rather the focus is to model directly the posteriors $p(w_c|I)$ or find a discriminative function that separates the categories in the feature space.

Support Vector Machines (SVMs) have played an important role in Machine Learning and Pattern Recognition research. Despite their simple geometric interpretation they achieve improved classification performance on different domains. The main idea behind SVMs is to find a hyper plane that separates the data to positive and negative classes while it maximises the margin between the two classes.

Assume we are given a training set comprised by a set of N input vectors $\mathbf{x}_1, \ldots, \mathbf{x}_N$ and the corresponding target values, or class labels, $t_1, \ldots, t_N$ where $t_n \in \{-1, 1\}$. The first assumption is that the data are linearly separable and thus a linear function such as the one in Equation (5.14) has at least a set of parameters $\mathbf{w}$ and b such that $f(\mathbf{x}_n) > 0$ for points with $t_n = 1$ and $f(\mathbf{x}_n) < 0$ for points with $t_n = -1$ and thus a new point can be classified by the sign of $f(\mathbf{x}_{new})$. Note that $\boldsymbol{\phi}(\mathbf{x})$ is a non linear transformation of $\mathbf{x}$ mapping from points in the original space to a higher dimensional or possible infinite feature space. The discriminant function $f(\mathbf{x})$ will be linear in the feature space of $\boldsymbol{\phi}(\cdot)$ but non-linear in the original space.

$$f(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) + b. \tag{5.14}$$

Since the data are assumed to be linearly separable in $\boldsymbol{\phi}(\cdot)$ there will be infinitely many solutions that separate the data, however, we are interest in the one that achieves the maximum separation. That is, we seek to find the plane which separates the data while it has the maximum perpendicular distance from all points in both categories. Without going through the derivations, which can be found in any of the good resources on SVMs such as Campbell (2001), Cortes & Vapnik (1995), Bishop (2006), we can achieve that by minimizing Equation (5.15) with respect to the constraints (5.16).

$$\arg\min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2, \tag{5.15}$$

$$t_n(\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n) + b) \geq 1, n = 1, \ldots, N. \tag{5.16}$$

Cortes & Vapnik (1995) relax the linear separation assumption and allow some of the training points to be misclassified thus considering also the case of overlapping class distributions in $\boldsymbol{\phi}(\cdot)$. To achieve this they introduce one *slack* variable $\xi_n \geq 0$ for each training point where $\xi_n = 0$ for training points correctly classified, i.e. points on or inside the correct margin boundary, and $\xi_n = |t_n - f(\mathbf{x})|$ for all other points. To find the separating hyper plane then we have to minimize Equation (5.17) subject to the constraints in (5.18) where C is a regularisation parameter controlling the tradeoff between the slack variable penalty and the margin. As $C \to 0$ the previous version of SVM is recovered.

$$\arg\min_{\mathbf{w},b,\{\xi_n\}} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{n=1}^{N}\xi_n, \tag{5.17}$$

$$t_n(\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n) + b) \geq 1 - \xi_n, n = 1, \ldots, N. \tag{5.18}$$

There are two main difficulties in directly applying SVMs for image annotation and object class recognition. Firstly, SVMs are originally formulated as binary classifiers and extending them to multi-class problems is not straight-forward and hides several difficulties. One simple yet effective solution is to train one classifier for each category while using as negative instances samples from all the other categories. This is also called the one-against-all classification. Secondly, the original SVM formulation assumes that samples are represented by a feature vector thus making it difficult in extending them for objects represented by sets of feature vectors such as images. One solution is to use the *bag of terms* representation thus transforming every image into a single feature vector of visual term frequencies (Csurka et al. 2004). Another approach is to construct a suitable kernel $\boldsymbol{\phi}(\cdot)$ for the particular object representation.

5.2.1 Probability Product Kernel Support Vector Machine

The hyper plane $\boldsymbol{w}$ for the soft margin classifier discussed above can be written (Cortes & Vapnik 1995) as

$$\boldsymbol{w} = \sum_{n=1}^{N} y_n a_n \boldsymbol{\phi}(\boldsymbol{x}_n).$$

This implies that for a new unseen point $\boldsymbol{x}^*$ the discriminant function (5.14) depends only on dot products of the form

$$f(\mathbf{x}) = \sum_{n=1}^{N} y_n a_n \boldsymbol{\phi}(\mathbf{x}^*)\boldsymbol{\phi}(\boldsymbol{x}_n) + b. \tag{5.19}$$

The basic idea behind kernel based learning is to construct generalised inner products in Hilbert space using a symmetric positive definite kernel function (Cristianini & Shawe-Taylor 2000) such that

$$\boldsymbol{\phi}(\mathbf{x}^*)\boldsymbol{\phi}(\boldsymbol{x}_n) = K(\boldsymbol{x}^*, \boldsymbol{x}_n).$$

In this way we do not have to explicitly compute the transformations $\boldsymbol{\phi}(\boldsymbol{x})$ as the algorithms can be expressed purely in terms of the kernel function.

There is a significant research interest in designing kernel functions for objects represented as unordered set of vectors, such as images (Kondor & Jebara 2003, Moreno et al. 2003). In this thesis we propose first to model the density of features in each image and then utilise a kernel function in the space of probability densities. In this way we reduce the computational complexity associated with kernel functions dependent directly on the image features as we represent images by densities parameterised using a small set of parameters. In Section 4.4 we discussed the use of Probability Product Kernels PPK (Jebara et al. 2004) as means for calculating a generalised inner product in the space of density functions. The form of the PPK for

two images I_i and I_j has a simple form:

$$\mathrm{K}\left[\tilde{p}(\boldsymbol{x}|\boldsymbol{I}_i), \tilde{p}(\boldsymbol{x}|\boldsymbol{I}_j)\right] = \sum_{k=1}^{K_{I_i}} \sum_{k'=1}^{K_{I_j}} p_{I_i,k} p_{I_j,k'} \mathcal{N}\left(\boldsymbol{m}_{I_i,k} | \boldsymbol{m}_{I_j,k'}, \boldsymbol{\Lambda}_{I_i,k}^{-1} + \boldsymbol{\Lambda}_{I_j,k'}^{-1}\right), \tag{5.20}$$

where $p_I, k, \boldsymbol{m}_{I,k}$ and $\boldsymbol{\Lambda}_{I,k}$ are the parameters for the image predictive densities obtained by the *variational EM* algorithm.

