Webmining: Learning from the World Wide Web

Jan Larsen, Lars Kai Hansen, Anna Szymkowiak,
Torben Christiansen and Thomas Kolenda*

Department of Mathematical Modeling,
Richard Petersens Plads, Building 321
Technical University of Denmark
DK-2800 Kongens Lyngby, Denmark
Phone: +45 4525 3923,3889,
Email: jl,lkhansen@imm.dtu.dk, Web: eivind.imm.dtu.dk

Abstract: Automated analysis of the world wide web is a new challenging area relevant in many applications, e.g., retrieval, navigation and organization of information, automated information assistants, and e-commerce. This paper discusses the use of unsupervised and supervised learning methods for user behavior modeling and content-based segmentation and classification of web pages. The modeling is based on independent component analysis and hierarchical probabilistic clustering techniques.

Keywords: Webmining, unsupervised learning, hierarchical probabilistic clustering

1. Introduction

Webmining is an increasingly important and very active research field which adapts advanced machine learning techniques for understanding the complex information flow of the world wide web, see e.g., (Nigam 00, Weigend 99). Web data are fundamentally multimedia streams of text, sound, images, and various database information. While optimal information retrieval, navigation or organization requires mining of all media modalities, this paper focuses on textmining and user behavior modeling.

Textmining (Hansen 00, Landuaer 98) is used to categorize text according to topic, to spot new topics, and in a broader sense to create more intelligent searches, e.g., by WWW search engines. Textmining proceeds by pattern

*This research is supported by the Danish Research Councils through the THOR Center for Neuroinformatics, Distributed Multimedia Technologies and Applications program within Center for Multimedia, and the Signal and Image Processing for Telemedicine (SITE) program. We would like to thank Jens Hørlück for assistance and support in setting up the user behavior modeling project.

recognition based on text features, typically document summary statistics. While numerous high-level language models for extraction of text features exists, simple summary statistics are still preferred because they are compact representation and can be adapted automatically and continuously, without costly manual intervention of language expertise.

Modeling the user's behavior when navigating a web site is very relevant in e-commerce applications (Cooley 99, Mobasher 99, Pei 00, Shahabi 97, Spiliopoulou 99, Yan 96). User modeling can be divided in three levels of functionality: the first level concerns automatic segmentation of users who display similar behavior. Second level concerns automatic classification of users using expert annotations of identified user segments. The third, and most elaborate level, involves interactive web pages continuously adapted to the user's behavior. This paper addresses merely automatic segmentation.

Section 2. describes a probabilistic hierarchical clustering framework based on the generalizable Gaussian mixture (GGM) model, which is reviewed. In section 3. we discuss the use of the GGM for supervised learning. Section 4. presents webmining applications using the methods of Sections 2.–3. covering: classification of webpages, hierarchical segmentation of emails, and user behavior segmentation.

2. Hierarchical Probabilistic Clustering

2.1. Generalizable Gaussian Mixture Model

The Gaussian mixture model is a very flexible pattern recognition device, see, e.g., (Ripley 96) for a review. The K component Gaussian mixture density of a feature vector $\boldsymbol{x}$ of dimension d, is defined as

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = \sum_{k=1}^{K} P(k) p(\boldsymbol{x}|k, \boldsymbol{\theta}_k), \quad p(\boldsymbol{x}|k, \boldsymbol{\theta}_k) = \frac{\exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_k)\right)}{\sqrt{|2\pi \boldsymbol{\Sigma}_k|}} \tag{1}$$

where the component Gaussians are mixed with proportions $\sum_k P(k) = 1$, $P(k) \geq 0$, and $\boldsymbol{\theta}_k \equiv \{\boldsymbol{\Sigma}_k, \boldsymbol{\mu}_k\}$ is a parameter vector. The parameters are estimated from a set of examples $\mathcal{D} = \{\boldsymbol{x}_n | n = 1, \cdots, N\}$. Traditionally mixture densities are estimated using maximum likelihood (ML), e.g., through various expectation-maximization (EM) methods (Ripley 96). The (negative log-) likelihood cost function is defined by $S_N(\boldsymbol{\theta}) = \sum_{n=1}^{N} - \log p(\boldsymbol{x}_n|\boldsymbol{\theta})$ and $\widehat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} S_N(\boldsymbol{\theta})$ are the estimated parameters. The objective of modeling is to ensure that the generalization error, defined as the expected cost on independent data, $G(\widehat{\boldsymbol{\theta}}) = -\int \log p(\boldsymbol{x}|\widehat{\boldsymbol{\theta}}) p^\circ(\boldsymbol{x})\, d\boldsymbol{x}$ is minimal. Here $p^\circ(\boldsymbol{x})$ denotes the "true" density.

