TECHNICAL REPORT TR-2001-15

UNIVERSITY OF BRITISH COLUMBIA

Bayesian Latent Semantic Analysis of Multimedia Databases

Nando de Freitas †   Kobus Barnard ‡

October 22, 2001

† Dept. of Computer Science, University of British Columbia
2366 Main Mall, Vancouver, B.C. Canada V6T 1Z4

‡ UC Berkeley, Computer Science Division
387 Soda Hall, Berkeley, CA 94720-1776 USA

Abstract

We present a Bayesian mixture model for probabilistic latent semantic analysis of documents with images and text. The Bayesian perspective allows us to perform automatic regularisation to obtain sparser and more coherent clustering models. It also enables us to encode *a priori* knowledge, such as word and image preferences. The learnt model can be used for browsing digital databases, information retrieval with image and/or text queries, image annotation (adding words to an image) and text illustration (adding images to a text).

1 Introduction

Recent probabilistic latent semantic analysis (PLSA) models for the analysis of text (Hofmann 1999) and text and images (Barnard and Forsyth 2001) have led to many exciting developments in computer vision, image understanding, multimedia database browsing, and document retrieval. Here, we extend one of the proposed models, namely a flat mixture model, by adopting the Bayesian paradigm and by allowing other categorical and/or continuous variables. (We restrict ourselves to this simple mixture model only for tutorial purposes. The Bayesian treatment of aspect an hierarchical models is presented in (de Freitas and Barnard 2001).) This paradigm allows us to encode preferences for application specific words and images. It also allows us to perform regularisation according to the principle of Ockham's Razor. That is, if two models describe the data reasonably well, the simpler model is automatically chosen.

2 Model Specification

Let $\boldsymbol{\mathcal{X}} \triangleq (\mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_{n_x})$ denote a collection of n_x documents in a heterogeneous database[1]. Each document $\mathbf{x}_i$ is assumed to have n_a different attributes, $\mathbf{x}_i = (\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,a}, \ldots, \mathbf{x}_{i,n_a})$. Attributes may be categorical or continuous. Examples of categorical attributes include standard document meta-data such as movie ratings, media type or word presence (does the document contain a specific word?). Features derived from images and other multimedia signals are typically continuous-valued attributes. (It should be noted that our database is non-relational in that relations such as "next to" are not treated.) In general, the goal of the analysis will be to group documents into homogeneous classes in a probabilistic way. To accomplish this, each document $\mathbf{x}_i$ is assumed to be drawn from the following finite mixture model

$$\mathbf{x}_i|\boldsymbol{\varphi} \overset{iid}{\sim} \sum_{c=1}^{n_c} \lambda_c p(\mathbf{x}_i|\boldsymbol{\theta}_c) = \sum_{c=1}^{n_c} \lambda_c \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c}), \tag{1}$$

where $\boldsymbol{\varphi} \triangleq (\boldsymbol{\lambda}, \boldsymbol{\theta})$ encompasses all the model parameters, $\boldsymbol{\lambda}$ denotes the mixing weights, $\boldsymbol{\theta}$ denotes the parameters of the mixture component densities, and n_c denotes the number of components. The mixture model is defined on the standard probability simplex $\{\boldsymbol{\lambda} : \lambda_c \geq 0 \text{ for all } c \text{ and } \sum_{c=1}^{n_c} \lambda_c = 1\}$. Generating a document involves two steps. First, we select

[1]NOTATION: We use boldface Roman letters to refer to collections of items and italic Roman letters to refer to individual items. For example, a single word is written as w_j, while a group of n_w words is written as $\mathbf{w} \triangleq \mathbf{w}_{1:n_w} \triangleq (w_1, \ldots, w_{n_w})$. For simplicity, we use $\mathbf{x}_t$ to denote both the random variable and its realisation.

a mixture component with probability λ_c and, then, we sample the document from this mixture component.

From a modelling and inference perspective, it is often convenient to introduce the latent allocation variables $z_i \in \{1, \ldots, n_c\}$ to indicate that a particular document $\mathbf{x}_i$ belongs to a specific group c. These indicator variables $\{z_i; i = 1, \ldots, n_x\}$ correspond to an i.i.d. sample from the distribution

$$p(z_i = c) = \lambda_c \qquad\qquad \text{independently for } c = 1, \ldots, n_c.$$

One can now express the mixture model in a different form that will be useful for our subsequent derivations

$$p(\mathbf{x}_i, z_i = c|\boldsymbol{\varphi}) = \prod_{c=1}^{n_c} \left[\lambda_c \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c}) \right]^{\mathbb{I}_c(z_i)}, \tag{3}$$

where $\mathbb{I}_c(z_i) = 1$ if $\mathbf{x}_i$ belongs to group c and $\mathbb{I}_c(z_i) = 0$ otherwise. Note that marginalising over z_i in equation (3) allows us to recover the expression given by equation (1). The learning problem under consideration is either known as supervised learning (when the allocation variables are known) or as unsupervised learning (otherwise).

Under our modelling assumptions, the likelihood function for the entire data set is given by

$$p(\mathbf{x}|\boldsymbol{\varphi}) = \prod_{i=1}^{n_x} \sum_{c=1}^{n_c} \lambda_c \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c}) = \sum_{c_1=1}^{n_c} \cdots \sum_{c_{n_x}=1}^{n_c} \prod_{i=1}^{n_x} \lambda_{c_i} \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c_i}). \tag{4}$$

In the following subsections we describe the various forms that this likelihood can take depending on the type of attributes.

2.0.1 Continuous attributes

The algorithms developed in Section 3 apply to any type of mixture component density belonging to the exponential family. However, we will concentrate on Gaussian and multinomial densities.

In this paper, Gaussian attributes arise because we treat various image features (colour, texture, etc.) as samples from Gaussian distributions (Barnard and Forsyth 2001, Belongie, Carson, Greenspan and Malik 1997). That is, each instance of $x_{i,a}$ is an n_{g_a}-dimensional vector drawn from the normal distribution, $x_{i,a} \sim \mathcal{N}_{n_{g_a}}(\boldsymbol{\mu}_{a,c}, \boldsymbol{\Sigma}_{a,c})$, with density

$$p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c}) = |2\pi\boldsymbol{\Sigma}_{a,c}|^{-1/2} \exp\left(-\frac{1}{2} \left(\mathbf{x}_{i,a} - \boldsymbol{\mu}_{a,c}\right)' \boldsymbol{\Sigma}_{a,c}^{-1} \left(\mathbf{x}_{i,a} - \boldsymbol{\mu}_{a,c}\right) \right), \tag{5}$$

where the parameter vector contains the means and covariances $\boldsymbol{\theta}_{a,c} = (\boldsymbol{\mu}_{a,c}, \boldsymbol{\Sigma}_{a,c})$. If, for example, we have a mixture model with n_a Gaussian attributes, we can represent it as follows

$$\mathbf{x}_i|\boldsymbol{\varphi} \overset{iid}{\sim} \sum_{c=1}^{n_c} \lambda_c \prod_{a=1}^{n_a} \mathcal{N}_{n_{g_a}}(\boldsymbol{\mu}_{a,c}, \boldsymbol{\Sigma}_{a,c}).$$

This mixture of Gaussian products can be expressed as a mixture of multivariate Gaussians of size $\sum_a n_{g_a}$ with a block diagonal covariance. In general, the structure of this covariance depends on how we partition the Gaussian attributes. As the dimension of the vectors increases so does the number of covariance parameters that we have to estimate. As suggested in (Hunt and Jorgensen 1999), one could start with a diagonal covariance (assume independence) to estimate the parameters, assign observations to the clusters and then study the within cluster correlation matrices. Variables that are highly correlated may then be grouped into a subvector and the estimation procedure is repeated.

2.0.2 Categorical attributes

Let us assume that the a-th attribute can take n_v discrete values, more precisely let $\mathbf{x}_{i,a} \in \{1, \ldots, j, \ldots, n_v\}$. The probability of the j-th value of $\mathbf{x}_{i,a}$ occurring is defined as $\delta_{a,c,j} \triangleq p(x_{i,a} = j|\boldsymbol{\theta}_{a,c})$, with $\sum_j \delta_{a,c,j} = 1$. We can model the distribution of this variable using the multinomial model

$$p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c}) = \prod_{j=1}^{n_v} \delta_{a,c,j}^{\mathbb{I}_j(\mathbf{x}_{i,a})}. \tag{7}$$

Thus, in the categorical case, the parameter vector is given by $\boldsymbol{\theta}_{a,c} = \boldsymbol{\delta}_{a,c} = (\delta_{a,c,1}, \ldots, \delta_{a,c,n_v})$. Note that we only need $n_v - 1$ parameters because of the normalisation condition. If we have a mixture with several categorical variables, it can be expressed as

$$\mathbf{x}_i|\boldsymbol{\varphi} \overset{iid}{\sim} \sum_{c=1}^{n_c} \lambda_c \prod_{a=1}^{n_a} \prod_{j=1}^{n_v} \delta_{a,c,j}^{\mathbb{I}_j(\mathbf{x}_{i,a})}. \tag{8}$$

If we have several categorical variables, we can either assume that some of them are independent

$$p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c}) = \prod_{j=1}^{n_{v_1}} \epsilon_{a,c,j}^{\mathbb{I}_j(\mathbf{x}_{i,a})} \prod_{k=1}^{n_{v_2}} \nu_{a,c,k}^{\mathbb{I}_k(\mathbf{x}_{i,a})},$$

or we can allow them to covary so as to take into account their correlations *a priori*

$$p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c}) = \prod_{j=1}^{n_{v_1} \times n_{v_2}} \delta_{a,c,j}^{\mathbb{I}_j(\mathbf{x}_{i,a})}.$$

This, of course, introduces extra parameters and poses the same trade-offs that we discussed in the context of Gaussian attributes in Section 2.0.1.

Categorical variables abound in text and multimedia applications. In information filtering, we can use categorical variables to model document features such as ratings (how much do people like the movie discussed in this document?) or indicator variables (does a particular object appear in the document?). In text mining tasks, typically arising in language modelling, machine translation and information retrieval, we can use categorical variables $x_{i,a}$ to model how frequently a word a appears in a document $\mathbf{x}_i$ (Beeferman, Berger and Lafferty 1999, Hofmann and Puzicha 1998, Nigam, McCallum, Thrun and Mitchell 2000). Similarly, we can extend this analysis to model documents with hypertext links in the World-Wide Web (Cohn and Hofmann 2001).

In the text scenario, we often assume that the data is available as a co-occurrence table of word counts $\mathbf{N}$, where $n_{i,a} \triangleq [\mathbf{N}]_{i,a}$ denotes the number of times word a appears in document $\mathbf{x}_i$. For computational simplicity, we model documents by adopting the standard bag of words model. That is, we ignore word order and other contextual information. This is a standard naive Bayes assumption whereby, given a cluster, each word is assumed to be independent of the remaining words. The cluster variable, therefore, models the correlation between the words. From these assumptions, each text document is distributed according to

$$\mathbf{x}_i \overset{iid}{\sim} \sum_{c=1}^{n_c} \lambda_c \prod_{a=1}^{n_a} \delta_{a,c}^{n_{i,a}}, \tag{9}$$

where n_a, in this case, denotes the number of words in the vocabulary (dictionary), $n_{i,:} = \sum_a n_{i,a}$ denotes the total number of words in document $\mathbf{x}_i$, and $\delta_{a,c} \triangleq p(x_{i,a}|\boldsymbol{\theta}_{a,c})$ denotes the probability of each word a in cluster c, with $\sum_a \delta_{a,c} = 1$. For notational simplicity, we have ignored the normalisation factor of the multinomial density (we assume that the document length is class-independent).

One can extend the text model by incorporating links into the table or word counts $\mathbf{N}$ (Cohn and Hofmann 2001). This information is of great relevance in the design of search engines (Brin and Page 1998, Kleinberg 1998). In (Cohn and Hofmann 2001), the text and links are weighted by the heuristic constants α and $(1-\alpha)$. In the Bayesian framework, this weighting is performed by the prior and can be automatically computed using the data.

2.0.3 Mixed attributes

It is natural to combine categorical and continuous attributes in multimedia applications. We may cluster documents with words and images by combining a multinomial model for the words and a Gaussian model for the image features (Barnard and Forsyth 2001)

$$\mathbf{x}_i \overset{iid}{\sim} \sum_{c=1}^{n_c} \lambda_c \prod_{a_w=1}^{n_{aw}} \delta_{a_w,c}^{n_{i,a_w}} \prod_{a_g=1}^{n_{ag}} \left|2\pi\boldsymbol{\Sigma}_{a_g,c}\right|^{\frac{-1}{2}} \exp\left(-\frac{1}{2}\left(\mathbf{x}_{i,a_g}-\boldsymbol{\mu}_{a_g,c}\right)'\boldsymbol{\Sigma}_{a_g,c}^{-1}\left(\mathbf{x}_{i,a_g}-\boldsymbol{\mu}_{a_g,c}\right)\right)$$

where, in this case, $n_a = n_{aw} + n_{ag}$ denotes the total number of attributes. Note that the Gaussian attributes can have different dimension.

Note that a particular attribute a can correspond to a mixture distribution. For example, conditionally Gaussian attributes, also known as location-scale models (Chang and Afifi 1974, Krzanowski 1983, Hunt and Jorgensen 1999, Olkin and Tate 1961), are often used to model within-cluster associations between a discrete variable and several continuous variables. Here, each instance of $x_{i,a}$ is an $n_{g_a}+1$-dimensional vector, where the first n_{g_a} entries are drawn from the normal distribution conditionally on a categorical variable ι (last entry). In mathematical terms, one has $x_{i,a} \sim \mathcal{N}_{n_{g_a}}(\boldsymbol{\mu}_{a,c}(\iota), \boldsymbol{\Sigma}_{a,c}(\iota))$. The categorical variable is assumed to be multinomially distributed.

2.1 Prior specification

Our Bayesian extension of the maximum likelihood approach is justified by the following points

1. Ill-conditioning: In the maximum likelihood framework, the likelihood is often unbounded. For example, when dealing with mixtures of Gaussians, nothing prevents a mixture component density from being assigned to a single observation. When this happens, the variance goes to zero and the likelihood goes to infinity, thus causing serious ill-conditioning problems. To circumvent this common problem, people either prune components by hand or add extra tuning parameters as in ridge regression (Marquardt and Snee 1975). From a Bayesian perspective, the introduction of a prior reduces this problem.

2. A priori knowledge: One can use the prior to specify domain-specific knowledge (some rules derived from an expert) or subjective preferences (favouring simpler models).

3. Regularisation: Since the data set is finite and noisy, one needs to take care of not overfitting the data. We will show later that the prior distribution can be used to favour simpler (smooth) models that avoid fitting the noise and, therefore, extrapolate reasonable well.