5.3 Relations with other models

Early approaches for the automatic image annotation problem where based on the Bag of Terms representation, e.g. (Csurka et al. 2004, Duygulu et al. 2002) and (Jeon et al. 2003). In Csurka et al. (2004) the visual vocabulary is constructed by clustering SIFT feature descriptors using K-means. Images are then transformed into histograms counting the presence of visual terms and the Naive Bayes and Support Vector Machine classifiers are applied. Due to the sparsity and high dimensionality of the histograms a linear kernel is used for the SVM. Their methodology can be easily extended in a Bayesian framework by the Multinomial-Dirichlet model as presented in Section 5.1.4.

Duygulu et al. (2002) treat the image annotation problem as a statistical machine translation problem where the aim is to translate from visual terms to keywords and thus the training set is treated as a bi-lingual collection. For estimating the joint probability of keywords and visual terms they apply standard models from statistical machine translation (Brown et al. 1993). Similarly Jeon et al. (2003) estimates the joint probability of key-

words and visual terms using the *Cross-lingual* relevance models of Lavrenko et al. (2002). Soon it was realised that the accuracy of classifiers trained with the Bag-of-Terms approach was sensitive to quantisation errors introduced by the K-means procedure. Lavrenko et al. (2003) generalised the model of Jeon et al. (2003) for continuous features by modelling the image densities using a kernel density estimation procedure. Similarly, Yavlinsky et al. (2005) develop a Naive Bayes classifier using a kernel density estimator for the class conditional densities.

Despite its success, non-parametric kernel density estimation is a computationally exhaustive method as it requires the computation of some kernel function with respect to all training images. Semi-parametric models such as mixtures of Gaussians have been used in Magalhães & Rüger (2006) and Carneiro et al. (2007) as an alternative. The method of Magalhães & Rüger (2006) is more similar to that of Csurka et al. (2004) since a Gaussian mixture model is fitted on features from all images in the collection to create a visual vocabulary where each term is represented by a Gaussian component. For classification, a logistic regression classifier is developed where the covariates are the probabilities of visual term components in each image.

In Chapter 4 we discussed the relationships between the Bag-of-Terms representation and density estimation and we have seen how the maximum likelihood solutions are in fact an approximation to the predictive densities. In this chapter we applied the same reasoning in order to model class conditional densities and develop Naive Bayes' and SVM classifiers using the predictive densities. The Naive Bayes classifier is similar to that of Carneiro

et al. (2007) with the difference that we marginalise the parameters for the class conditional densities and obtain the model order (number of components) automatically. Finally, the SVM classifier presented here is similar to that used by many Bag-of-Terms approaches such as those in Csurka et al. (2004). However, by modelling the image densities directly we avoid the quantisation errors and the computational complexity associated with K-means applied on a large number of high dimensional feature descriptors.

5.4 Experiments

To empirically evaluate the performance of the methods presented here we applied them on an image classification task. Classification accuracy is measured in terms of precision and recall for each keyword. Moreover, using the classified images we evaluate semantic image retrieval for single keyword queries in terms of Mean Average Precision (MAP) and precision at different recall cutoff points, e.g P@10. For the experiments of this section we use the Corel5K collection discussed in Chapter 4. The ground truth comprises only the keywords associated with each image and not the high level image categories. In total there are 260 distinct keywords in the training set of 4,500 images. The collection is pre-processed as discussed in Section 4.5.1 and the parameters estimated by the GMM-QL and PD-QL methods are re-used as inputs for the hierarchical EM and variational hierarchical EM algorithms respectively.

For the hierarchical EM algorithm we follow Carneiro et al. (2007) and

fit mixtures of 64 components while for the variational algorithm presented in this Chapter we use 200 components and after convergence we remove components with $\widetilde{n}_k = 0$. The priors are again set as discussed in Section 3.3 since we want to take into account collection statistics for smoothing our estimates. The keyword priors $p(w_c)$ for all cases are set to the normalised frequency of the keywords in the training collection. For the SVM classifiers the α parameter of the Probability Product Kernel (PPK) is set to 1 and the parameters estimated by the GMM-QL and PD-QL methods in Section 4.5.1 are re-used as inputs for calculating the Kernel Gram matrix. One SVM classifier for each keyword is trained in the one-versus-all scheme where the negative class consists of all images in the training collection that don't contain the corresponding keyword.

5.5 Results

Table 5.1 shows the annotation results in terms of precision, recall, F-score and the number of keywords with recall larger than zero. All results are

Method	**words R** > 0	**Avg. Prec.**	**Recall**	**F score**
SVM-EM	121	0.088	0.321	0.138
SVM-VEM	83	$\mathbf{0.129}^*$	0.195	$\mathbf{0.155}^*$
HGMM-EM	107	$\mathbf{0.138}^*$	$\mathbf{0.244}^*$	$\mathbf{0.176}^*$
HGMM-VEM	107	$\mathbf{0.214}^*$	$\mathbf{0.337}^*$	$\mathbf{0.262}^*$

Table 5.1: Image annotation results for 500 query images in the test set. $*$ indicates statistical significance from the results of the previous row using a Wilcoxon rank-sum test with 1% significance level. Results are averages over the 260 keywords in the collection.

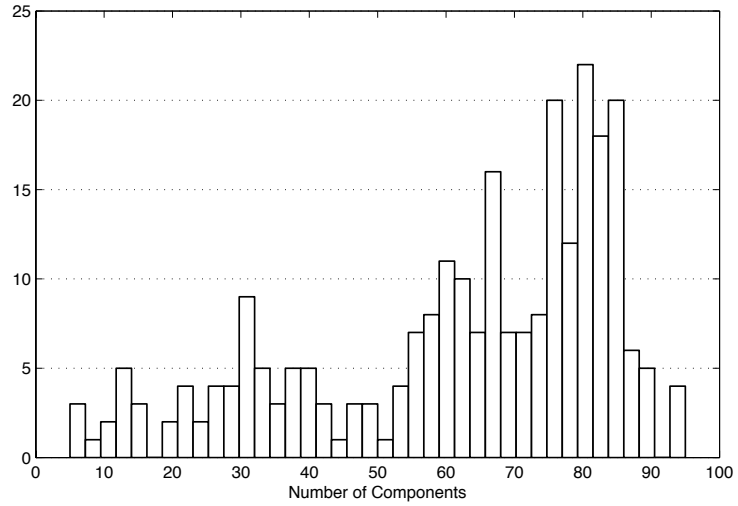

Figure 5.7: Distribution of the number of components K for the 260 class conditional densities as estimated by the hierarchical variational EM algorithm.

averaged over the whole vocabulary of 260 keywords in order to ensure a fair comparison. We can see that utilising the predictive densities obtained by marginalising the approximate posteriors from the variational EM algorithm significantly improves performance in all cases. Figure 5.7 depicts the distribution of the number of components for the 260 class conditional densities calculated by the hierarchical variational EM algorithm. The mean is 62.41 with standard deviation 22.3 which explains the results of Carneiro et al. (2007), where the best performance is obtained by setting the number of components for the class conditional densities to 64. In contrast to the results in Chapter 4 the distribution can not be approximated by a normal and there is a large variation from the mean which also justifies the higher increase in performance compared to the EM algorithm.