The Gaussian mixture model is extremely flexible and simply minimizing the above cost function will lead to an "infinite overfit". This solution is optimal for the training set, but unfortunately has a generalization error roughly equal to that of the single component Gaussian model, as the singular components have zero measure w.r.t. test data[1]. This instability has lead to much confusion in the literature and needs to be addressed carefully. Basically, there is no way to distinguish generalizable from non-generalizable solutions if we only consider the likelihood function. The only way to ensure generalizability is to invoke the concept of generalization in the estimation procedure. The most common remedy is to bias the distributions so that they have a common shared covariance matrix, see e.g., (Hastie 96). In fact, classical EM algorithms only work under this assumption. A more principled method is to invoke regularization in terms of priors in a Bayesian framework (Rasmussen 00).

Here we adopt the Generalizable Gaussian Mixture model presented in (Hansen 00) which combines three approaches to ensure generalizability. First, we compute centers and covariances on different resamples of the data set. Secondly, we make an exception rule for sparsely populated components in which the covariance matrix defaults to the scaled full-sample covariance matrix. Thirdly, we estimate the number of mixture components by the AIC-criterion (Akaike 69, Hansen 96). The algorithm allows for individual component covariance matrices which enables a flexible local metric in contrast to methods assuming common covariance matrix, hence a global metric.

The Generalizable Gaussian Mixture algorithm is a modified EM procedure (Dempster 77) and is provided in Figure 1 for a fixed number of mixture components, K.

2.2. Hierarchical Clustering

There are numerous contributions within hierarchical clustering (see e.g., (Ripley 96)). Here the focus is to construct a relatively simple agglomerative hierarchical clustering using a probabilistic model which is based on the work in (Szymkowiak 01). For recent approaches to full hierarchical probabilistic clustering techniques the reader is referred to (Vasconcelos 99, Williams 00).

Define $p_j(\boldsymbol{x}|k)$ as the conditional probability[2] density of $\boldsymbol{x}$ for cluster $\mathcal{C}_k^j$ $k = 1, 2, \cdots, K - j + 1$ in layer $j = 1, 2, \cdots, K$ of a hierarchy. Further define $P_j(k)$ as the priors of the clusters (mixing proportions). At the most detailed level $j = 1$, the density is modeled by the GGM described above, i.e., $p_1(\boldsymbol{x}|k)$ are Gaussian densities. At each consecutive level two clusters with minimum distance are merged until we reach one cluster at level $j = K$. As distance measure we suggest to use the symmetric Kullback-Leibler divergence[3] between

[1]The cost function has a trivial (infinite) minimum attained by setting $\boldsymbol{\mu}_k = \boldsymbol{x}_k$ for $k = 1, \cdots, K - 1$, and $\boldsymbol{\Sigma}_k = \boldsymbol{0}$. The remaining K'th Gaussian is adapted to the remaining $N - K + 1$ data points, with $\boldsymbol{\mu}_K = (N - K + 1)^{-1} \sum_{n=K}^{N} \boldsymbol{x}_n$, and $\boldsymbol{\Sigma}_K = (N - K + 1)^{-1} \sum_{n=K}^{N} (\boldsymbol{x}_n - \boldsymbol{\mu}_K)(\boldsymbol{x}_n - \boldsymbol{\mu}_K)^\top$.

[2]For notation convenience, we omitted the condition on the model parameters in what follows.

[3]See e.g., (Ripley 96) for the classical Kullback-Leibler definition.