4. Multiple overlapping copies of clusters: ML estimation often splits an underlying category into several components with identical parameters whose component weights λ_c add up to the correct one. Bayesian estimation avoids this problem by specifying priors that favour sparse models.

5. Starting point for more sophisticated modelling: The Bayesian perspective lays the groundwork for more sophisticated models that enable us to, for example, achieve robustness with respect to the specification of the prior distributions (no parameter

tuning), perform model selection, extend point estimators to average estimators and consider different loss functions in a principled way: see for example (Andrieu, de Freitas and Doucet 2000, Bernardo and Smith 1994, Stephens 1997).

We follow a hierarchical Bayesian strategy, where the unknown parameters $\boldsymbol{\varphi} = \{\boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\delta}\}$ and the allocation variables $\mathbf{z}$ are regarded as being drawn from appropriate prior distributions. We often acknowledge our uncertainty about the exact form of the prior by specifying it in terms of some unknown parameters (hyperparameters). These hyperparameters are, in turn, assumed to be drawn from appropriate hyperpriors. The idea of this hierarchical approach is that by increasing the levels of inference, we can make the higher level priors increasingly more diffuse. That is, we avoid having to specify too many parameters and, therefore, are more likely to obtain results that are independent of parameter tuning.

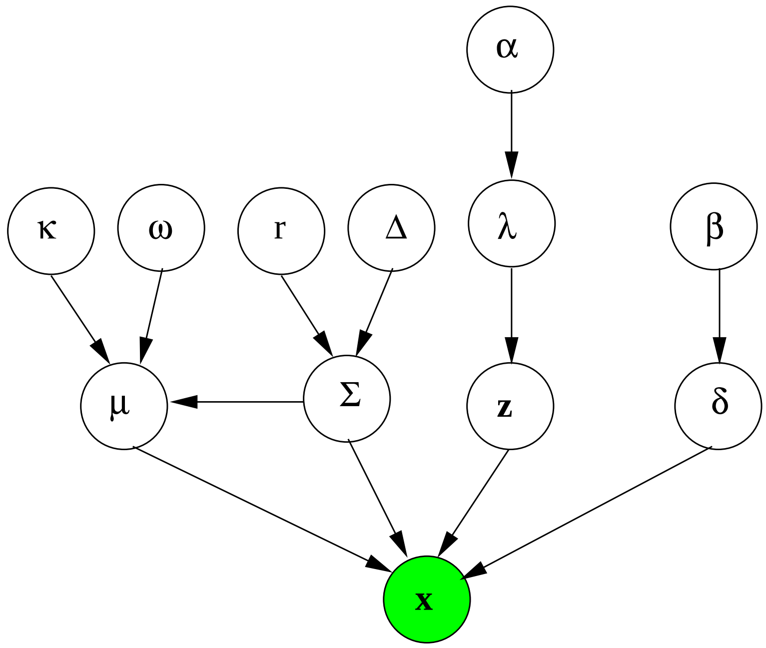

Figure 1: Directed probabilistic graphical model for our LSA model.

As shown in Figure 1, our hierarchical Bayesian model has the following three levels of inference

$$\begin{aligned} \text{Level I:} \qquad & p(\mathbf{x}|\boldsymbol{\varphi}, \mathbf{z}) = \prod_{i=1}^{n_x} p(\mathbf{x}_i|\boldsymbol{\varphi}, \mathbf{z}_i) \\ \text{Level II:} \qquad & p(\boldsymbol{\varphi}, \mathbf{z}|\boldsymbol{\eta}) = \prod_{i=1}^{n_x} p(\boldsymbol{\varphi}, \mathbf{z}_i|\boldsymbol{\eta}) \\ \text{Level III:} \qquad & p(\boldsymbol{\eta}) \end{aligned}$$

where, in the case of mixtures with Gaussian and discrete components, the parameters are $\boldsymbol{\theta} = (\boldsymbol{\lambda}, \boldsymbol{\delta}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$, and the hyperparameters are $\boldsymbol{\eta} = (\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\omega}, \kappa, r, \boldsymbol{\Delta})$. We discuss our choice of prior and hyperprior models in the following subsections.

2.1.1 Priors on the mixing variables and categorical parameters

The allocation variables z_i are assumed to be drawn from a multinomial distribution, $z_i \sim \mathcal{M}_{n_c}(1; \boldsymbol{\lambda})$, which admits the density

$$p(z_i|\boldsymbol{\lambda}) = \prod_{c=1}^{n_c} \lambda_c^{\mathbb{I}_c(z_i)}.$$

We place a conjugate Dirichlet prior on the mixing coefficients $\boldsymbol{\lambda} \sim \mathcal{D}_{n_c}(\boldsymbol{\alpha})$, having the following density

$$p(\boldsymbol{\lambda}|\boldsymbol{\alpha}) = \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1)\cdots\Gamma(\alpha_{n_c})} \prod_{c=1}^{n_c} \lambda_c^{\alpha_c - 1} \mathbb{I}_{\{\sum_c \lambda_c = 1\}}, \tag{11}$$

where $\Gamma(\cdot)$ denotes the Gamma function and $\alpha_0 = \sum_c \alpha_c$. Similarly, we place a Dirichlet prior distribution on each $\delta_{a,c}$, and assume that these priors are independent

$$p(\boldsymbol{\delta}|\boldsymbol{\beta}) = \prod_{c=1}^{n_c}\prod_{a=1}^{n_a} \frac{\Gamma(\beta_{a,c,0})}{\Gamma(\beta_{a,c,1})\cdots\Gamma(\beta_{a,c,n_v})} \prod_{j=1}^{n_v} \delta_{a,c,j}^{\beta_{a,c,j}-1} \mathbb{I}_{\{\sum_j \delta_{a,c,j} = 1\}}. \tag{12}$$

For example, in the text mining application, we can place a single Dirichlet prior over the word probabilities $\delta_{a,c}$ as follows

$$p(\boldsymbol{\delta}|\boldsymbol{\beta}) = \prod_{c=1}^{n_c} \frac{\Gamma(\beta_{0,c})}{\Gamma(\beta_{1,c})\ldots\Gamma(\beta_{n_a,c})} \prod_{a=1}^{n_a} \delta_{a,c}^{\beta_{a,c}-1} \tag{13}$$

To limit the computational and storage cost, it is often reasonable to set all the hyper-parameters to the same value $\beta_{a,c,n_v} = \beta$.

2.1.2 Priors on the Gaussian parameters

When considering multivariate normal distributions, we adopt the following normal-inverse Wishart prior (Bensmail, Celeux, Raftery and Robert 1997, Diebolt and Robert 1994, McLachlan and Peel 2000)

$$\boldsymbol{\mu}_{a,c} \sim \mathcal{N}_{n_{ga}}(\boldsymbol{\omega}_{a,c}, \boldsymbol{\Sigma}_{a,c}/\kappa_{a,c})$$
$$\boldsymbol{\Sigma}_{a,c}^{-1} \sim \mathcal{W}_{n_{ga}}(r_{a,c}, \boldsymbol{\Delta}_{a,c})$$

where $\mathcal{W}_{n_{ga}}(r_{a,c}, \boldsymbol{\Delta}_{a,c})$ denotes a Wishart distribution, with density

$$p(\boldsymbol{\Sigma}_{a,c}^{-1}|r_{a,c}, \boldsymbol{\Delta}_{a,c}) = \frac{|\boldsymbol{\Sigma}_{a,c}|^{-\frac{1}{2}(r_{a,c}-n_{ga}-1)} \exp\left[-\frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Sigma}_{a,c}^{-1}\boldsymbol{\Delta}_{a,c}^{-1}\right)\right]}{2^{\frac{1}{2}r_{a,c}n_{ga}}\pi^{\frac{1}{4}n_{ga}(1-n_{ga})}|\boldsymbol{\Delta}_{a,c}|^{\frac{1}{2}r_{a,c}}\prod_{l=1}^{n_{ga}}\Gamma(\frac{1}{2}(r_{a,c}-l+1))}$$

In the above expressions, $\mathbf{\Delta}_{a,c}$ is a symmetric, positive definite, $n_{g_a} \times n_{g_a}$ matrix, while $\kappa_{a,c}$ and $r_{a,c}$ are regularisation parameters, with $r_{a,c} > n_{g_a}$. In the univariate case, the Wishart distribution reduces to a Gamma distribution $\mathcal{G}a\left(\frac{r_{a,c}}{2}, \frac{\Delta_{a,c}^{-1}}{2}\right)$. Lastly, homoscedastic normal components are handled simply by imposing the condition $\mathbf{\Sigma}_{a,c} = \mathbf{\Sigma}_a$ for all $c \in \{1, \ldots, n_c\}$.

2.1.3 Hyperparameters, regularisation, empirical Bayes and model selection

The marginal posterior probability of the parameters $p(\boldsymbol{\varphi}, \mathbf{z}|\mathbf{x})$ and the marginal likelihood $p(\mathbf{x}|\boldsymbol{\eta})$ are two important distributions arising in Bayesian statistics. They are given in terms of the following integrals (where, for notational simplicity, we have introduced the parameter vector $\boldsymbol{\phi} \triangleq (\boldsymbol{\varphi}, \mathbf{z})$)

$$p(\boldsymbol{\phi}|\mathbf{x}) = \int p(\boldsymbol{\phi}|\boldsymbol{\eta}, \mathbf{x}) p(\boldsymbol{\eta}|\mathbf{x}) d\boldsymbol{\eta} \tag{16}$$

$$p(\mathbf{x}|\boldsymbol{\eta}) = \int p(\mathbf{x}|\boldsymbol{\phi}, \boldsymbol{\eta}) p(\boldsymbol{\phi}|\boldsymbol{\eta}) d\boldsymbol{\phi}. \tag{17}$$

The marginal posterior is needed for computing parameters, such as the probability of a particular word within a cluster given the document corpus. The marginal likelihood plays a fundamental role in model selection. In particular, when comparing two model hypotheses ($\mathcal{H}_1$ and $\mathcal{H}_2$), we need to compute the ratio of the marginal likelihoods (known as the Bayes factor)

$$\mathcal{B}_{1,2} = \frac{\int p(\mathbf{x}|\boldsymbol{\phi}, \mathcal{H}_1) p(\boldsymbol{\phi}|\mathcal{H}_1) d\boldsymbol{\phi}}{\int p(\mathbf{x}|\boldsymbol{\phi}, \mathcal{H}_2) p(\boldsymbol{\phi}|\mathcal{H}_2) d\boldsymbol{\phi}}.$$

Intuitively, this ratio provides a measure of whether the data has increased or decreased the odds of one model with respect to the other. In our case, different model hypothesis correspond to different sets of hyperparameters $\boldsymbol{\eta}$. Notice that the two marginals are related via Bayes rule

$$p(\boldsymbol{\phi}|\boldsymbol{\eta}, \mathbf{x}) = \frac{p(\mathbf{x}|\boldsymbol{\phi}) p(\boldsymbol{\phi}|\boldsymbol{\eta})}{p(\mathbf{x}|\boldsymbol{\eta})} \tag{18}$$

$$p(\boldsymbol{\eta}|\mathbf{x}) = \frac{p(\mathbf{x}|\boldsymbol{\eta}) p(\boldsymbol{\eta})}{p(\mathbf{x})}. \tag{19}$$

That is, the likelihood at the hyperparameter inference level becomes the normalising distribution (evidence) at the parameter inference level.

A rigorous Bayesian analysis would involve specifying the priors on the hyperparameters. This would require that we develop computationally demanding estimation algorithms such as variational methods or Markov chain Monte Carlo simulation. Here, we opt for more pragmatic solutions. We either choose the hyperparameters based on our *a priori* preferences or use the data to estimate a point estimate of the hyperparameters. In the latter case, we aim to find an $\boldsymbol{\eta}^{\star}$ that maximises $p(\mathbf{x}|\boldsymbol{\eta})$. That is, we are trying to find the most likely model hypothesis. This approach of estimating the priors from the data is an empirical Bayes method known as maximum likelihood type II (Carlin and Louis 2000, Good 1983). It is based on the assumption that $p(\boldsymbol{\eta}|\mathbf{x})$ is fairly sharply peaked around its mode $\boldsymbol{\eta}^\star$ and, consequently, approximations such as

$$p(\boldsymbol{\phi}|\mathbf{x}) \approx p(\boldsymbol{\phi}|\boldsymbol{\eta}^\star, \mathbf{x}) \tag{20}$$

are valid. We describe an EM algorithm for carrying out the maximisation of the marginal likelihood in Section 3.3.

The Bayes Factor model selection approach has an inherent Ockham factor (regulariser). If several hypotheses explain the data reasonably, simpler hypotheses will be preferred, even if the prior probability does not favour the simple models (Gull 1988, Jefferys and Berger 1992, Mackay 1992). To illustrate this, consider a situation where we have to decide between

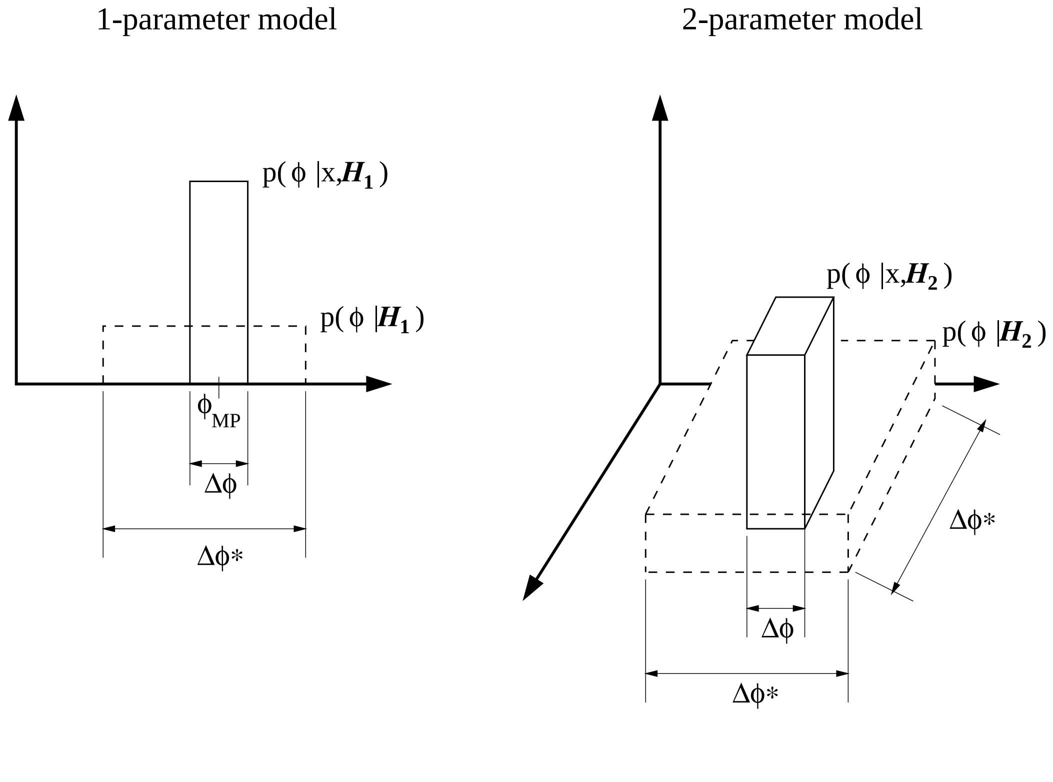

Figure 2: Comparing two models with uniform priors. The analysis shows that the Bayesian approach will favour the simpler model as long as it explains the data.