The poor performance of SVM based models compared to the Naive Bayes' classifiers can be attributed to the following. The test images are also modelled with Gaussian mixture models and the PPK is calculated between training and test image densities. The modelling of test images by a mixture model can be seen as a lossy compression of the information carried by the feature descriptors. In contrast, in the Naive Bayes' classifiers the likelihood, or the predictive likelihood, is directly estimated by the test image's feature descriptors. However, in our experiments, classifying new images with the SVM classifiers is significantly faster compared to the Naive Bayes' classifiers.

Having annotated images with keywords we can evaluate the retrieval performance of semantic retrieval. In contrast to classification where a hard decision is made, in semantic retrieval the user submits a keyword query and images are ranked based on $p(w_q|I)$. Table 5.2 presents results using 260 single keyword queries for retrieving images from the 500 test images in the test set. Figure 5.8 also shows the interpolated precision recall curves for the four methods. For obtaining a probability estimate from the SVM classifiers we fitted sigmoid function as described in Platt (2000).

Method	**MAP**	**R-Prec.**	**P@5**	**P@10**	**P@20**
SVM-EM	0.0539	0.0745	0.0783	0.0646	0.0432
SVM-VEM	**0.0776***	**0.0744***	**0.1004***	**0.0715***	**0.0548***
HGMM-EM	**0.1081***	**0.0979***	**0.1163***	**0.0837***	**0.0523***
HGMM-VEM	**0.1219***	**0.1307***	**0.1437***	**0.1141***	**0.0741***

Table 5.2: Retrieval results for 260 query keywords in the test set. $*$ indicates statistical significance from the results of the previous row using a Wilcoxon rank-sum test with 1% significance level.

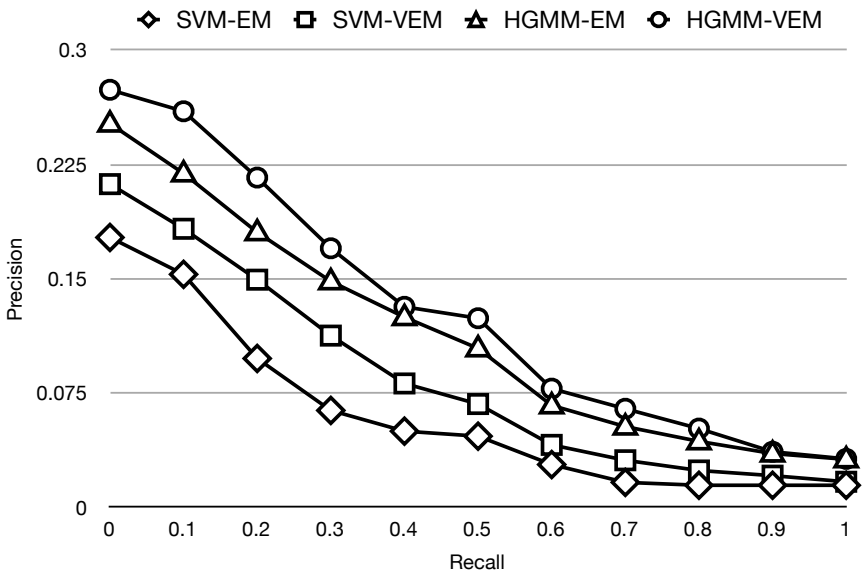

Figure 5.8: Interpolated Precision-Recall curves.

5.5.1 Note on computational complexity

Both families of algorithms discussed in this chapter, namely SVM-EM, SVM-VEM and HGMM-EM, HGMM-VEM, require the images in the training set to be modelled using Gaussian mixture models. This is done using the results from Chapter 4 and thus the discussion in Section 4.6.1 applies here too.

For the training phase of SVM models the Kernel matrix has to be computed which requires $N_c \times (N_c - 1)/2$ evaluations of the PPK, equation (5.20), where N_c is the number of training images for class c. Moreover, the quadratic optimisation problem for SVMs scales as $O(N_c^3)$ although modern solvers exploiting the particular structure of the SVM optimisation problem scale as $O(N_c^2)$ (Bottou & Lin 2007). Additionally, in order to estimate the

regularisation parameter C in equation (5.17) a cross validation scheme is needed.

Despite the quadratic scaling of the training phase, the SVMs solutions are usually sparse. That is, many of the parameters α_n in equation (5.19) are zero and thus predictions can bee made in a very efficient manner since the Kernel function needs to be evaluated only for the set of images in the training set with non-zero α_n parameter. In the experiments presented in this chapter the number of non-zero α_n parameters for the each class was ranging from 3, for classes with few training examples, to 116 for classes with many training examples. However the query image also needs to be modelled by a Gaussian mixture density and thus a EM or variational EM algorithm needs to be run for each test image before prediction.

For the Naive Bayes classifiers using the hierarchical EM and hierarchical VEM algorithms to model the class conditional densities, training time scales linearly with the number of training examples for each class. Note however that for the hierarchical EM algorithm a cross validation scheme is needed in order to determine the optimal number of mixture components. In contrast for the hierarchical VEM the number of mixture components is automatically determined using a single run of the algorithm starting from a model with many components.

For prediction using the Naive Bays classifiers the likelihood of the query image has to be estimated for each class conditional density model. For each class conditional density, estimation of the query image likelihood scales as $O(N_Q \times K_c \times D)$ where $N_Q = 1,457$ is the number of feature vectors of the

query image, K_c is the number of mixture components for class c and $D = 70$ is the dimensionality of the feature vectors.

5.6 Discussion

In this chapter we have presented classifiers for automatically annotating images into semantic categories in order to facilitate semantic retrieval using keyword queries. Specifically we developed a generalisation of the image annotation algorithm of Carneiro et al. (2007) using the predictive likelihood for the class conditional densities. Our algorithm differs in that instead of making a single point estimate of the mixture model parameters, we marginalise them using their posterior. In that way we avoid over-fitting and local maxima problems. Moreover the number of mixture components is automatically estimated from the data and there is no need to manually tune algorithm parameters. The only free model parameters are those associated with the prior for which we can use the collection statistics.

We have also explored the applicability of kernel based classifiers such as Support Vector Machines using the Probability Product Kernel (PPK) (Jebara et al. 2004) for computing generalised inner products in the space of probability density functions. In contrast to the Bag-of-Terms method, we directly model the density of continuous image feature descriptors without resorting to quantisation procedures such as K-means. The densities of feature descriptors for each image are modelled using Gaussian mixture models for which we use the EM algorithm to obtain maximum likelihood parameter

estimates and the variational EM to directly obtain the predictive densities.

To validate our approach we conduct experiments on the Corel 5K collection and evaluate annotation and semantic retrieval performance. Our results indicate that the predictive densities provide a better approximation of the true models and thus result in statistically significant improvements. In contrast to Csurka et al. (2004), in our experiments the SVM classifiers do not achieve the same level of performance as the Naive Bayes classifiers. This can be attributed to the nature of the image collection. In Csurka et al. (2004) the collection used is more similar to the Caltech256 Griffin et al. (2007) collection where each image is associated with one class. The Corel5K collection is more challenging for SVMs since the training sets for each binary classifier are overlapping across many different classes. Moreover, in Csurka et al. (2004), the same image representation is used for both the Naive Bayes and SVM classifiers, i.e. images are represented by the frequency of visual terms. In our approach the Naive Bayes classifiers utilise the image feature descriptors of the test image and calculate their joint probability under each class conditional model. In contrast the SVM classifiers approximate the density of the test image using a Gaussian mixture model. Finally, although the asymptotic error of generative classifiers is much higher than discriminative methods, they often perform better when the size of the training set is small (Ng & Jordan 2001). In the Corel5K collection used here the majority of the classes have very few training instances.

Chapter 6

Conclusions

In this thesis we have studied the application of Bayesian inference for developing image retrieval systems. We focused on a particular family of models based on Gaussian mixtures modelling the distribution of image features in images in a similar way language models for information retrieval are used to model term frequencies in documents. Based on these models we developed algorithms for query by example and semantic image retrieval. The following sections discuss our findings and provide interesting future research directions.

6.1 Discussion

6.1.1 Bayesian inference for information retrieval

Bayesian inference provides a sound theoretical framework in which we can interpret smoothing procedures frequently used in the development of infor-

mation retrieval systems. Moreover it allows to handle the uncertainty in the model parameters when making predictions by marginalising using the posterior distribution. For models developed in the language modelling framework for information retrieval the posterior and predictive distributions have exact analytical form (Zaragoza et al. 2003) and thus they can be directly applied in practical implementations. In Chapter 4 we show that this is also directly applicable for Bag of Terms models for image retrieval.

For most models of practical interest the posterior and predictive densities do not have closed form solutions. In Chapter 3 we discuss how we can obtain samples from the posterior of Gaussian mixture models in order to numerically estimate the predictive densities required for ranking images. We have evaluated several MCMC algorithms in terms of efficiency and computational complexity. Our empirical findings suggest that Riemann manifold MCMC algorithms (Girolami & Calderhead 2011) have better mixing and convergence properties and require significantly less manual tuning provided higher order derivatives of the joint log likelihood function and the Fisher information are available. The Metropolis-Hastings although is the least efficient from the algorithms we evaluated allows for a generic implementation since it only requires the joint log likelihood which is directly available for generative probabilistic models.

The application of MCMC algorithms for practical information retrieval systems is not straight forward however. In the language modelling framework and for the models considered in this thesis we have to estimate the posterior for every document or image in the collection. This means that for

each image or document several chains have to be run in order to assess convergence and then sub-sampled. The indexing structure has to store several posterior samples of high dimension for every item in the collection and at the retrieval phase these samples have to be used for obtaining a numerical estimate of the predictive density. Finally, a less serious issue is that tuning and monitoring of convergence cannot be easily automated. In Chapter 3 we discussed variational approximation as a viable alternative to MCMC for information retrieval systems. In Chapters 4 and 5 we show that the methodology can be applied in practice to develop image retrieval systems.

6.1.2 Predictive densities for image retrieval models

In Chapter 4 we generalise the image retrieval model proposed by Westerveld et al. (2003) and Vasconcelos & Lippman (2000) using the predictive image densities for ranking. We showed that retrieval performance is significantly improved compared to maximum likelihood and maximum a posteriori estimates while the variational EM algorithm has the same order of complexity as the EM algorithm. In a Bayesian inference framework the number of mixture components modelling the distribution of feature descriptors in images can be automatically obtained without resorting to external optimisation procedures such as cross validation. This is an important characteristic of the proposed methodology as the number of components is optimised for each image in the collection independently.

The methodology has been also applied for developing SVM and Naive Bayes classifiers. The Naive Bayes classifier proposed in this thesis generalises

the approach of Carneiro et al. (2007) and in Chapter 5 we show that the proposed method significantly improves classification and semantic retrieval performance.

6.1.3 Bag of Terms and generative probabilistic models

The Bag of Terms representation has allowed well studied information retrieval models to be applied for image retrieval. However, the quantisation errors introduced at the code block generation stage have been shown to negatively affect the discriminative ability of image descriptors Boiman et al. (2008). Moreover, weighting and ranking functions used in information retrieval make particular assumptions about the distribution of terms in the collection which are not necessarily valid for the visual vocabulary generated by the K means algorithm. Finally, the quantisation and code block generation stages are computationally demanding tasks. Approximations using vocabulary trees Nister & Stewenius (2006), Philbin et al. (2007) significantly reduce the computational overhead but they exacerbate the quantisation problems.

In Chapter 4 we compared algorithms based on the Bag of Terms representation and algorithms that directly model the density of continuous image features using Gaussian mixture models. Our results are in agreement with previous research (Lavrenko et al. 2003, Yavlinsky et al. 2005, Boiman et al. 2008) and suggest that the quantisation errors negatively affect retrieval performance. Indexing of the image collection is also simplified since

the Variational EM algorithm for approximating the parameters posterior for each image in the collection can be easily parallelised. In Chapter 4 we also proposed to model the query density using a Gaussian mixture model and rank images in the collection using the Probability Product Kernel between densities. This simplifies the retrieval algorithm at the expense of retrieval performance. Despite the superior retrieval performance of the mixture models compared to the Bag of Terns representation, scalability to large scale collections remains an important issue since the predictive densities for all images in the collection have to be evaluated for each query.