a 1-parameter ($\mathcal{H}_1$) and a 2-parameter model ($\mathcal{H}_2$). We choose uniform priors

$$p(\boldsymbol{\phi}|\mathcal{H}_1) = \frac{1}{\Delta\boldsymbol{\phi}^*} \qquad \text{and} \qquad p(\boldsymbol{\phi}|\mathcal{H}_2) = \frac{1}{\Delta\boldsymbol{\phi}^*\Delta\boldsymbol{\phi}^*}$$

as shown in Figure 2. Let us assume that, around a very probable point $\boldsymbol{\phi}_{MP}$, the posteriors can be approximated as

$$p(\boldsymbol{\phi}_{MP}|\mathbf{x}, \mathcal{H}_1) \approx \frac{1}{\Delta\boldsymbol{\phi}} \qquad \text{and} \qquad p(\boldsymbol{\phi}_{MP}|\mathbf{x}, \mathcal{H}_2) \approx \frac{1}{\Delta\boldsymbol{\phi}\Delta\boldsymbol{\phi}}.$$

It follows from Bayes rule that the marginal likelihood for model $\mathcal{H}_1$ is

$$p(\mathbf{x}|\mathcal{H}_1) \approx p(\mathbf{x}|\boldsymbol{\phi}_{MP}, \mathcal{H}_1) p(\boldsymbol{\phi}_{MP}|\mathcal{H}_1) \Delta\boldsymbol{\phi}$$

and consequently the Bayes factor is

$$\mathcal{B}_{1,2} = \frac{p(\mathbf{x}|\mathcal{H}_1)}{p(\mathbf{x}|\mathcal{H}_2)} \approx \frac{p(\mathbf{x}|\boldsymbol{\phi}_{MP}, \mathcal{H}_1)}{p(\mathbf{x}|\boldsymbol{\phi}_{MP}, \mathcal{H}_2)} \frac{p(\boldsymbol{\phi}|\mathcal{H}_1)\Delta\boldsymbol{\phi}}{p(\boldsymbol{\phi}|\mathcal{H}_2)\Delta\boldsymbol{\phi}\Delta\boldsymbol{\phi}} = \frac{p(\mathbf{x}|\boldsymbol{\phi}_{MP}, \mathcal{H}_1)}{p(\mathbf{x}|\boldsymbol{\phi}_{MP}, \mathcal{H}_2)} \frac{\Delta\boldsymbol{\phi}^*}{\Delta\boldsymbol{\phi}}$$

Because $\Delta\boldsymbol{\phi}^* > \Delta\boldsymbol{\phi}$, we see that the Bayes factor is the ratio of the likelihoods times a factor that favours the simpler model. Even if we start with uniform distributions on the parameters, the Bayesian approach will still favour the simpler hypothesis as long as it explains the data.

In our mixture setting, if we initially guess more components than required, the Bayesian method penalises the extra components. We can take this argument further by considering the possibility of having an infinite number of components *a priori*. In the mixture model, if $\boldsymbol{\theta}_c$ is an element of $\boldsymbol{\Theta}$, it follows that $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_{n_c})$ may be interpreted as a probability distribution over $\boldsymbol{\Theta}$, with $\lambda_c = \Pr(\boldsymbol{\theta} = \boldsymbol{\theta}_c)$ for $c = 1, \ldots, n_c$ (Titterington, Smith and Makov 1985). If $G(\cdot)$ denotes the probability measure that puts mass λ_c at the support point $\boldsymbol{\theta}_c$, model (1) can be expressed in terms of the following Lebesgue-Stieltjes integral

$$p(\mathbf{x}_i|\boldsymbol{\varphi}) = \int_{\boldsymbol{\Theta}} p(\mathbf{x}_i|\boldsymbol{\theta}) dG(\boldsymbol{\theta}).$$

That is, we have and infinite number of components, but they are appropriately weighed by the prior $G(\boldsymbol{\theta})$. Notice that instead of sampling parameters, we could simply sample prior distributions (Dirichlet process).

2.2 Posterior distribution

The posterior distribution $p(\mathbf{z}, \boldsymbol{\varphi}|\mathbf{x})$ can be obtained by multiplying the prior and likelihood distributions and then normalising this product. This distribution is the solution to the inference problem in Bayesian statistics. From it, we can derive any estimates of interest, such as the mean, mode and high probability intervals. Since the data is i.i.d, the posterior can be factorised as follows

$$p(\mathbf{z}, \boldsymbol{\varphi}|\mathbf{x}) = \prod_{i=1}^{n_x} p(z_i, \boldsymbol{\varphi}|\mathbf{x}_i) = \prod_{i=1}^{n_x} p(z_i|\boldsymbol{\varphi}, \mathbf{x}_i) p(\boldsymbol{\varphi}).$$

Using Bayes rule, the first term on the right hand side can be expanded in terms of its likelihood and prior

$$p(z_i|\boldsymbol{\varphi}, \mathbf{x}_i) = \frac{p(\mathbf{x}_i|z_i, \boldsymbol{\varphi}) p(z_i|\boldsymbol{\varphi})}{p(\mathbf{x}_i|\boldsymbol{\varphi})} = \prod_{c=1}^{n_c} \left[\frac{\lambda_c \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c})}{\sum_{c'=1}^{n_c} \lambda_{c'} \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c'})} \right]^{\mathbb{I}_c(z_i)}.$$

Hence, we have $z_i|(\boldsymbol{\varphi}, \mathbf{x}_i) \sim \mathcal{M}_{n_c}(1; \boldsymbol{\xi}_i)$, where

$$\xi_{i,c} \triangleq \frac{\lambda_c \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c})}{\sum_{c'=1}^{n_c} \lambda_{c'} \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c'})}. \tag{23}$$

Using this result, the posterior distribution can be expressed in the general form

$$p(\mathbf{z}, \boldsymbol{\varphi}|\mathbf{x}) \propto \left[\prod_{c=1}^{n_c} \lambda_c^{\alpha_c - 1}\right]\left[\prod_{a=1}^{n_a}\prod_{c=1}^{n_c} p(\boldsymbol{\theta}_{a,c}|\cdot)\right]\left[\prod_{i=1}^{n_x}\prod_{c=1}^{n_c}\left[\frac{\lambda_c \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c})}{\sum_{c'=1}^{n_c} \lambda_{c'} \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c'})}\right]^{\mathbb{I}_c(z_i)}\right]. \tag{24}$$

In the following subsections, we derive expressions for the posterior in the specific cases of categorical and Gaussian attributes.

2.2.1 Posterior for categorical attributes

When considering a categorical attribute a, we can substitute the likelihood of equation (8) and the prior of equation (12) into equation (24) to obtain

$$p(\mathbf{z}, \boldsymbol{\varphi}|\mathbf{x}) \propto \left(\prod_{c=1}^{n_c} \lambda_c^{k_c + \alpha_c - 1}\right)\left(\prod_{c=1}^{n_c}\prod_{a=1}^{n_a}\prod_{j=1}^{n_v} \delta_{a,c,j}^{\sum_i \mathbb{I}_j(\mathbf{x}_{i,a})\mathbb{I}_c(z_i) + \beta_{a,c,j} - 1}\right),$$

where $k_c \triangleq \sum_{i=1}^{n_x} \mathbb{I}_c(z_i)$ denotes the total number of documents assigned to class c. According to this unnormalised posterior, one draws

$$\boldsymbol{\lambda}|(\mathbf{z}, \mathbf{x}, \boldsymbol{\alpha}) \sim \mathcal{D}_{n_c}\left(k_1 + \alpha_1, \ldots, k_{n_c} + \alpha_{n_c}\right) \tag{26}$$

and for each cluster c and each attribute a, one draws

$$\boldsymbol{\delta}_{a,c}|(\mathbf{z}, \mathbf{x}, \boldsymbol{\beta}) \sim \mathcal{D}_{n_v}\left(\sum_i \mathbb{I}_1(\mathbf{x}_{i,a})\mathbb{I}_c(z_i) + \beta_{a,c,1}, \ldots, \sum_i \mathbb{I}_{n_v}(\mathbf{x}_{i,a})\mathbb{I}_c(z_i) + \beta_{a,c,n_v}\right). \tag{27}$$

For example, in text mining, using the likelihood defined in terms of equation (9) and the prior of equation (13), we have

$$p(\mathbf{z}, \boldsymbol{\varphi}|\mathbf{x}) \propto \left(\prod_{c=1}^{n_c} \lambda_c^{k_c + \alpha_c - 1}\right)\left(\prod_{c=1}^{n_c}\prod_{a=1}^{n_a} \delta_{a,c}^{\sum_i n_{i,a}\mathbb{I}_c(z_i) + \beta_{a,c} - 1}\right).$$

Therefore, we may draw

$$\boldsymbol{\delta}_c|(\mathbf{z}, \mathbf{x}, \boldsymbol{\alpha}) \sim \mathcal{D}_{n_a}\left(\sum_i n_{i,1}\mathbb{I}_c(z_i) + \beta_{1,c}, \ldots, \sum_i n_{i,n_a}\mathbb{I}_c(z_i) + \beta_{n_a,c}\right). \tag{29}$$

2.2.2 Posterior for Gaussian attributes

By multiplying the priors of equation (14) and the likelihood of equation (5) and completing squares, one obtains the following conjugated posterior values for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$

$$\boldsymbol{\mu}_{a,c}|(\mathbf{z}, \mathbf{x}) \sim \mathcal{N}_{n_{ga}}\left(\boldsymbol{\omega}^\star_{a,c}, (k_c + \kappa_{a,c})^{-1}\boldsymbol{\Sigma}_{a,c}\right)$$

$$\boldsymbol{\Sigma}^{-1}_{a,c}|(\mathbf{z}, \mathbf{x}) \sim \mathcal{W}_{n_{ga}}\left(k_c + r_{a,c}, \boldsymbol{\Delta}^\star_{a,c}\right),$$

where the new expressions for the Gaussian mean $\boldsymbol{\omega}^{\star}_{a,c}$ and Wishart variance parameter $\boldsymbol{\Delta}^{\star}_{a,c}$ are

$$\begin{aligned}
\boldsymbol{\omega}^{\star}_{a,c} &= \left(k_c\overline{\mathbf{x}}_{a,c} + \kappa_{a,c}\boldsymbol{\omega}_{a,c}\right)\left(k_c + \kappa_{a,c}\right)^{-1} \\
\boldsymbol{\Delta}^{\star}_{a,c} &= \left(\boldsymbol{\Delta}^{-1}_{a,c} + k_c\mathbf{V}_{a,c} + \frac{k_c r_{a,c}}{k_c + r_{a,c}}(\overline{\mathbf{x}}_{a,c} - \boldsymbol{\omega}_{a,c})(\overline{\mathbf{x}}_{a,c} - \boldsymbol{\omega}_{a,c})'\right)^{-1},
\end{aligned}$$

with

$$\begin{aligned}
\overline{\mathbf{x}}_{a,c} &\triangleq \frac{1}{k_c}\sum_{i=1}^{n_x}\mathbb{I}_c(z_i)\mathbf{x}_{i,a} \\
\mathbf{V}_{a,c} &\triangleq \frac{1}{k_c}\sum_{i=1}^{n_x}\mathbb{I}_c(z_i)(\mathbf{x}_{i,a} - \overline{\mathbf{x}}_{a,c})(\mathbf{x}_{i,a} - \overline{\mathbf{x}}_{a,c})'.
\end{aligned}$$

3 Computation

The parameters of the mixture model cannot be computed analytically unless one knows the mixture indicator variables. As a result, we have to resort to numerical methods. In this section, we present EM algorithms (Baum, Petrie, Soules and Weiss 1970, Dempster, Laird and Rubin 1977) to compute the ML and MAP point estimates of the mixture model. Although the convergence rate of EM is linear and very slow if the mixture components are very close, it is very simple, easy to program and guaranteed to converge monotonically to a local maximum of the likelihood function $p(\mathbf{x}|\boldsymbol{\varphi})$ (Lindsay 1988, McLachlan and Peel 2000, Redner and Walker 1984). In particular, the design of EM algorithms is straightforward when the family of densities $p(\cdot|\boldsymbol{\varphi})$ possesses a sufficient statistic of fixed dimension $t(\mathbf{x})$ for the parameters $\boldsymbol{\varphi}$. When this is the case, $p(\mathbf{x}|\boldsymbol{\varphi})$ can be factored into two terms $p(\mathbf{x}|\boldsymbol{\varphi}) = p(\mathbf{x})p(\boldsymbol{\varphi}|t(\mathbf{x}))$ such that $p(\mathbf{x})$ is independent of $\boldsymbol{\varphi}$ and the kernel density $p(\boldsymbol{\varphi}|t(\mathbf{x}))$ only depends on $\mathbf{x}$ through the sufficient statistics. Among the distributions of interest, only the exponential families have this property. Lastly, we present an maximum likelihood type II algorithm to estimate the hyperparameters.

3.1 Maximum likelihood estimation with the EM algorithm

After initialisation, the EM algorithm for ML estimation iterates between the following two steps.

1. **E step:** Compute the expected value of the complete log-likelihood function with respect to the distribution of the allocation variables $\mathbf{Q}^{\mathrm{ML}} = \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\boldsymbol{\varphi}^{(\mathrm{old})})}\left[\log p(\mathbf{z},\mathbf{x}|\boldsymbol{\varphi})\right]$, where $\boldsymbol{\varphi}^{(\mathrm{old})}$ refers to the value of the parameters at the previous time step.

2. **M step:** Perform the following maximisation $\boldsymbol{\varphi}^{(\mathrm{new})} = \arg\max_{\boldsymbol{\varphi}} \mathbf{Q}^{\mathrm{ML}}$.