6.2 Future work

6.2.1 Approximate retrieval using LSH

In Chapter 4 we discussed the difficulty in designing efficient data structures for image retrieval using Gaussian mixture models in order to allow algorithms to scale for large image collections. Given the large relative difference in retrieval performance compared to Bag of Terms models we believe that is crucial to further investigate this issue. Even if approximations must be considered as we did in Section 4.4 the performance margin remains considerably large.

Kulis & Grauman (2009) have recently developed the methodology for performing Locality Sensitive Hashing (LSH) (Andoni & Indyk 2006) over arbitrary kernel functions. Given the results in Section 4.4 for the Probability Product Kernel we believe that this is a natural extension of our work.

6.2.2 Modelling correlations between local descriptors

The models considered in this thesis assume that local image feature descriptors are conditionally independent. This is an over simplifying assumption necessary to obtain tractable models and is similar to the term independence assumption in text document retrieval. It has been shown that by exploiting correlations between local descriptors retrieval performance can be improved (Philbin et al. 2007). However, for the Bag of Terms model such procedures are applied in a post processing step (Lowe 2004) and are difficult to be casted in a probabilistic framework. Fergus et al. (2003) have studied a generative probabilistic model that takes into account the spatial geometry of local feature descriptors for image classification although it also relies on a Bag of Terms representation. Generalisations of this model such as those presented in the chapter is an interesting future direction.

Bibliography

Amari, S.-i. & Nagaoka, H. (2000), *Methods of Information Geometry*, Translations of Mathematical Monographs, American Mathematical Society.

Andoni, A. & Indyk, P. (2006), Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions, *in* 'Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science', IEEE Computer Society, Washington, DC, USA, pp. 459–468.

Andrieu, C. & Thoms, J. (2008), 'A tutorial on adaptive MCMC', *Statistics and Computing* **18**(4), 343–373.

Attias, H. (2000), A variational bayesian framework for graphical models, *in* 'In Advances in Neural Information Processing Systems 12', MIT Press, pp. 209–215.

Basu, A., Harris, Ian, R., Hjort, N. L. & Jones, M. C. (1998), 'Robust and efficient estimation by minimising a density power divergence', *Biometrika* **85**(3), 549–559.

Berger, A. & Lafferty, J. (1999), Information retrieval as statistical translation, *in* 'Proceedings of the 22nd annual international ACM SIGIR con-

ference on Research and development in information retrieval', SIGIR '99, ACM, New York, NY, USA, pp. 222–229.

Bishop, C. M. (2006), *Pattern Recognition and Machine Learning*, Springer.

Bishop, C. M. & Corduneanu, A. (2001), Variational Bayesian model selection for mixture distributions, *in* 'Artificial Intelligence and Statistics'.

Blei, D. M. & Jordan, M. I. (2003), Modeling annotated data, *in* 'Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval', SIGIR '03, ACM, New York, NY, USA, pp. 127–134.
URL: *http://doi.acm.org/10.1145/860435.860460*

Blei, D. M., Ng, A. Y. & Jordan, M. I. (2003), 'Latent dirichlet allocation', *J. Mach. Learn. Res.* **3**, 993–1022.
URL: *http://dl.acm.org/citation.cfm?id=944919.944937*

Boiman, O., Shechtman, E. & Irani, M. (2008), In defense of nearest-neighbor based image classification, *in* 'Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on', pp. 1 –8.

Bottou, L. & Lin, C.-J. (2007), Support Vector Machine Solvers, *in* L. Bottou, O. Chapelle, D. decoste & J. Weston, eds, 'Large Scale Kernel Machines', MIT Press, Cambridge, MA, USA, pp. 301–320.

Brooks, S. P. & Gelman, A. (1998), 'General methods for monitoring convergence of iterative simulations', *Journal of Computational and Graphical Statistics* **7**(4), pp. 434–455.

Brown, P. F., Pietra, V. J. D., Pietra, S. A. D. & Mercer, R. L. (1993), 'The mathematics of statistical machine translation: parameter estimation', *Comput. Linguist.* **19**, 263–311.

Campbell, C. (2001), An introduction to kernel methods, *in* 'Radial basis function networks 1: recent developments in theory and applications', Physica Verlag Rudolf Liebing KG, Vienna, Austria, Austria, pp. 155–192.

Carneiro, G., Chan, A. B., Moreno, P. J. & Vasconcelos, N. (2007), 'Supervised learning of semantic classes for image annotation and retrieval', *TPAMI: IEEE Transactions on Pattern Analysis and Machine Intelligence* **29**(3), 394–410.

Carneiro, G. & Vasconcelos, N. (2005), Formulating semantic image annotation as a supervised learning problem, *in* 'CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2', IEEE Computer Society, Washington, DC, USA, pp. 163–168.

Celeux, G. (2006), Mixture models for classification, *in* 'Advances in Data Analysis, Proceedings of the 30th Annual Conference of the Gesellschaft für Klassifikation e.V., Freie Universität Berlin', pp. 3–14.

Christensen, O., F., Roberts, G., O. & Rosenthal, J., S. (2005), 'Scaling limits for the transient phase of local Metropolis-Hastings algorithms', *Journal Of The Royal Statistical Society Series B* **67**(2), 253–268.

Chum, O., Philbin, J. & Zisserman, A. (2008), Near duplicate image detec-

tion: min-hash and tf-idf weighting, *in* 'British Machine Vision Conference'.

Cortes, C. & Vapnik, V. (1995), 'Support vector networks', *Machine Learning* **20**(3), 273–297.

Cristianini, N. & Shawe-Taylor, J. (2000), *An introduction to support Vector Machines: and other kernel-based learning methods*, Cambridge University Press, New York, NY, USA.

Csurka, G., Dance, C. R., Fan, L., Willamowski, J. & Bray, C. (2004), Visual categorization with bags of keypoints, *in* 'ECCV International Workshop on Statistical Learning in Computer Vision'.

Dean, J. & Ghemawat, S. (2008), 'Mapreduce: simplified data processing on large clusters', *Commun. ACM* **51**(1), 107–113.
URL: *http://doi.acm.org/10.1145/1327452.1327492*

Dellaportas, P. & Papageorgiou, I. (2006), 'Multivariate mixtures of normals with unknown number of components', *Statistics and Computing* **16**(1), 57–68.

Deselaers, T., Keysers, D. & Ney, H. (2008), 'Features for image retrieval: an experimental comparison', *Inf. Retr.* **11**, 77–107.

Dietterich, T. G., Lathrop, R. H. & Lozano-Pérez, T. (1997), 'Solving the multiple instance problem with axis-parallel rectangles', *Artificial Intelligence* **89**(1-2), 31–71.

Donald, K. & Smeaton, A. (2005), A Comparison of Score, Rank and Probability-Based Fusion Methods for Video Shot Retrieval, *in* 'Proceesings of 4th International Vonference CIVR', Springer, pp. 61–70.

Duane, S., Kennedy, A., B., Pendleton, J., B. & Roweth, D. (1987), 'Hybrid Monte Carlo', *Physics Letters B* **195**(2), 216–222.

Duygulu, P., Barnard, K., de Freitas, J. F. G. & Forsyth, D. A. (2002), Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary, *in* 'ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part IV', Springer-Verlag, London, UK, pp. 97–112.

Escobar, Michael, D. & West, M. (1994), 'Bayesian density estimation and inference using mixtures', *Journal of the American Statistical Association* **90**, 577–588.

Feng, S. L., Manmatha, R. & Lavrenko, V. (2004), Multiple bernoulli relevance models for image and video annotation, *in* 'Proceedings of the 2004 IEEE computer society conference on Computer vision and pattern recognition', CVPR'04, IEEE Computer Society, Washington, DC, USA, pp. 1002–1009.
URL: *http://dl.acm.org/citation.cfm?id=1896300.1896446*

Fergus, R., Perona, P. & Zisserman, A. (2003), Object class recognition by unsupervised scale-invariant learning, *in* 'Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition', Vol. 2, pp. 264–271.

Fergus, R., Perona, P. & Zisserman, A. (2005), A sparse object category model for efficient learning and exhaustive recognition, *in* 'CVPR '05:

Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1', IEEE Computer Society, Washington, DC, USA, pp. 380–387.

Gelman, A., Carlin, J. B., Stern, H. S. & Rubin, D. B. (2003), *Bayesian Data Analysis, Second Edition (Chapman & Hall/CRC Texts in Statistical Science)*, 2 edn, Chapman and Hall/CRC.

Geman, S. & Geman, D. (1984), 'Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **6**(6), 721–741.

Geyer, Charles, J. (1992), 'Practical Markov chain Monte Carlo', *Statistical Science* **7**(4), 473–483.

Girolami, M. & Calderhead, B. (2011), 'Riemann manifold langevin and hamiltonian monte carlo methods', *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **73**(2), 123–214.
URL: *http://dx.doi.org/10.1111/j.1467-9868.2010.00765.x*

Green, P., J. (1995), 'Reversible jump Markov chain Monte Carlo computation and Bayesian model determination', *Biometrika* **82**(4), 711–732.

Griffin, G., Holub, A. & Perona, P. (2007), Caltech-256 object category dataset, Technical Report 7694, California Institute of Technology.

Haario, H., Saksman, E. & Tamminen, E. (2005), 'Componentwise adaptation for high dimensional MCMC', *Computational Statistics* **20**, 265–273.

Haario, H., Saksman, E. & Tamminen, J. (1998), 'An adaptive Metropolis algorithm', *Bernoulli* **7**, 223–242.

Hamilton, E. (1992), Jpeg file interchange format, Technical report, C-Cube Microsystems, Milpitas, CA, USA.

Harris, C. & Stephens, M. (1988), A Combined Corner and Edge Detection, *in* 'Proceedings of The Fourth Alvey Vision Conference', pp. 147–151.

Harter, S. P. (1975), 'A probabilistic approach to automatic keyword indexing. part i and part ii', *Journal of the American Society for Information Science* **26**(4), 197–206.

Hershey, J. R. & Olsen, P. A. (2007), Approximating the Kullback Leibler Divergence Between Gaussian Mixture Models, *in* 'IEEE International Conference on Acoustics, Speech and Signal Processing, 2007. ICASSP 2007.', Vol. 4, pp. IV–317–IV–320.
URL: *http://dx.doi.org/10.1109/ICASSP.2007.366913*

Hiemstra, D. (2001), Using Language Models for Information Retrieval, PhD thesis, University of Twente, Enschede.

Hofmann, T. (1999), Probabilistic latent semantic indexing, *in* 'Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval', SIGIR '99, ACM, New York, NY, USA, pp. 50–57.
URL: *http://doi.acm.org/10.1145/312624.312649*

Jasra, A., Holmes, C., C. & Stephens, D., A. (2005), 'MCMC and the label

switching problem in Bayesian mixture models', *Statistical Science* **20**, 50–67.

Jebara, T., Kondor, R. & Howard, A. (2004), 'Probability product kernels', *J. Mach. Learn. Res.* **5**, 819–844.

Jegou, H., Douze, M. & Schmid, C. (2009), On the burstiness of visual elements, *in* 'Computer Vision and Pattern Recognition', pp. 1169–1176.

Jeon, J., Lavrenko, V. & Manmatha, R. (2003), Automatic image annotation and retrieval using cross-media relevance models, *in* 'SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval', ACM Press, New York, NY, USA, pp. 119–126.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S. & Saul, L. K. (1998), An introduction to variational methods for graphical models, *in* 'Proceedings of the NATO Advanced Study Institute on Learning in graphical models', Kluwer Academic Publishers, Norwell, MA, USA, pp. 105–161.

Kass, R. E. & Raftery, A. E. (1995), 'Bayes factors', *Journal of the American Statistical Association* **90**(430), pp. 773–795.

Kondor, R. & Jebara, T. (2003), A kernel between sets of vectors, *in* 'Proc. of ICML-2003, Washington DC.'.

Koniusz, P. & Mikolajczyk, K. (2010), On a quest for image descriptors based on unsupervised segmentation maps, *in* 'Proceedings of the 2010

20th International Conference on Pattern Recognition', ICPR '10, IEEE Computer Society, Washington, DC, USA, pp. 762–765.

Kshirsagar, A. M. (1959), 'Bartlett decomposition and wishart distribution', *The Annals of Mathematical Statistics* **30**(1), pp. 239–241.

Kühnel, W. (2005), *Differential Geometry: Curves - Surfaces - Manifolds*, Vol. 16 of *Student Mathematical Library*, AMS.

Kulis, B. & Grauman, K. (2009), Kernelized locality-sensitive hashing for scalable image search, *in* 'ICCV', pp. 2130–2137.

Lavrenko, V., Choquette, M. & Croft, W. B. (2002), Cross-lingual relevance models, *in* 'Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval', SIGIR '02, ACM, New York, NY, USA, pp. 175–182.

Lavrenko, V., Manmatha, R. & Jeon, J. (2003), A model for learning the semantics of pictures, *in* S. Thrun, L. Saul & B. Schölkopf, eds, 'Advances in Neural Information Processing Systems 16', MIT Press, Cambridge, MA.

Llorente, A. & Rüger, S. (2009), Using second order statistics to enhance automated image annotation, *in* 'Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval', ECIR '09, Springer-Verlag, Berlin, Heidelberg, pp. 570–577.