The $\mathbf{Q}^{\mathrm{ML}}$ function can be expanded as follows (see Appendix B)

$$\mathbf{Q}^{\mathrm{ML}} = \sum_{i=1}^{n_x}\sum_{c=1}^{n_c} p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\mathrm{old})}) \log\left[\lambda_c \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c})\right]. \tag{33}$$

In the E step, we have to compute $p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\mathrm{old})})$, which can be easily accomplished as follows

$$p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\mathrm{old})}) = \frac{p(z_i = c, \mathbf{x}_i|\boldsymbol{\varphi}^{(\mathrm{old})})}{\sum_{c'=1}^{n_c} p(z_i = c', \mathbf{x}_i|\boldsymbol{\varphi}^{(\mathrm{old})})} = \frac{\lambda_c \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c})}{\sum_{c'=1}^{n_c} \lambda_{c'} \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c'})} = \xi_{i,c}. \tag{34}$$

Hence, the E step for a model with a single categorical variable (a-th attribute) involves computing

$$\xi_{i,c} = \frac{\lambda_c \prod_{j=1}^{n_v} \delta_{a,c,j}^{\mathbb{I}_j(\mathbf{x}_{i,a})}}{\sum_{c'=1}^{n_c} \lambda_{c'} \prod_{j=1}^{n_v} \delta_{a,c',j}^{\mathbb{I}_j(\mathbf{x}_{i,a})}}. \tag{35}$$

The corresponding M step requires that we maximise $\mathbf{Q}^{\mathrm{ML}}$ subject to the constraints $\sum_{c=1}^{n_c} \lambda_c = 1$ and $\sum_{j=1}^{n_v} \delta_{a,c,j} = 1$. Appendix C.1 shows how to use Lagrange multipliers to accomplish this goal. The resulting expressions are

$$\lambda_c = \frac{1}{n_x}\sum_{i=1}^{n_x} \xi_{i,c} \qquad \text{and} \qquad \delta_{a,c,j} = \frac{\sum_{i=1}^{n_x} \mathbb{I}_j(\mathbf{x}_{i,a})\xi_{i,c}}{\sum_{i=1}^{n_x} \xi_{i,c}}. \tag{36}$$

In the text mining example, $\mathbf{Q}^{\mathrm{ML}}$ is given by

$$\mathbf{Q}^{\mathrm{ML}} = \sum_{i=1}^{n_x}\sum_{c=1}^{n_c} p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\mathrm{old})}) \log\left[\lambda_c \prod_{a=1}^{n_a} \delta_{a,c}^{n_{i,a}}\right]$$

Hence, under the constraint $\sum_{a=1}^{n_a} \delta_{a,c} = 1$, the EM derivation for text documents yields

$$\begin{aligned}
\xi_{i,c} &= \frac{\lambda_c \prod_{a=1}^{n_a} \delta_{a,c}^{n_{i,a}}}{\sum_{c'=1}^{n_c} \lambda_{c'} \prod_{a=1}^{n_a} \delta_{a,c'}^{n_{i,a}}} \\
\lambda_c &= \frac{1}{n_x}\sum_{i=1}^{n_x} \xi_{i,c} \\
\delta_{a,c} &= \frac{\sum_{i=1}^{n_x} n_{i,a}\xi_{i,c}}{\sum_{i=1}^{n_x} n_{i,:}\xi_{i,c}}
\end{aligned}$$

For Gaussian attributes, the E step is again a straightforward application of equation (34). It is also not hard to show that by maximising the $\mathbf{Q}^{\mathrm{ML}}$ function, under the constraint that probabilities add up to one, one obtains the following update equations in the M step (McLachlan and Peel 2000, Chapter 3)

$$\begin{aligned}
\boldsymbol{\mu}_{a,c} &= \mathbf{T}_{a,c,2} T_{a,c,1}^{-1} \\
\boldsymbol{\Sigma}_{a,c} &= \left(\mathbf{T}_{a,c,3} - T_{a,c,1}^{-1}\mathbf{T}_{a,c,2}\mathbf{T}'_{a,c,2}\right) T_{a,c,1}^{-1}
\end{aligned}$$

with

$$
\begin{aligned}
T_{a,c,1} &\triangleq \sum_{i=1}^{n_x} \xi_{i,c} \\
\mathbf{T}_{a,c,2} &\triangleq \sum_{i=1}^{n_x} \xi_{i,c}\mathbf{x}_{i,a} \\
\mathbf{T}_{a,c,3} &\triangleq \sum_{i=1}^{n_x} \xi_{i,c}\mathbf{x}_{i,a}\mathbf{x}'_{i,a}
\end{aligned}
$$

In the case of homoscedastic components, the update for the estimate of the common covariance is

$$
\mathbf{\Sigma}_a = \frac{1}{n_x}\sum_{c=1}^{n_c} T_{a,c,1}\mathbf{\Sigma}_{a,c} \tag{40}
$$

where $\mathbf{\Sigma}_{a,c}$ is as in equation (39). The remaining estimates stay as in the heteroscedastic case.

3.2 Maximum a posteriori estimation with the EM algorithm

The EM formulation for MAP estimation is straightforward. One simply has to augment the objective function in the M step, $\mathbf{Q}^{\mathrm{ML}}$, by adding to it the log prior densities. That is, the MAP objective function is

$$
\begin{aligned}
\mathbf{Q}^{\mathrm{MAP}} &= \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\boldsymbol{\varphi}^{(\mathrm{old})})}\left[\log p(\mathbf{z},\mathbf{x},\boldsymbol{\varphi})\right] \\
&= \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\boldsymbol{\varphi}^{(\mathrm{old})})}\left[\log p(\mathbf{z},\mathbf{x}|\boldsymbol{\varphi}) + \log p(\boldsymbol{\varphi})\right] \\
&= \mathbf{Q}^{\mathrm{ML}} + \log p(\boldsymbol{\varphi})
\end{aligned}
$$

This is done in Appendix C.2. The resulting expressions for a model with a single discrete attribute are

$$
\lambda_c = \frac{\sum_{i=1}^{n_x}\xi_{i,c} + \alpha_c - 1}{n_x + \sum_{c'}\alpha'_c - n_c} \qquad \text{and} \qquad \delta_{a,c,j} = \frac{\sum_{i=1}^{n_x}\mathbb{I}_j(\mathbf{x}_{i,a})\xi_{i,c} + \beta_{a,c,j} - 1}{\sum_{i=1}^{n_x}\xi_{i,c} + \sum_{j'}\beta_{a,c,j'} - n_v} \tag{41}
$$

and, in the particular case of text

$$
\delta_{a,c} = \frac{\sum_{i=1}^{n_x} n_{i,a}\xi_{i,c} + \beta_{a,c} - 1}{\sum_{i=1}^{n_x} n_{i,:}\xi_{i,c} + \sum_{a'}\beta_{a',c} - n_a} \tag{42}
$$

These expressions can also be derived by considering the posterior modes

$$
\begin{aligned}
Mode[\boldsymbol{\lambda}_c|\mathbf{z},\mathbf{x},\boldsymbol{\alpha}] &= \frac{k_c + \alpha_c - 1}{\sum_{c'}(k_{c'} + \alpha'_c) - n_c} = \frac{k_c + \alpha_c - 1}{n_x - n_c + \sum_{c'}\alpha'_c} \\
Mode[\boldsymbol{\delta}_{a,c,j}|\mathbf{z},\mathbf{x},\boldsymbol{\beta}] &= \frac{\sum_i \mathbb{I}_j(\mathbf{x}_{i,a})\mathbb{I}_c(z_i) + \beta_{a,c,j} - 1}{\sum_{j'}\left(\sum_i \mathbb{I}_{j'}(\mathbf{x}_{i,a})\mathbb{I}_c(z_i) + \beta_{a,c,j'}\right) - n_v} = \frac{\sum_i \mathbb{I}_j(\mathbf{x}_{i,a})\mathbb{I}_c(z_i) + \beta_{a,c,j} - 1}{k_c - n_v + \sum_{j'}\beta_{a,c,j'}} \\
Mode[\delta_{a,c}|\mathbf{z},\mathbf{x},\boldsymbol{\beta}] &= \frac{\sum_i n_{i,a}\mathbb{I}_c(z_i) + \beta_{a,c} - 1}{\sum_{a'}\left(\sum_i n_{i,a'}\mathbb{I}_c(z_i) + \beta_{a',c}\right) - n_a} = \frac{\sum_i n_{i,a}\mathbb{I}_c(z_i) + \beta_{a,c} - 1}{\sum_i n_{i,:}\mathbb{I}_c(z_i) + \sum_{a'}\beta_{a',c} - n_a}
\end{aligned}
$$

and by replacing the cluster indicator variable with its posterior expectation. The ML updates follow by setting the Dirichlet hyper-parameters to a uniform prior ($\boldsymbol{\alpha} \rightarrow \mathbf{1}$ and $\boldsymbol{\beta} \rightarrow \mathbf{1}$).

Similarly, in the Gaussian case, we can compute the MAP estimates by maximising the log-posterior to obtain

$$\boldsymbol{\mu}_{a,c} = \left(T_{a,c,2} + \kappa_{a,c}\boldsymbol{\omega}_{a,c}\right)\left(T_{a,c,1} + \kappa_{a,c}\right)^{-1}$$
$$\boldsymbol{\Sigma}_{a,c} = \left(T_{a,c,1} + r_{a,c} - n_{g_a}\right)^{-1}\left(\boldsymbol{\Delta}_{a,c}^{-1} + \widehat{\mathbf{V}}_{a,c} + \kappa_{a,c}(\boldsymbol{\omega}_{a,c} - \boldsymbol{\mu}_{a,c})(\boldsymbol{\omega}_{a,c} - \boldsymbol{\mu}_{a,c})'\right)$$

with

$$\widehat{\mathbf{V}}_{a,c} \triangleq \sum_{i=1}^{n_x} \xi_{i,c}(\mathbf{x}_{i,a} - \boldsymbol{\mu}_{a,c})(\mathbf{x}_{i,a} - \boldsymbol{\mu}_{a,c})'$$

By adopting uninformative priors with $\boldsymbol{\kappa} \rightarrow \mathbf{0}$, $r_{a,c} \rightarrow n_{g_a}$ and $\boldsymbol{\Delta}^{-1} \rightarrow \mathbf{0}$, the MAP estimates of $\boldsymbol{\mu}_{a,c}$ and $\boldsymbol{\Sigma}_{a,c}$ simplify to the estimates of equation (39).

3.3 Estimating the hyperparameters

In Appendix D, we derive the M step of an EM algorithm to maximise $p(\boldsymbol{\eta}|\mathbf{x})$ in the case of discrete distributions. It works by maximising the expected marginal log-likelihood given by

$$\begin{aligned}
\mathbf{Q}^{\mathrm{E}} &= \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\boldsymbol{\eta})}\left[\log p(\mathbf{z},\mathbf{x}|\boldsymbol{\eta})\right] \\
&= \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\boldsymbol{\eta})}\left[\log \int p(\mathbf{z},\mathbf{x}|\boldsymbol{\varphi})p(\boldsymbol{\varphi}|\boldsymbol{\eta})d\boldsymbol{\varphi}\right]
\end{aligned}$$

This results on a set of equations that needs to be iterated in order to increase the lower bound on the marginal likelihood

$$\alpha_c^{(\mathrm{new})} = \alpha_c \frac{\Psi\left(\sum_{i=1}^{n_x} \xi_{i,c} + \alpha_c\right) - \Psi(\alpha_c)}{\Psi\left(n_x + \sum_{c=1}^{n_c} \alpha_c\right) - \Psi\left(\sum_{c=1}^{n_c} \alpha_c\right)}$$

$$\beta_{a,c,j}^{(\mathrm{new})} = \beta_{a,c,j} \frac{\Psi\left(\sum_{i=1}^{n_x} \xi_{i,c}\mathbb{I}_j(\mathbf{x}_{i,a}) + \beta_{a,c,j}\right) - \Psi(\beta_{a,c,j})}{\Psi\left(\sum_{j=1}^{n_v}\left(\sum_{i=1}^{n_x} \xi_{i,c}\mathbb{I}_j(\mathbf{x}_{i,a}) + \beta_{a,c,j}\right)\right) - \Psi\left(\sum_{j=1}^{n_v} \beta_{a,c,j}\right)}$$

where $\Psi(\alpha_c) \triangleq \frac{\partial}{\partial \alpha_c} \log \Gamma(\alpha_c)$ is the digamma function. It is also possible to derive faster Newton-Raphson schemes as shown in Appendix D. However, one has to take care that this algorithm does not become numerically unstable. Lastly, it is also possible to derive estimators to compute the hyperparameters of the Gaussian components, as discussed in (Chen 1979, Gelman, Carlin, Stern and Rubin 1995).

4 Applications

4.1 Classification and information retrieval

An important facility for image and text databases, such as the World-Wide Web, is retrieval based on user queries. We wish to support queries based on text, image features, categorical variables or combinations of these, as depicted in Figure 3. We also would like the queries

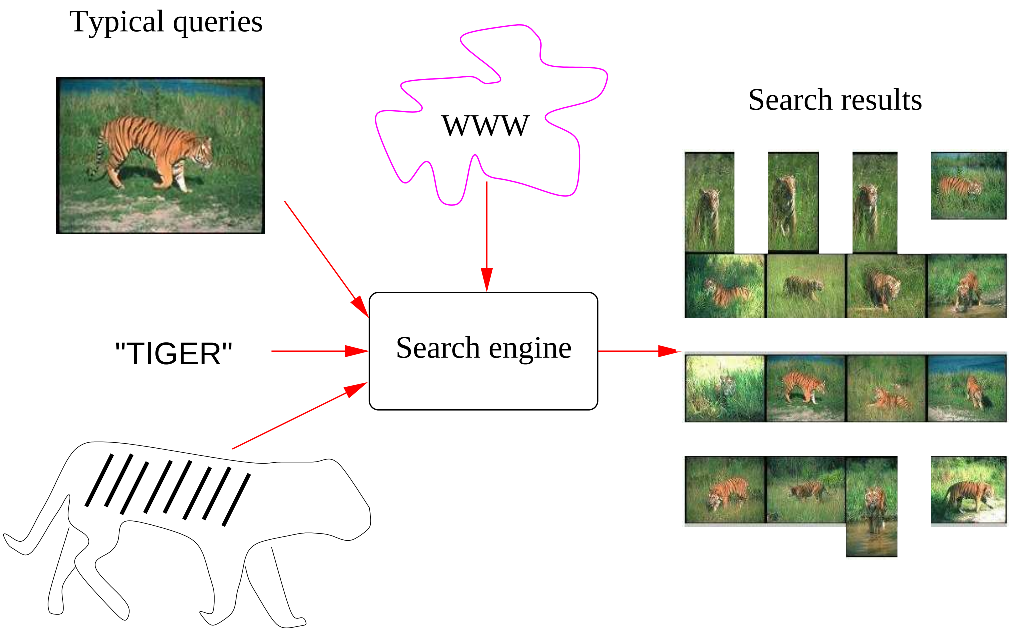

Figure 3: In multimedia databases, we may want to retrieve documents using images, text or cartoons.

to be soft in the sense that the combinations of items is taken into consideration, but documents which do not have a given item should still be considered. Finally, we would like the queries to be easily specified without reference to images already found. Mathematically, the problem of probabilistic classification of a new observation (query $\mathbf{x}_q$) into one of the n_c populations reduces to the derivation of

$$p(z_q = c|\mathbf{x}_q, \mathbf{x}_{1:n_x}) = \frac{p(\mathbf{x}_q|z_q = c)p(z_q = c|\mathbf{x}_{1:n_x})}{\sum_{c'=1}^{n_c} p(\mathbf{x}_{1:n_x}|z_q = c')p(z_q = c'|\mathbf{x}_{1:n_x})} \tag{48}$$

That is, to computing the probability that the query belongs to cluster c. Note that the first term in the numerator results from the following marginalisation

$$p(\mathbf{x}_q|z_q = c) = \int p(\mathbf{x}_q|z_q = c, \boldsymbol{\varphi})p(\boldsymbol{\varphi}|z_q = c)d\boldsymbol{\varphi} \tag{49}$$

4.1.1 Browsing, data visualisation and data mining

Mixture models are very suitable for studying coherence within groups. They allow us to visualise the data and identify hidden patterns. This feature is of great benefit when browsing image databases. Typically, setting up image databases so that their content is easy to internalize and thus navigate is difficult, and normally involves much human input. One of our goals in this work is to automate this task.