Lowe, D. G. (2004), 'Distinctive image features from scale-invariant keypoints', *Int. J. Comput. Vision* **60**(2), 91–110.

Luhn, H. P. (1957), 'A Statistical Approach to Mechanized Encoding and Searching of Literary Information', *IBM Journal of Research and Development* **1**(4), 309–317.

Lux, M. (2009), Caliph & emir: Mpeg-7 photo annotation and retrieval, *in* 'Proceedings of the 17th ACM international conference on Multimedia', MM '09, ACM, New York, NY, USA, pp. 925–926.

MacKay, D. J. C. & Bauman Peto, L. C. (1994), 'A Hierarchical Dirichlet Language Model', *Natural Language Engineering* **1**, 1–19.

Magalhães, J. a. & Rüger, S. (2010), 'An information-theoretic framework for semantic-multimedia retrieval', *ACM Trans. Inf. Syst.* **28**, 19:1–19:32. **URL:** *http://doi.acm.org/10.1145/1852102.1852105*

Magalhães, J. & Rüger, S. M. (2006), Logistic regression of generic codebooks for semantic image retrieval, *in* 'CIVR', pp. 41–50.

Manning, C. D., Raghavan, P. & Schtze, H. (2008), *Introduction to Information Retrieval*, Cambridge University Press, New York, NY, USA.

Marin, M. J., Mengersen, K. & Robert, C., P. (2005), 'Bayesian Modelling and Inference on Mixtures of Distributions', *Handbook of Statistics* pp. 15840–15845.

Maron, O. & Lozano-Pérez, T. (1998), A framework for multiple-instance learning, *in* 'NIPS '98: Proceedings of the 1998 conference on Advances in neural information processing systems', Vol. 10, MIT Press, Cambridge, MA, USA, pp. 570–576.

McLachlan, G. J. & Baek, J. (2010), 'Clustering of high-dimensional data via finite mixture models', *Advances in Data Analysis, Data Handling and Business Intelligence* pp. 33–44.

McLachlan, G. & Peel, D. (2000), *Finite Mixture Models*, Wiley Series in Probability and Statistics, Wiley-Interscience.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. & Teller, E. (1953), 'Equation of State Calculations by Fast Computing Machines', *The Journal of Chemical Physics* **21**(6), 1087–1092.
URL: *http://dx.doi.org/10.1063/1.1699114*

Mikolajczyk, K., Leibe, B. & Schiele, B. (2005), Local features for object class recognition, *in* 'ICCV: IEEE International Conference on Computer Vision', IEEE Computer Society, Washington, DC, USA, pp. 1792–1799.

Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T. & Van Gool, L. J. (2005), 'A comparison of affine region detectors', *International Journal of Computer Vision* **65**(1-2), 43–72.

Miller, D. R. H., Leek, T. & Schwartz, R. M. (1999), A hidden markov model information retrieval system, *in* 'Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval', SIGIR '99, ACM, New York, NY, USA, pp. 214–221.

Moreno, P. J., Ho, P. P. & Vasconcelos, N. (2003), A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications, *in* 'In Advances in Neural Information Processing Systems 16'.

Nasios, N. & Bors, A. (Aug. 2006), 'Variational learning for gaussian mixture models', *Systems, Man, and Cybernetics, Part B, IEEE Transactions on* **36**(4), 849–862.

Neal, Radford, M. (1993), Probabilistic inference using Markov chain Monte Carlo methods, Technical Report CRG-TR-93-1, Dept. of Computer Science, University of Toronto.

Neal, Radford, M. (1999), 'Suppressing random walks in Markov chain Monte Carlo using ordered overrelaxation', pp. 205–228.

Newton, M. A. & Raftery, A. E. (1994), 'Approximate bayesian inference with the weighted likelihood bootstrap', *Journal of the Royal Statistical Society. Series B (Methodological)* **56**(1), pp. 3–48.

Ng, A. Y. & Jordan, M. I. (2001), On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes, *in* 'NIPS', pp. 841–848.

Nister, D. & Stewenius, H. (2006), Scalable recognition with a vocabulary tree, *in* 'Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on', Vol. 2, pp. 2161 – 2168.

Nowak, E., Jurie, F. & Triggs, B. (2006), Sampling strategies for bag-of-features image classification, *in* 'European Conference on Computer Vision', Springer.

Nowak, S., Llorente, A., Motta, E. & Rüger, S. (2010), The effect of semantic relatedness measures on multi-label classification evaluation, *in* 'Proceed-

ings of the ACM International Conference on Image and Video Retrieval', CIVR '10, ACM, New York, NY, USA, pp. 303–310.

Obdrzálek, S. & Matas, J. (2003), Image retrieval using local compact dct-based representation., *in* B. Michaelis & G. Krell, eds, 'DAGM-Symposium', Vol. 2781 of *Lecture Notes in Computer Science*, Springer, pp. 490–497.

Oliva, A. & Torralba, A. B. (2001), 'Modeling the shape of the scene: A holistic representation of the spatial envelope.', *International Journal of Computer Vision* **42**(3), 145–175.

Parisi, G. (1988), *Statistical Field Theory*, Vol. 66 of *Frontiers in Physics*, Addison-Wesley.

Philbin, J., Chum, O., Isard, M., Sivic, J. & Zisserman, A. (2007), Object retrieval with large vocabularies and fast spatial matching, *in* 'Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on', pp. 1–8.

Philbin, J., Chum, O., Isard, M., Sivic, J. & Zisserman, A. (2008), Lost in quantization: Improving particular object retrieval in large scale image databases, *in* 'Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition'.

Pinheiro, Jos, C. & Bates, Douglas, M. (1996), 'Unconstrained parametrizations for variance-covariance matrices', *Statistics and Computing* **6**(3), 289–296.

Platt, J. C. (2000), 'Probabilistic outputs for support vector machines and comparison to regularized likelihood methods', *B Schölkopf C J C Burges A J Smola editors Advances in Kernel Methods Support Vector Learning* pp. 61–74.

Ponte, J. M. & Croft, W. B. (1998), A language modeling approach to information retrieval, *in* 'Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval', SIGIR '98, ACM, New York, NY, USA, pp. 275–281.

Robert, Christian, P. & Casella, G. (2005), *Monte Carlo Statistical Methods*, Springer Texts in Statistics, Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Roberts, G., O., Gelman, A. & Gilks, W., R. (1997), 'Weak convergence and optimal scaling of random walk Metropolis algorithms', *The Annals of Applied Probability* **7**(1), 110–120.