A key issue to browsing is whether the clusters found make sense to the user. If the user finds the clusters coherent, then they can begin to internalize the kind of structure they represent. Furthermore, a small portion of the cluster can be used to represent the whole, and will accurately suggest the kinds of pictures that will be found by exploring that cluster further.

4.1.2 Data compression

Clustering methods allow us to obtain a lower dimensional representation of the items in the database. This may reduce storage requirements and increase the efficiency of retrieval systems. In general, it is easier to search over a structured and lower dimensional space of clusters than to search over all the items in the database.

4.1.3 Annotation, Illustration and Recognition

Within our framework, one can build an application that takes text selected from a document, and suggests images to go with the text (Barnard and Forsyth 2001). This "auto-illustrate" application is essentially a process of linking pictures to words. However, it should be clear by symmetry that we can just as easily go the other way, and link words to pictures. This "auto-annotate" process is very interesting for a number of reasons. First, given an image, if we can produce reasonable words for an image feature, then we can use existing text search infrastructure to broaden searches beyond the confines of our system. For example, consider image search by user sketch. If the sketch contains an orange ball in the upper right corner, the annotation model might return the words "sun" and "sunset". These words can, in turn, be used to search for images that match the sketch using a text based search engine.

The association of text with images is even more interesting from a computer vision perspective because it is a form of minimally supervised learning of semantic labels for image features (Barnard and Forsyth 2001). As shown in Figure 4, the goal of recognition would be to label each of the image segments reasonably. Although extreme accuracy in this task is clearly wishful thinking, we argue that doing significantly better than chance is useful, and

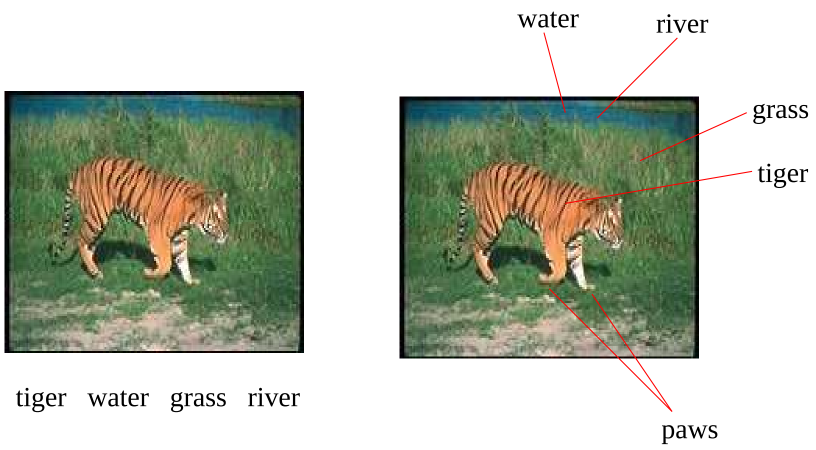

Figure 4: Annotation (left) and recognition (right) goals.

with care could be used to further bootstrap machine recognition. Doing significantly better than chance on this general task indicates that the system has learnt some correspondences between image components and words. This means that the system has learnt, with minimal supervision, something about recognition. This in turn is interesting in the face of a key vision problem, namely how to approach general recognition. Systems have been built which are relatively effective at recognizing specific things, usually under specific circumstances. Doing well at these tasks has generally required a lot of high quality training data. We also use a lot of data, but it is of a much more available nature. There is no shortage of image data with text, especially if one includes video. It seems that the information required is contained in these data sets, and therefore looking at the recognition problem in this way should bear fruit.

5 Experiments

In this section, we present two synthetic examples to illustrate the behaviour of the various algorithms. We then proceed to assess the performance of the algorithms on the Corel annotated image database.

5.1 Synthetic experiments

5.1.1 Text only example

It is possible to create artificial documents by using a multinomial model to generate them. This way we know the exact cluster and word probabilities and can use these to assess the performance of the various learning techniques. In this first experiment, we randomly generated 10 documents with 4 different words (allowing for repetition) from 2 clusters. The two clusters were selected with probabilities 0.2 and 0.8. We then tried to estimate the parameters of the generating model using the ML, MAP and empirical Bayes (EB) approaches. For all the algorithms, the number of clusters was assumed to be 6 and the number of EM iterations was set to 20. In the MAP case, we set $\alpha = 1$ and $\beta = 2$. In the EB case, the choice of these parameters is done automatically. Figure 5 shows the

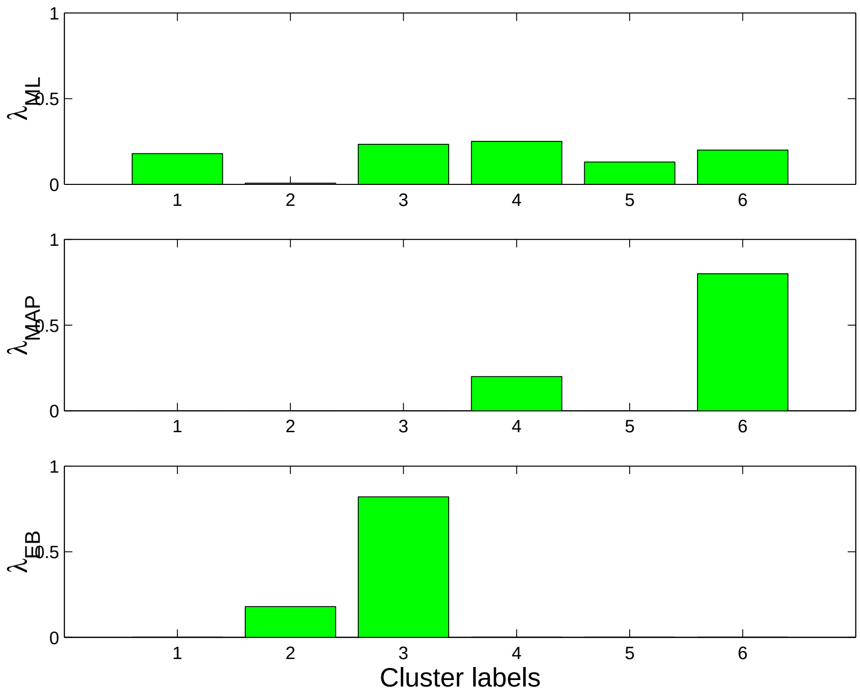

Figure 5: Cluster probabilities for the simple text example. Both the MAP and EB approaches recover the true cluster probabilities.

cluster probabilities (λ) computed by each method. Clearly, the MAP and EB approaches recover the generating probabilities, but the greedy maximum likelihood approach fails to accomplish this.

It is common in the LSA and LSI literature to prune the "negligible" clusters using heuristic procedures. These ad-hoc methods play a substantial role in the end results. A great advantage of the Bayesian approach is that this problem is eradicated to a large extent. If one varies a threshold over the values of λ and measures how many clusters are active, the Bayesian methods perform better as shown in Figure 6. The figure also shows that pruning, in the ML context, is very sensitive to the value of the chosen threshold.

We can assess the performance the three algorithms on a simple retrieval task. Docu-

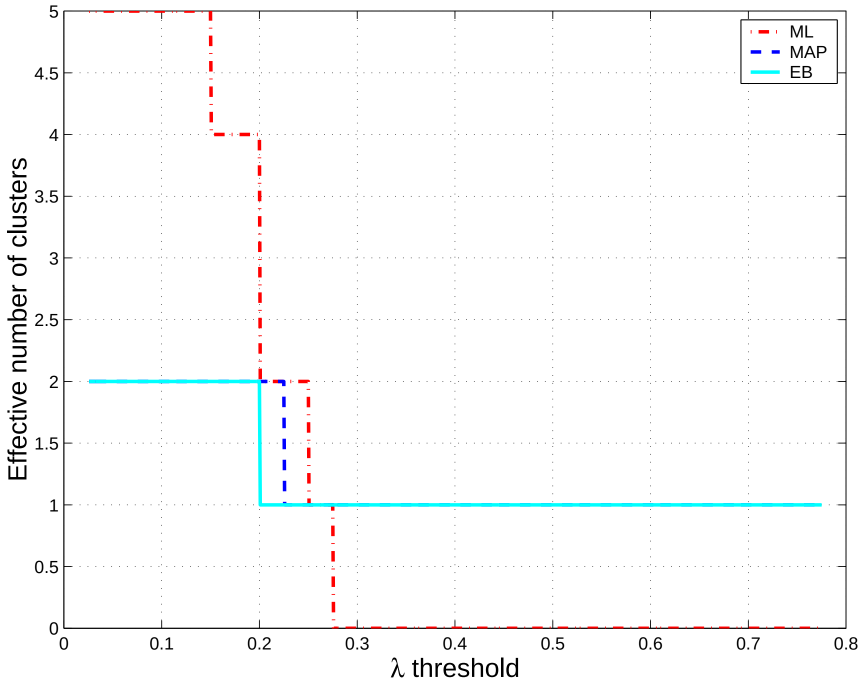

Figure 6: Effective number of clusters. Both the MAP and EB approaches are fairly insensitive to cluster pruning. The same is not true of the ML technique.

ments where generated using the following word probabilities

$$\boldsymbol{\delta}_{\text{True}} = \begin{pmatrix} 0.10 & 0.00 & 0.40 & 0.50 \\ 0.60 & 0.39 & 0.00 & 0.01 \end{pmatrix}$$

and the same cluster probabilities as before. After the EM maximisation, we queried the models generated by the three algorithms using a document from the training set, one from a test set and a single word. The queries and results are shown in Table 1. Given that the queries originate from the same cluster, the EB method is the most consistent of the three. The ability of assigning documents from the training and test set, that were generated by the same cluster, to the same estimated cluster can be used as a measure of performance. We repeated the experiment 100 times to compute the number of times the models failed to assign the test and training set documents to the same cluster. To gain some insight into the variance of this test, we repeated it 10 times. The results obtained are shown in Table 2. They confirm that the Bayesian approaches are more reliable.

One of the problems with the ML approach is that it creates multiple repetitions of the same mixture component so that the mixture weights of these repetitions add up, closely, to the true mixture weight. The Bayesian schemes on the other hand place irrelevant mixture components in regions of low probability. This is illustrated by means of an example

Query	ML	MAP	EB
(2 - 1 2)	0.05 0.00 **0.38** 0.33 0.21 0.04	0.00 0.07 0.00 0.01 0.03 **0.90**	0.00 **0.99** 0.00 0.00 0.01 0.00
(- - 3 2)	**0.49** 0.00 0.078 0.07 0.04 0.32	0.00 0.02 0.00 0.01 0.00 **0.97**	0.00 **1.00** 0.00 0.00 0.00 0.00
(- - - 1)	0.13 0.00 **0.32** 0.28 0.18 0.09	0.00 0.08 0.00 0.02 0.05 **0.85**	0.00 **0.96** 0.00 0.00 0.04 0.00

Table 1: The table shows the probabilities of cluster membership of a document from the training set (top), a document from a test set (middle) and a single word. The three documents belong to the same cluster. The EB and MAP methods outperform the ML approach.

λ		ML	MAP	EB
(0.2 0.8)	**Mean**	38	5	3
	Variance	8	1	1
(0.5 0.5)	**Mean**	28	4	3
	Variance	6	2	1

Table 2: Number of times the algorithms fail to assign documents in the training and test set (that originated from the same cluster in the generating model) to the same cluster in the learnt models. The table shows results for two choices of the mixing proportions in the generative model.

involving Gaussian distributions and text in the following section.

5.1.2 Text and arbitrary Gaussian features demo

In this example we cluster documents with three attributes: text with 4 words in the vocabulary, a univariate Gaussian and a bivariate Gaussian. Fifty documents were generated from two clusters with probability 0.5 each. We used the same word probabilities as in the previous experiment and set the two generating Gaussians in the one-dimensional case to $\mathcal{N}(0, 0.01)$ and $\mathcal{N}(1, 0.05)$. In the two-dimensional case, we set the two Gaussian clusters to $\mathcal{N}([0.5\ \ 0.5]', 0.01\mathbf{I})$ and $\mathcal{N}([2\ \ 2]', 0.05\mathbf{I})$. The hypothesized number of clusters was chosen to be 6, while the number of EM iterations was fixed to 10. We chose the same discrete prior parameters as before and selected the Gaussian hyperparameters as follows: $\kappa_{a,c} = 1$, $r_{a,c} = 5$, $\boldsymbol{\omega}_{a,c} = \mathbf{T}_{a,c,2} T_{a,c,1}^{-1}$ and $\boldsymbol{\Delta}_{a,c} = \left(\left(\mathbf{T}_{a,c,3} - T_{a,c,1}^{-1} \mathbf{T}_{a,c,2} \mathbf{T}'_{a,c,2}\right) T_{a,c,1}^{-1}\right)^{-1}$. We copied these values from (Bensmail et al. 1997) as they seem ensure that the estimates are relatively insensitive to reasonable changes in the prior. The EB method in this experiment was only applied to estimate the hyperparameters α_c.

Figure 7 shows the contour plot of the bivariate Gaussian clusters and the means of the six clusters computed by the three algorithms. Note that the the Bayesian approaches place some of the mixture components in a region of low probability. On the other hand, the ML

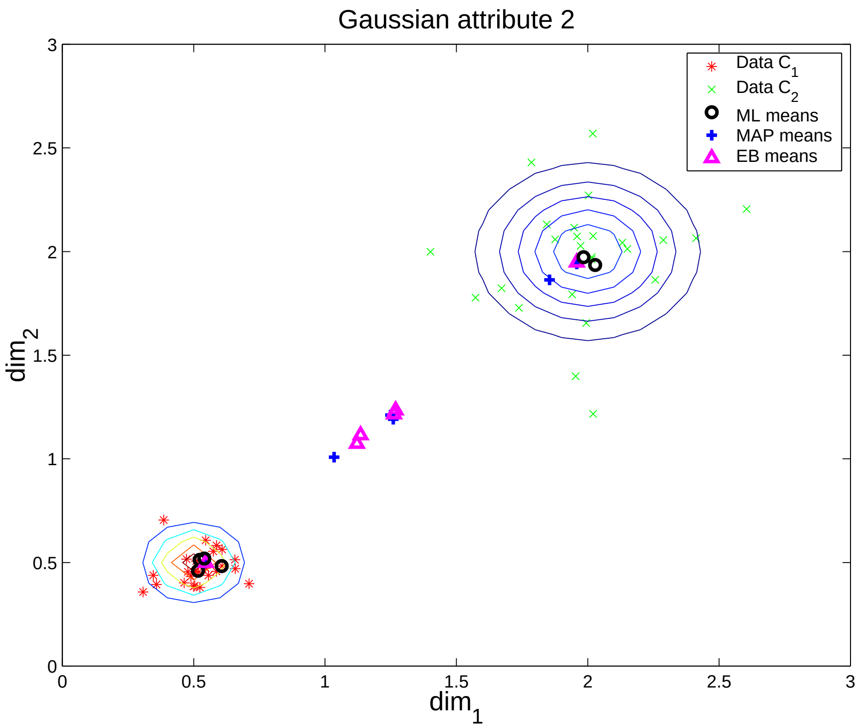

Figure 7: Contour plots of the bivariate attribute in the second synthetic example. The figure also shows means of the cluster components found with the ML, MAP and EB approaches.

approach places the mixture components at the same location. It still works because the mixture components' weights add up to the right value. It is, however, somehow wasteful and unsatisfactory.

5.2 Large image and text database

We performed several experiments on the Corel image database. This database contains images annotated with approximately 3 to 5 keywords each. The images on each of the CDs provided by Corel are samples from a particular theme. This makes this database very suitable for testing our algorithms.

We clustered the documents (each corresponding to one image and its keywords) using 3 different sets of attributes: text only, image features only and combined text and images. The image features were derived from image histograms. Specifically, the image histogram is projected onto a lower dimensional space of histograms. This space is found by PCA on the space of image histograms for a much larger set of images: see Blobworld (Belongie et al. 1997) for more details, but note that, in this example, the entire image is treated as one blob and the histogram features (30 PCA components) are for the entire image. It is, however, possible to use hierarchical models to treat the image as a mixture of blobs (Barnard and Forsyth 2001).

We found that the MAP and EB approaches lead to more parsimonious representations in each of the three testing scenarios[2]. For example, when clustering the contents of the first 10 CDs of the database (1000 documents), assuming the number of clusters to be 20, the three algorithms yielded the cluster probabilities, λ, shown in Figure 8. Clearly, the

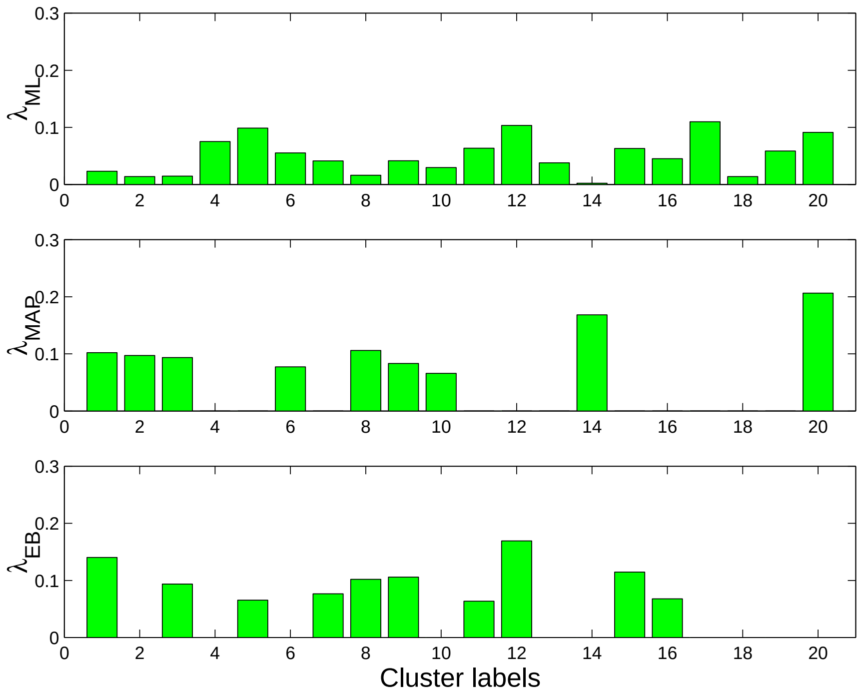

Figure 8: Cluster probabilities in the Corel example.

Bayesian schemes find a number of themes that is in more accordance with the human Corel choice. In addition, the clusters are more coherent.

One of the advantages of combining image and text attributes in the model is that people relate to images using both semantic and visual content. For example, if we want pictures of tigers on light green grasslands, we might do a search with the word "tiger" and a light grassland picture. This will hopefully not return images of tigers in dark places. Figure 9 shows an example where clustering images from CDs containing tigers, using text and image features, results in the two separate, coherent clusters.

[2]Since it is difficult to present these results in this paper format, we have made them available at `http://elib.CS.Berkeley.EDU/papers/clustering/bayesian/index.html`.

Figure 9: Result of clustering documents using both images and keywords. The two example clusters have obvious text and image semantics. That is, at the text level both groups relate to the word "tiger" and at the image level there are tigers on light green grasslands and tigers in dark places.

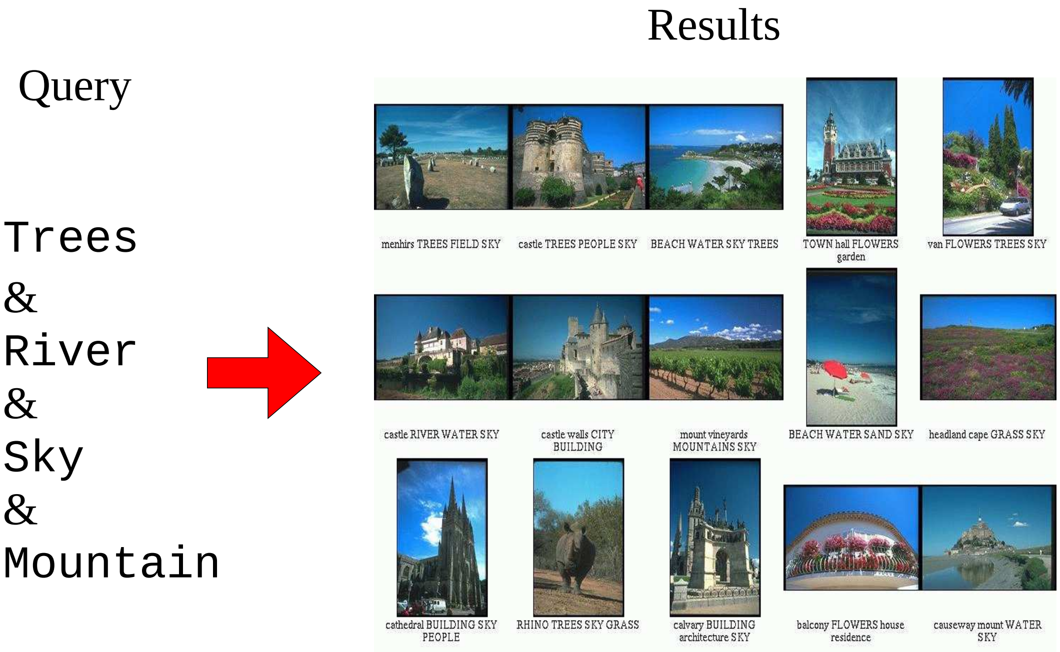

Figure 10: Results using text query with conjunction of words that does not appear in the database. The model generalises to a reasonable extent.

We also performed some retrieval experiments using the learnt models. Figure 11 shows that for a query consisting of a conjunction of words that does not appear in the database, we are able to retrieve images where it is clear that the models are generalising reasonably.

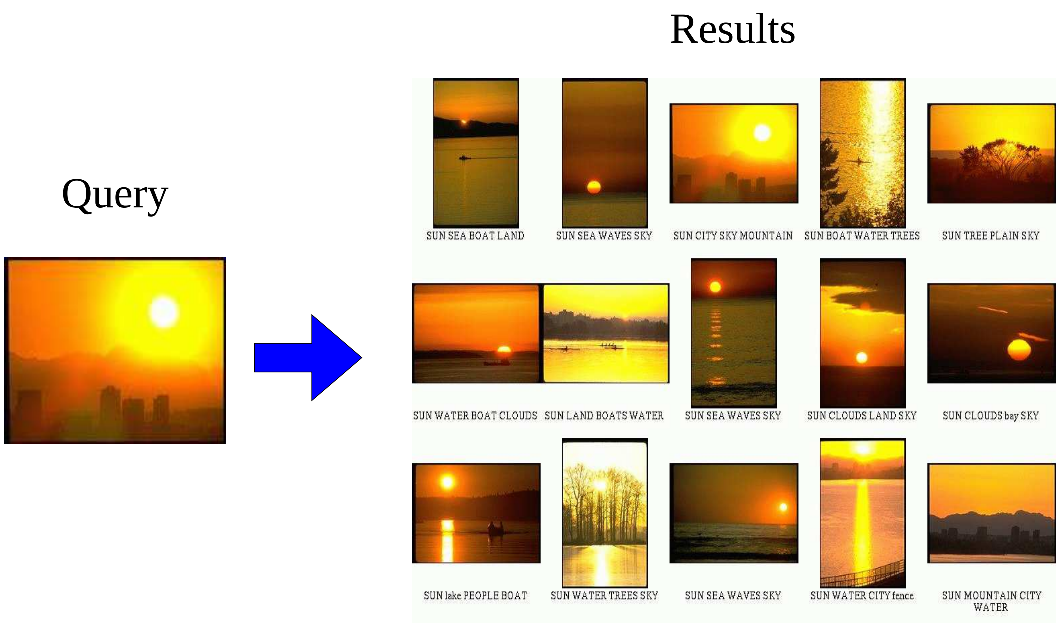

Figure 11: Results using image query.

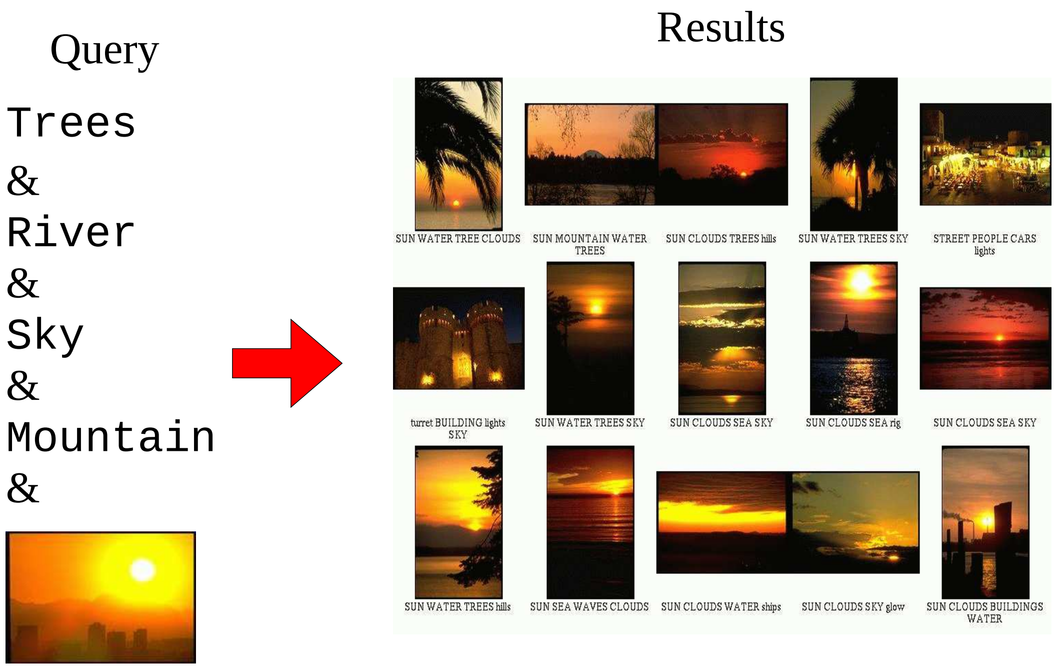

Figure 12: Results using image query of Figure 11 with added text. The model still returns images of sunsets, but this time there are trees.

Figures 10 and 12 show that the querying system works well and that by adding words to an image we can bias the results in useful ways.

6 Conclusions

We have shown that we can improve probabilistic semantic modelling by adopting a Bayesian approach. In particular, one is able to obtain more parsimonious and coherent models. We are currently exploring many modelling, algorithmic, application, and validation extensions.

Acknowledgements

We would like to thank David Cohn, Arnaud Doucet, David Forsyth, Jan Puzicha and Stuart Russell.

A Notation

Specific meanings

n_a	Number of attributes.
n_c	Number of clusters.
n_x	Number of documents.
n_v	Number of discrete values.
$n_{i,a}$	Number of times word a appears in document i.
$n_{i,:}$	Number of words in document i.
$\boldsymbol{\lambda}$	Mixture coefficients.
$\mathbf{z}$	Mixture indicator variables.
$\boldsymbol{\delta}$	Parameters of multinomial components.
$\boldsymbol{\mu}$	Mean of Gaussian components.
$\boldsymbol{\Sigma}$	Covariance of Gaussian components.

Symbols

$\mathbf{z}_{1:t}$	Stacked vector $\mathbf{z}_{1:t} \triangleq (z_1, ..., z_{j-1}, z_j, z_{j+1}, ..., z_t)'$.
$\mathbf{z}_{-j}$	Vector with j-th component missing $\mathbf{z}_{-j} \triangleq (z_1, ..., z_{j-1}, z_{j+1}, ..., z_k)'$.
$\mathbf{A}_{i,j}$	Entry of the matrix $\mathbf{A}$ in the i^{th} row and j^{th} column.
$\mathbf{A}_{1:p,1:q,1:r}$	Three-dimensional matrix of size $p \times q \times r$.
$\mathbf{I}_n$	Identity matrix of dimension $n \times n$.
$\mathbf{1}_{n,n}$	Matrix of ones of dimension $n \times n$.
$\mathbf{0}_{n,n}$	Matrix of zeros of dimension $n \times n$.
$\mathbb{R}^n$	Euclidean n-dimensional space.
$p(d\mathbf{z})$	Distribution of $\mathbf{z}$.
$p(\mathbf{z})$	Density of $\mathbf{z}$.
$p(d\mathbf{z}\|\mathbf{y})$	Conditional distribution of $\mathbf{z}$ given $\mathbf{y}$.
$p(d\mathbf{z}, d\mathbf{y})$	Joint distribution of $\mathbf{z}$ and $\mathbf{y}$.
$\mathbf{z} \sim p(d\mathbf{z})$	$\mathbf{z}$ is distributed according to $p\,(d\mathbf{z})$.