Roberts, G., O. & Stramer, O. (2003), 'Langevin diffusions and Metropolis-Hastings algorithms', *Methodology And Computing In Applied Probability* **4**, 337–357(21).

Robertson, S. E. & Jones, K. S. (1976), 'Relevance weighting of search terms', *J. Am. Soc. Inf. Sci.* **27**(3), 129–146.

Robertson, S. E. & Walker, S. (1994), Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval, *in* 'Proceedings of the 17th annual international ACM SIGIR conference on Research

and development in information retrieval', SIGIR '94, Springer-Verlag New York, Inc., New York, NY, USA, pp. 232–241.

Robertson, S. E. & Zaragoza, H. (2009), 'The probabilistic relevance framework: Bm25 and beyond', *Foundations and Trends in Information Retrieval* **3**, 333–389.

Russell, B. C., Torralba, A., Murphy, K. P. & Freeman, W. T. (2008), 'Labelme: A database and web-based tool for image annotation', *Int. J. Comput. Vision* **77**(1-3), 157–173.

Salembier, P. & Sikora, T. (2002), *Introduction to MPEG-7: Multimedia Content Description Interface*, John Wiley & Sons, Inc., New York, NY, USA.

Salton, G. & Buckley, C. (1988), 'Term-weighting approaches in automatic text retrieval', *Inf. Process. Manage.* **24**, 513–523.

Scott, D. W. (2001), 'Parametric statistical modeling by minimum integrated square error', *Technometrics* **43**(3), 274–285.

Sebastiani, F. (2002), 'Machine learning in automated text categorization', *ACM Comput. Surv.* **34**, 1–47.

Shi, J. & Malik, J. (1997), Normalized cuts and image segmentation, *in* 'Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)', CVPR '97, IEEE Computer Society, Washington, DC, USA, pp. 731–.

Sivic, J. & Zisserman, A. (2003), Video google: a text retrieval approach to object matching in videos, *in* 'ICCV, 2003', pp. 1470 –1477 vol.2.

Smeulders, A. W. M., Worring, M., Santini, S., Gupta, A. & Jain, R. (2000), 'Content-based image retrieval at the end of the early years', *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(12), 1349–1380.

Sparck-Jones, K. (1972), 'A statistical interpretation of term specificity and its application in retrieval', *Journal of Documentation* **28**(1), 11–20.

Stathopoulos, V. & Girolami, M. (2010), *Mixture Estimation and Applications*, Wiley series in probability and statistics, Wiley and Sons., chapter Manifold MCMC for Mixtures.

Stathopoulos, V. & Girolami, M. (2011), MCMC sampling of finite mixture and admixture models: when will geometry help?, Presented at the 3rd Workshop on Geometric and Algebraic Statistics (WOGAS3), The University of Warwick, Warwick, UK.

Stathopoulos, V. & Jose, J. (2009), Bayesian mixture hierarchies for automatic image annotation, *in* 'European Conference on Information Retrieval', Springer.

Stathopoulos, V. & Jose, J. (2011), 'Bayesian probabilistic models for image retrieval', *To appear in Journal of Machine Learning Research - Proceedings Track* **?**, ?–?

Titterington, D., M., Smith, A., F. & Makov, U., E. (1985), *Statistical Anal-*

ysis of Finite Mixture Distributions, Wiley series in probability and statistics, Wiley.

Turnbull, D., Barrington, L., Torres, D. & Lanckriet, G. (2008), 'Semantic annotation and retrieval of music and sound effects', *Audio, Speech, and Language Processing, IEEE Transactions on* **16**(2), 467 –476.

Tuytelaars, T. (2010), Dense interest points, *in* 'Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on', pp. 2281 –2288.

Ueda, N. & Nakano, R. (1998), 'Deterministic annealing em algorithm', *Neural Networks* **11**(2), 271 – 282.

van de Sande, K. E. A., Gevers, T. & Snoek, C. G. M. (2010), 'Evaluating color descriptors for object and scene recognition', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **32**(9), 1582–1596.

van Rijsbergen, C. J. (1979), *Information Retrieval*, Butterworths, London.

Vasconcelos, N. (2000), Bayesian Models for Visual Information Retrieval, PhD thesis, Massachusetts Institute of Technology, June.

Vasconcelos, N. (2001), Image indexing with mixture hierarchies, *in* 'CVPR '01: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition', pp. 3–10.

Vasconcelos, N. & Lippman, A. (1998), Learning mixture hierarchies, *in* 'Proceedings of Neural Information Processing Systems 11', MIT Press, Cambridge, MA, USA, pp. 606–612.

Vasconcelos, N. & Lippman, A. (2000), A probabilistic architecture for content-based image retrieval, *in* 'CVPR '00, Proceedings of the 2000 IEEE Conference on Computer Vision and Pattern Recognition', Vol. 1, pp. 216–221.

Weber, R., Schek, H.-J. & Blott, S. (1998), A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces, *in* 'Proceedings of the 24rd International Conference on Very Large Data Bases', VLDB '98, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 194–205.

Westerveld, T. & de Vries, A. P. (2003), Experimental result analysis for a generative probabilistic image retrieval model, *in* 'SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval', ACM, New York, NY, USA, pp. 135–142.

Westerveld, T., de Vries, A. P., van Ballegooij, A., de Jong, F. & Hiemstra, D. (2003), 'A probabilistic multimedia retrieval model and its evaluation', *EURASIP: Journal of Applied Signal Processing* **2003**(1), 186–198.

Williams, P. M. (1995), 'Bayesian regularization and pruning using a laplace prior', *Neural Comput.* **7**, 117–143.

Yavlinsky, A., Schofield, E. J. & Rüger, S. (2005), Automated image annotation using global features and robust nonparametric density estimation, *in* 'Proceedings of the 4th International Conference on Image and Video Re-

trieval (CIVR)', Vol. 3568 of *Lecture Notes in Computer Science (LNCS)*, Springer-Verlag, Singapore, pp. 507–517.

Zaragoza, H., Hiemstra, D. & Tipping, M. (2003), Bayesian extension to the language model for ad hoc information retrieval, *in* 'Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval', SIGIR '03, ACM, New York, NY, USA, pp. 4–9.

Zhai, C. & Lafferty, J. (2001), A study of smoothing methods for language models applied to ad hoc information retrieval, *in* 'Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval', SIGIR '01, ACM, New York, NY, USA, pp. 334–342.

Zhu, L., Rao, A. B. & Zhang, A. (2002), 'Theory of keyblock-based image retrieval', *ACM Trans. Inf. Syst.* **20**, 224–257.