Operators and functions

$\mathbf{A}'$	Transpose of matrix $\mathbf{A}$.
$\mathbf{A}^{-1}$	Inverse of matrix $\mathbf{A}$.
$\mathrm{tr}(\mathbf{A})$	Trace of matrix $\mathbf{A}$.
$\lvert\mathbf{A}\rvert$	Determinant of matrix $\mathbf{A}$.
$\mathrm{diag}(\mathbf{z})$	Matrix with entries $\mathbf{z}$ in its diagonal.
$\mathbb{I}_E(\mathbf{z})$	Indicator function of the set E (1 if $\mathbf{z} \in E$, 0 otherwise).
$\delta_{\mathbf{z}_i}(d\mathbf{z})$	Dirac delta function (impulse function).
$\mathbb{E}(\mathbf{z})$	Expectation of the random variable $\mathbf{z}$.
$var(\mathbf{z})$	Variance of the random variable $\mathbf{z}$.
$\exp(\cdot)$	Exponential function.
$\Gamma(\cdot)$	Gamma function.
$\Psi(\cdot)$	Digamma function.
$\log(\cdot)$	Logarithmic function of base e (ln).
min, max	Extrema with respect to an integer value.
inf, sup	Extrema with respect to a real value.
$\arg\min\limits_{\mathbf{z}}$	The argument $\mathbf{z}$ that minimises the operand.
$\arg\max\limits_{\mathbf{z}}$	The argument $\mathbf{z}$ that maximises the operand.

Standard probability distributions

Bernoulli	$\mathcal{B}r(z\|\alpha)$	$\alpha^z(1-\alpha)^{(1-z)}\mathbb{I}_{\{0,1\}}(z)$
Beta	$\mathcal{B}e(z\|\alpha,\beta)$	$\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}z^{\alpha-1}(1-z)^{\beta-1}\mathbb{I}_{(0,1)}(z)$
Multinomial	$\mathcal{M}_k(\mathbf{z}\|n,\boldsymbol{\alpha})$	$\binom{n}{z_1\ldots z_k}\prod_{i=1}^k \alpha_i^{z_i}\mathbb{I}_{\sum_i z_i=n}$
Dirichlet	$\mathcal{D}_k(\mathbf{z}\|\boldsymbol{\alpha})$	$\frac{\Gamma(\sum_i \alpha_i)}{\prod_i \Gamma(\alpha_i)}\prod_{i=1}^k \mathbf{z}^{\alpha_i-1}\mathbb{I}_{\sum_i z_i=1}$
Gaussian	$\mathcal{N}(\mathbf{z}\|\boldsymbol{\mu},\Sigma)$	$\|2\pi\boldsymbol{\Sigma}\|^{-1/2}\exp\left(-\frac{1}{2}(\mathbf{z}-\boldsymbol{\mu})'\boldsymbol{\Sigma}^{-1}(\mathbf{z}-\boldsymbol{\mu})\right)$
Wishart	$\mathcal{W}_k\left(\boldsymbol{\Sigma}^{-1}\|r,\boldsymbol{\Delta}\right)$	$\frac{\|\boldsymbol{\Sigma}\|^{-\frac{1}{2}(r-k-1)}\exp\left[-\frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Sigma}^{-1}\boldsymbol{\Delta}^{-1}\right)\right]}{2^{\frac{1}{2}k}\pi^{\frac{1}{4}k(1-k)}\|\boldsymbol{\Delta}\|^{\frac{1}{2}r}\prod_{l=1}^k\Gamma(\frac{1}{2}(r-l+1))}$
Gamma	$\mathcal{G}a(z\|\alpha,\beta)$	$\frac{\beta^\alpha}{\Gamma(\alpha)}z^{\alpha-1}\exp(-\beta z)\mathbb{I}_{[0,+\infty)}(z)$
Inverse Gamma	$\mathcal{IG}(z\|\alpha,\beta)$	$\frac{\beta^\alpha}{\Gamma(\alpha)}z^{-\alpha-1}\exp(-\beta/z)\mathbb{I}_{[0,+\infty)}(z)$
Poisson	$\mathcal{P}n(z\|\lambda)$	$\frac{\lambda^z}{z!}\exp(-\lambda)\mathbb{I}_{\mathbb{N}}(z)$
Uniform	$\mathcal{U}_A(\mathbf{z})$	$\left[\int_A d\mathbf{z}\right]^{-1}\mathbb{I}_A(\mathbf{z})$

B Expanding the $\mathbf{Q}^{\mathrm{ML}}$ Function

The $\mathbf{Q}^{\mathrm{ML}}$ function can be expanded as follows

$$\begin{aligned}
\mathbf{Q}^{\mathrm{ML}} &= \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\boldsymbol{\varphi}^{(\mathrm{old})})}\left\{\log\left[\prod_{i=1}^{n_x}p(\mathbf{x}_i,z_i=c|\boldsymbol{\varphi})\right]\right\}\\
&= \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\boldsymbol{\varphi}^{(\mathrm{old})})}\left\{\log\left[\prod_{i=1}^{n_x}\prod_{c=1}^{n_c}\left(\lambda_c\prod_{a=1}^{n_a}p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c})\right)^{\mathbb{I}_c(z_i)}\right]\right\}\\
&= \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\boldsymbol{\varphi}^{(\mathrm{old})})}\left\{\sum_{i=1}^{n_x}\sum_{c=1}^{n_c}\mathbb{I}_c(z_i)\log\left[\lambda_c\prod_{a=1}^{n_a}p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c})\right]\right\}\\
&= \sum_{c_1=1}^{n_c}\cdots\sum_{c_{n_x}=1}^{n_c}\left\{\sum_{i=1}^{n_x}\sum_{c=1}^{n_c}\mathbb{I}_c(z_i)\log\left[\lambda_c\prod_{a=1}^{n_a}p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c})\right]\right\}\prod_{j=1}^{n_x}p(z_j|\mathbf{x}_j,\boldsymbol{\varphi}^{(\mathrm{old})})\\
&= \sum_{i=1}^{n_x}\sum_{c=1}^{n_c}\log\left[\lambda_c\prod_{a=1}^{n_a}p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c})\right]\sum_{c_1=1}^{n_c}\cdots\sum_{c_{n_x}=1}^{n_c}\mathbb{I}_c(z_i)\prod_{j=1}^{n_x}p(z_j|\mathbf{x}_j,\boldsymbol{\varphi}^{(\mathrm{old})})
\end{aligned}$$

This expression can be greatly simplified by noticing that

$$\begin{aligned}
&\sum_{c_1=1}^{n_c}\cdots\sum_{c_{n_x}=1}^{n_c}\mathbb{I}_c(z_i)\prod_{j=1}^{n_x}p(z_j|\mathbf{x}_j,\boldsymbol{\varphi}^{(\mathrm{old})})\\
&\qquad= \left[\sum_{c_1=1}^{n_c}\cdots\sum_{c_{i-1}=1}^{n_c}\sum_{c_{i+1}=1}^{n_c}\cdots\sum_{c_{n_x}=1}^{n_c}\prod_{j=1,j\neq i}^{n_x}p(z_j|\mathbf{x}_j,\boldsymbol{\varphi}^{(\mathrm{old})})\right]p(z_i=c|\mathbf{x}_i,\boldsymbol{\varphi}^{(\mathrm{old})})\\
&\qquad= \prod_{j=1,j\neq i}^{n_x}\left[\sum_{c_j=1}^{n_c}p(z_j|\mathbf{x}_j,\boldsymbol{\varphi}^{(\mathrm{old})})\right]p(z_i=c|\mathbf{x}_i,\boldsymbol{\varphi}^{(\mathrm{old})})\\
&\qquad= p(z_i=c|\mathbf{x}_i,\boldsymbol{\varphi}^{(\mathrm{old})})
\end{aligned}$$

Consequently, the $\mathbf{Q}^{\mathrm{ML}}$ function simplifies to

$$\mathbf{Q}^{\mathrm{ML}} = \sum_{i=1}^{n_x} \sum_{c=1}^{n_c} p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\mathrm{old})}) \log \left[\lambda_c \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c}) \right]$$

C M Step for Categorical Attributes

C.1 ML estimates

In the M step for categorical variables, we need to maximise $\mathbf{Q}^{\mathrm{ML}}$ subject to the constraints $\sum_{c=1}^{n_c} \lambda_c = 1$ and $\sum_{j=1}^{n_v} \delta_{a,c,j} = 1$. To compute λ, we introduce Lagrange multipliers, μ, and maximise the Lagrangian

$$L = \mathbf{Q}^{\mathrm{ML}} + \mu \left(1 - \sum_{c=1}^{n_c} \lambda_c \right)$$

by differentiating with respect to λ_c and equating to zero. That is, we want to compute

$$\begin{aligned} \frac{\partial}{\partial \lambda_c} \left\{ \sum_{i=1}^{n_x} \sum_{c=1}^{n_c} p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\mathrm{old})}) \log \left[\lambda_c \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c}) \right] + \mu \left(1 - \sum_{c=1}^{n_c} \lambda_c \right) \right\} &= 0 \\ \sum_{i=1}^{n_x} p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\mathrm{old})}) \frac{1}{\lambda_c} - \mu &= 0 \end{aligned}$$

Summing both sides over $c = 1, \ldots, n_c$, we get $\mu = n_x$. Therefore, the estimate for λ_c, with $\xi_{i,c} = p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\mathrm{old})})$, is

$$\lambda_c = \frac{1}{n_x} \sum_{i=1}^{n_x} \xi_{i,c}$$

When the mixture components are discrete, we need to compute

$$\begin{aligned} \frac{\partial}{\partial \delta_{a,c,j}} \left\{ \sum_{i=1}^{n_x} \sum_{c=1}^{n_c} p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\mathrm{old})}) \log \left[\lambda_c \prod_{a=1}^{n_a} \prod_{j=1}^{n_v} \delta_{a,c,j}^{\mathbb{I}_j(\mathbf{x}_{i,a})} \right] + \mu \left(1 - \sum_{j=1}^{n_v} \delta_{a,c,j} \right) \right\} &= 0 \\ \sum_{i=1}^{n_x} p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\mathrm{old})}) \mathbb{I}_j(\mathbf{x}_{i,a}) \frac{1}{\delta_{a,c,j}} - \mu &= 0 \end{aligned}$$

Summing both sides over $j = 1, \ldots, n_v$, we get $\mu = \sum_{i=1}^{n_x} p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\mathrm{old})})$ and, hence, the estimate of $\delta_{a,c,j}$ is

$$\delta_{a,c,j} = \frac{\sum_{i=1}^{n_x} \mathbb{I}_j(\mathbf{x}_{i,a}) \xi_{i,c}}{\sum_{i=1}^{n_x} \xi_{i,c}}$$

C.2 MAP estimates

The unconstrained objective function is

$$\mathbf{Q}^{\mathrm{MAP}} = \mathbf{Q}^{\mathrm{ML}} + \log p(\boldsymbol{\varphi})$$

Hence, to compute λ_c, we proceed as in the previous section by differentiating the augmented Lagrangian with respect to λ_c and equating to zero

$$\begin{aligned}\frac{\partial}{\partial \lambda_c}\left\{\sum_{i=1}^{n_x}\sum_{c=1}^{n_c} p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\text{old})}) \log\left[\lambda_c \prod_{a=1}^{n_a} p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c})\right] + \mu\left(1 - \sum_{c=1}^{n_c}\lambda_c\right)\right. & \quad + \\ \left.\log\left[\frac{\Gamma(\sum_{c=1}^{n_c}\alpha_c)}{\prod_{c=1}^{n_c}\Gamma(\alpha_c)}\prod_{c=1}^{n_c}\lambda_c^{\alpha_c-1}\right]\right\} & = 0 \\ \sum_{i=1}^{n_x}\left(p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\text{old})}) + \alpha_c - 1\right)\frac{1}{\lambda_c} - \mu & = 0\end{aligned}$$

Summing both sides over $c = 1, \ldots, n_c$, we get $\mu = n_x - n_c + \sum_c \alpha_c$. Hence, the estimate for λ_c is

$$\lambda_c = \frac{\sum_{i=1}^{n_x}\xi_{i,c} + \alpha_c - 1}{n_x + \sum_{c'}\alpha'_c - n_c}$$

Similarly, the constrained maximisation for $\delta_{a,c,j}$ requires that we compute

$$\begin{aligned}\frac{\partial}{\partial \delta_{a,c,j}}\left\{\sum_{i=1}^{n_x}\sum_{c=1}^{n_c} p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\text{old})}) \log\left[\lambda_c \prod_{a=1}^{n_a}\prod_{j=1}^{n_v}\delta_{a,c,j}^{\mathbb{I}_j(\mathbf{x}_{i,a})}\right] + \mu\left(1 - \sum_{j=1}^{n_v}\delta_{a,c,j}\right)\right. & \quad + \\ \left.\log\left[\prod_{c=1}^{n_c}\prod_{a=1}^{n_a}\frac{\Gamma(\sum_{j=1}^{n_v}\beta_{a,c,j})}{\prod_{j=1}^{n_v}\Gamma(\beta_{a,c,j})}\prod_{j=1}^{n_v}\delta_{a,c,j}^{\beta_{a,c,j}-1}\right]\right\} & = 0 \\ \sum_{i=1}^{n_x}\left(p(z_i = c|\mathbf{x}_i, \boldsymbol{\varphi}^{(\text{old})})\mathbb{I}_j(\mathbf{x}_{i,a}) + \beta_{a,c,j} - 1\right)\frac{1}{\delta_{a,c,j}} - \mu & = 0\end{aligned}$$

thus yielding

$$\delta_{a,c,j} = \frac{\sum_{i=1}^{n_x}\mathbb{I}_j(\mathbf{x}_{i,a})\xi_{i,c} + \beta_{a,c,j} - 1}{\sum_{i=1}^{n_x}\xi_{i,c} + \sum_{j'}\beta_{a,c,j'} - n_v}$$

D M Step for the Categorical Hyperparameters

To compute the hyperparameters, one has to maximise the expected marginal log-likelihood as follows

$$\begin{aligned}\mathbf{Q}^{\mathrm{E}} &= \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\boldsymbol{\varphi}^{(\text{old})})}\left[\log p(\mathbf{z},\mathbf{x}|\boldsymbol{\eta})\right] \\ &= \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\boldsymbol{\varphi}^{(\text{old})})}\left[\log\int p(\mathbf{z},\mathbf{x}|\boldsymbol{\varphi})p(\boldsymbol{\varphi}|\boldsymbol{\eta})d\boldsymbol{\varphi}\right]\end{aligned}$$

For instance, we can compute α, by maximising

$$
\begin{aligned}
\mathbf{Q}_{\alpha}^{\mathrm{E}} &= \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\boldsymbol{\varphi}^{(\mathrm{old})})}\left\{\log\int\left[\prod_{i=1}^{n_x}\prod_{c=1}^{n_c}\left(\lambda_c\prod_{a=1}^{n_a}p(\mathbf{x}_{i,a}|\boldsymbol{\theta}_{a,c})\right)^{\mathbb{I}_c(z_i)}\right]\left[\frac{\Gamma(\sum_{c=1}^{n_c}\alpha_c)}{\prod_{c=1}^{n_c}\Gamma(\alpha_c)}\prod_{c=1}^{n_c}\lambda_c^{\alpha_c-1}\right]d\boldsymbol{\lambda}\right\} \\
&\propto \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\boldsymbol{\varphi}^{(\mathrm{old})})}\left\{\log\frac{\Gamma(\sum_{c=1}^{n_c}\alpha_c)}{\prod_{c=1}^{n_c}\Gamma(\alpha_c)}\int\prod_{c=1}^{n_c}\lambda_c^{k_c+\alpha_c-1}d\boldsymbol{\lambda}\right\} \\
&= \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\boldsymbol{\varphi}^{(\mathrm{old})})}\left\{\log\frac{\Gamma(\sum_{c=1}^{n_c}\alpha_c)}{\prod_{c=1}^{n_c}\Gamma(\alpha_c)}\frac{\prod_{c=1}^{n_c}\Gamma(k_c+\alpha_c)}{\Gamma(\sum_{c=1}^{n_c}(k_c+\alpha_c))}\right\} \\
&= \mathbb{E}_{p(\mathbf{z}|\mathbf{x},\boldsymbol{\varphi}^{(\mathrm{old})})}\left\{\log\frac{\Gamma(\sum_{c=1}^{n_c}\alpha_c)}{\Gamma(n_x+\sum_{c=1}^{n_c}\alpha_c))}\prod_{c=1}^{n_c}\frac{\Gamma(k_c+\alpha_c)}{\Gamma(\alpha_c)}\right\} \\
&= \log\frac{\Gamma(\sum_{c=1}^{n_c}\alpha_c)}{\Gamma(n_x+\sum_{c=1}^{n_c}\alpha_c))}\prod_{c=1}^{n_c}\frac{\Gamma(\sum_{i=1}^{n_x}\xi_{i,c}+\alpha_c)}{\Gamma(\alpha_c)} \\
&= \log\Gamma\left(\sum_{c=1}^{n_c}\alpha_c\right)-\log\Gamma\left(n_x+\sum_{c=1}^{n_c}\alpha_c\right)+\sum_{c=1}^{n_c}\left(\log\Gamma\left(\sum_{i=1}^{n_x}\xi_{i,c}+\alpha_c\right)-\log\Gamma(\alpha_c)\right)
\end{aligned}
$$

The derivative of $\mathbf{Q}_{\alpha}^{\mathrm{E}}$ with respect to α_c yields

$$
\mathbf{g}_c=\frac{\partial}{\partial\alpha_c}\mathbf{Q}_{\alpha}^{\mathrm{E}}=\Psi\left(\sum_{c=1}^{n_c}\alpha_c\right)-\Psi\left(n_x+\sum_{c=1}^{n_c}\alpha_c\right)+\Psi\left(\sum_{i=1}^{n_x}\xi_{i,c}+\alpha_c\right)-\Psi(\alpha_c)
$$

where $\Psi(\alpha_c)\triangleq\frac{\partial}{\partial\alpha_c}\log\Gamma(\alpha_c)$ is the digamma function. We may solve these fixed point equations for α_c using the following iterative updates (Minka 2000)

$$
\alpha_c^{(\mathrm{new})}=\alpha_c\frac{\Psi\left(\sum_{i=1}^{n_x}\xi_{i,c}+\alpha_c\right)-\Psi(\alpha_c)}{\Psi\left(n_x+\sum_{c=1}^{n_c}\alpha_c\right)-\Psi\left(\sum_{c=1}^{n_c}\alpha_c\right)}
$$

It is also possible to employ Newton-Raphson schemes (Narayanan 1991, Ronning 1989). This requires that we compute the Hessian matrix as follows

$$
\begin{aligned}
\mathbf{H} &= \frac{\partial^2}{\partial\alpha_c\partial\alpha_k}\mathbf{Q}_{\alpha}^{\mathrm{E}} \\
&= \mathbf{1}_{n_c\times n_c}\left[\Psi'\left(\sum_{c=1}^{n_c}\alpha_c\right)-\Psi'\left(n_x+\sum_{c=1}^{n_c}\alpha_c\right)\right]+\mathrm{diag}\left(\Psi'\left(\sum_{i=1}^{n_x}\xi_{i,c}+\alpha_c\right)-\Psi'(\alpha_c)\right)
\end{aligned}
$$

where $\Psi'(\cdot)$ is the trigamma function (second derivative). The Newton-Raphson algorithm proceeds as follows

$$
\boldsymbol{\alpha}^{(\mathrm{new})}=\boldsymbol{\alpha}^{(\mathrm{old})}+\mathbf{H}^{-1}\mathbf{g}
$$

where, for efficiency, $\mathbf{H}^{-1}$ can be expanded in terms of the matrix inversion lemma.

A similar estimate for $\beta_{a,c,j}$ can be obtained by first defining the objective function

$$\begin{aligned}
\mathbf{Q}_\beta^{\mathrm{E}} &= \log \int \left[\prod_{i=1}^{n_x} \prod_{c=1}^{n_c} \left(\lambda_c \prod_{a=1}^{n_a} \prod_{j=1}^{n_v} \delta_{a,c,j}^{\mathbb{I}_j(\mathbf{x}_{i,a})} \right)^{\xi_{i,c}} \right] \left[\prod_{c=1}^{n_c} \prod_{a=1}^{n_a} \frac{\Gamma(\sum_{j=1}^{n_v} \beta_{a,c,j})}{\prod_{j=1}^{n_v} \Gamma(\beta_{a,c,j})} \prod_{j=1}^{n_v} \delta_{a,c,j}^{\beta_{a,c,j}-1} \right] d\boldsymbol{\delta} \\
&\propto \log \left[\prod_{c=1}^{n_c} \prod_{a=1}^{n_a} \frac{\Gamma(\sum_{j=1}^{n_v} \beta_{a,c,j})}{\prod_{j=1}^{n_v} \Gamma(\beta_{a,c,j})} \int \prod_{j=1}^{n_v} \delta_{a,c,j}^{\sum_{i=1}^{n_x} \xi_{i,c} \mathbb{I}_j(\mathbf{x}_{i,a}) + \beta_{a,c,j} - 1} d\boldsymbol{\delta} \right] \\
&= \sum_{c=1}^{n_c} \sum_{a=1}^{n_a} \log \frac{\Gamma(\sum_{j=1}^{n_v} \beta_{a,c,j})}{\Gamma(\sum_{j=1}^{n_v} (\sum_{i=1}^{n_x} \xi_{i,c} \mathbb{I}_j(\mathbf{x}_{i,a}) + \beta_{a,c,j}))} \prod_{j=1}^{n_v} \frac{\Gamma(\sum_{i=1}^{n_x} \xi_{i,c} \mathbb{I}_j(\mathbf{x}_{i,a}) + \beta_{a,c,j})}{\Gamma(\beta_{a,c,j})}
\end{aligned}$$

The derivative of $\mathbf{Q}_\beta^{\mathrm{E}}$ with respect to $\beta_{a,c,j}$ yields

$$\begin{aligned}
&\mathbf{g}_{a,c,j} = \frac{\partial}{\partial \beta_{a,c,j}} \mathbf{Q}_\beta^{\mathrm{E}} \\
&= \Psi\left(\sum_{j=1}^{n_v} \beta_{a,c,j} \right) - \Psi\left(\sum_{j=1}^{n_v} \left(\sum_{i=1}^{n_x} \xi_{i,c} \mathbb{I}_j(\mathbf{x}_{i,a}) + \beta_{a,c,j} \right) \right) + \Psi\left(\sum_{i=1}^{n_x} \xi_{i,c} \mathbb{I}_j(\mathbf{x}_{i,a}) + \beta_{a,c,j} \right) - \Psi(\beta_{a,c,j})
\end{aligned}$$

The corresponding Hessian matrix is

$$\begin{aligned}
\mathbf{H} &= \frac{\partial^2}{\partial \alpha_c \partial \alpha_k} \mathbf{Q}_\alpha^{\mathrm{E}} \\
&= \mathbf{1}_{n_c \times n_c} \left[\Psi'\left(\sum_{j=1}^{n_v} \beta_{a,c,j} \right) - \Psi'\left(\sum_{j=1}^{n_v} \left(\sum_{i=1}^{n_x} \xi_{i,c} \mathbb{I}_j(\mathbf{x}_{i,a}) \right) + \beta_{a,c,j} \right) \right] \\
&\quad + \operatorname{diag}\left(\Psi'\left(\sum_{i=1}^{n_x} \sum_{i=1}^{n_x} \xi_{i,c} \mathbb{I}_j(\mathbf{x}_{i,a}) + \beta_{a,c,j} \right) - \Psi(\beta_{a,c,j}) \right)
\end{aligned}$$

The fixed point updates follow from the previous case.

References

Andrieu, C., de Freitas, N. and Doucet, A. (2000). Robust full Bayesian learning for radial basis networks, to appear in Neural Computation.

Barnard, K. and Forsyth, D. (2001). Learning the semantics of words and pictures, to appear in ICCV2001.

Baum, L. E., Petrie, T., Soules, G. and Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains, *Annals of Mathematical Statistics* **41**: 164–171.

Beeferman, D., Berger, A. and Lafferty, J. (1999). Statistical models for text segmentation, *Machine Learning, Special Issue on Natural Language Learning* **34**(1-3): 177–210.

Belongie, S., Carson, C., Greenspan, H. and Malik, J. (1997). Color and texture-based image segmentation using EM and its application to content-based image retrieval, *International Conference on Computer Vision*, pp. 675–682.

Bensmail, H., Celeux, G., Raftery, A. E. and Robert, C. P. (1997). Inference in model-based cluster analysis, *Statistics and Computing* **7**: 1–10.

Bernardo, J. M. and Smith, A. F. M. (1994). *Bayesian Theory*, Wiley Series in Applied Probability and Statistics.

Brin, S. and Page, L. (1998). Anatomy of a large-scale hypertextual web search engine, *7th International World Wide Web Conference.*

Carlin, B. P. and Louis, T. A. (2000). *Bayes and Empirical Bayes Methods for Data Analysis*, second edn, Chapman and Hall.

Chang, P. C. and Afifi, A. A. (1974). Classification based on dichotomous and continuous variables, *Journal of the American Statistical Association* **69**(346): 336–339.

Chen, C. F. (1979). Bayesian inference for a normal dispersion matrix and its application to stochastic multiple regression analysis, *Journal of the Royal Statistical Society. Series B* **41**(2): 235–248.

Cohn, D. and Hofmann, T. (2001). The missing link - a probabilistic model of document content and hypertext connectivity, *in* T. K. Leen, T. G. Dietterich and V. Tresp (eds), *Advances in Neural Information Processing Systems*, Vol. 10.

de Freitas, N. and Barnard, K. (2001). Bayesian latent semantic analysis, under review, Computer Science Division, UC Berkeley.

Dempster, A. P., Laird, N. M. and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society Series B* **39**: 1–38.

Diebolt, J. and Robert, C. P. (1994). Estimation of finite mixture distributions through Bayesian sampling, *Journal of the Royal Statistical Society B* **56**(2): 363–375.

Gelman, A., Carlin, J. B., Stern, H. S. and Rubin, D. B. (1995). *Bayesian Data Analysis*, Chapman and Hall.

Good, I. J. (1983). *Good Thinking: The Foundations of Probability and its Applications*, Minnesota Press, Minneapolis.

Gull, S. (1988). Bayesian inductive inference and maximum entropy, *in* G. J. Erickson and C. R. Smith (eds), *Maximum Entropy and Bayesian Methods in Science and Engineering*, Kluwer Academic Publishers, pp. 53–74.

Hofmann, T. (1999). Probabilistic latent semantic analysis, *Uncertainty in Artificial Intelligence*.

Hofmann, T. and Puzicha, J. (1998). Unsupervised learning from dyadic data, *Technical Report TR-98-042*, International Computer Science Institute.

Hunt, L. A. and Jorgensen, M. A. (1999). Mixture model clustering: A brief introduction to the MULTIMIX program, *Australian and New Zealand Journal of Statistics* **40**: 153–171.

Jefferys, W. and Berger, J. (1992). Ockham's razor and Bayesian analysis, *American Scientist* **80**: 64–72.

Kleinberg, J. (1998). Authoritative sources in a hyperlinked environment, *Nineth Annual ACM-SIAM Symposium on Discrete Algorithms.*, Vol. 15.

Krzanowski, W. J. (1983). Distance between populations using mixed continuous and categorical variables, *Biometrika* **70**(1): 235–243.

Lindsay, B. G. (1988). *Mixture Models: Theory, Geometry and Applications*, NSF-CBMS Regional Conference Series in Probability and Statistics, Alexandria, Virginia: Institute of Mathematical Statistics and the American Statistical Association.

Mackay, D. J. C. (1992). Bayesian interpolation, *Neural Computation* **4**(3): 415–447.

Marquardt, D. W. and Snee, R. D. (1975). Ridge regression in practice, *American Statistician* **29**(1): 3–20.

McLachlan, G. and Peel, D. (2000). *Finite Mixture Models*, Wiley Series in Probability and Statistics, New York.

Minka, T. P. (2000). Estimating a Dirichlet distribution, Unpublished. Department of Computer Science, MIT.

Narayanan, A. (1991). Maximum likelihood estimation of the parameters of the Dirichlet distribution, *Applied Statistics* **40**(2): 365–374.

Nigam, K., McCallum, A., Thrun, S. and Mitchell, T. (2000). Text classification from labeled and unlabeled documents using EM, to appear in Machine Learning.

Olkin, I. and Tate, R. F. (1961). Multivariate correlation models with mixed discrete and continuous variables, *Annals of Mathematical Statistics* **32**(2): 448–465.

Redner, R. A. and Walker, H. F. (1984). Mixture densities, maximum likelihood and the EM algorithm, *SIAM Review* **26**: 195–239.

Ronning, G. (1989). Maximum likelihood estimation of dirichlet distributions, *Journal of Statistical Computation and Simulation* **32**(4): 215–221.

Stephens, M. (1997). *Bayesian Methods for Mixtures of Normal Distributions*, PhD thesis, Department of Statistics, Oxford University, England.

Titterington, D. M., Smith, A. F. M. and Makov, U. E. (1985). *Statistical Analysis of Finite Mixture Distributions*, John Wiley and Sons, San Diego.