Figure 1: *Generalizable Gaussian Mixture Algorithm.*

```
Initialization for $K$ components
    1. Compute the mean vector $\boldsymbol{\mu}_0 = N^{-1} \sum_n \boldsymbol{x}_n$.
    2. Compute the covariance matrix of the data set:
       $\boldsymbol{\Sigma}_0 = N^{-1} \sum_n (\boldsymbol{x}_n - \boldsymbol{\mu}_0)(\boldsymbol{x}_n - \boldsymbol{\mu}_0)^\top$.
    3. Initialize $\boldsymbol{\mu}_k \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$.
    4. Initialize $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}_0$.
    5. Initialize $P(k) = 1/K$.
Repeat until convergence
    1. Compute $p(k|\boldsymbol{x}_n) = p(\boldsymbol{x}_n|k)p(k)/\sum_\ell p(\boldsymbol{x}_n|\ell)p(\ell)$ and assign $\boldsymbol{x}_n$
       to the most likely component.
    2. Split the data set in two parts $\mathcal{D}_{\boldsymbol{\mu}}$, $\mathcal{D}_{\boldsymbol{\Sigma}}$. Often 50/50 splitting is
       used.
    3. For each $k$ estimate $\boldsymbol{\mu}_k$ on the points in $\mathcal{D}_{\boldsymbol{\mu}}$ assigned to compo-
       nent $k$.
    4. For each $k$ estimate $\boldsymbol{\Sigma}_k$ on the points in $\mathcal{D}_{\boldsymbol{\Sigma}}$ assigned to compo-
       nent $k$. If the number of data points assigned to the $k$'th compo-
       nent, $N_k$, is less than $d+1$, then $\boldsymbol{\Sigma}_k \leftarrow (N_k \boldsymbol{\Sigma}_k + \boldsymbol{\Sigma}_0)/(N_k+1)$.
    5. Estimate $P(k)$ as the frequency of assignments to component $k$.
```

the mixture components, as defined by

$$D(k_1, k_2) = \frac{1}{2} \int p(\boldsymbol{x}|k_1) \log \frac{p(\boldsymbol{x}|k_1)}{p(\boldsymbol{x}|k_2)}\, d\boldsymbol{x} + \frac{1}{2} \int p(\boldsymbol{x}|k_2) \log \frac{p(\boldsymbol{x}|k_2)}{p(\boldsymbol{x}|k_1)}\, d\boldsymbol{x} \qquad (2)$$

For layer $j = 1$ in which the cluster densities are Gaussian the distance can be expressed as (Szymkowiak 01):

$$\begin{aligned} D_1(k_1, k_2) = & -\frac{d}{2} + \frac{1}{4}\left(\mathrm{Tr}[\boldsymbol{\Sigma}_{k_1}^{-1}\boldsymbol{\Sigma}_{k_2}] + \mathrm{Tr}[\boldsymbol{\Sigma}_{k_2}^{-1}\boldsymbol{\Sigma}_{k_1}]\right) + \\ & \frac{1}{4}(\boldsymbol{\mu}_{k_1} - \boldsymbol{\mu}_{k_2})^\top(\boldsymbol{\Sigma}_{k_1}^{-1} + \boldsymbol{\Sigma}_{k_2}^{-1})(\boldsymbol{\mu}_{k_1} - \boldsymbol{\mu}_{k_2}) \end{aligned} \qquad (3)$$

When proceeding from level j to $j+1$ suppose that clusters $\mathcal{C}_{k_1}^j$ and $\mathcal{C}_{k_2}^j$ are merged. Then the merged density of cluster $\mathcal{C}_k^{j+1}$ at level $j+1$ is a mixture given by:

$$p_{j+1}(\boldsymbol{x}|k) = \frac{P_j(k_1)p_j(\boldsymbol{x}|k_1) + P_j(k_2)p_j(\boldsymbol{x}|k_2)}{P_j(k_1) + P_j(k_2)}, \quad P_{j+1}(k) = P_j(k_1) + P_j(k_2) \quad (4)$$

The remaining densities are unchanged.

At level 1 the expression for the distance in Eq. (4) is exact, while exact calculation at other levels cannot be cast into a simple analytical form. Consequently, we suggest to use a simple combination rule in which the distances to a merged cluster is original distances weighted by the mixing proportions, as in Eq. (4), i.e., $D_{j+1}(k, \ell) = (P_j(k_1)D_j(k_1, \ell) + P_j(k_2)D_j(k_2, \ell))/(P_j(k_1) + P_j(k_2))$, where clusters $\mathcal{C}_{k_1}^j, \mathcal{C}_{k_2}^j$ have been merged into $\mathcal{C}_k^{j+1}$ at level j, and ℓ indexes a cluster at level $j+1$.

Using a Bayes optimal decision strategy (assuming simple 0/1 loss function, see e.g., (Ripley 96)), a specific training example $\boldsymbol{x}_n$ is assigned to cluster k if

$$k = \arg\max_{\ell} P_j(\ell|\boldsymbol{x}_n) = \arg\max_{\ell} \frac{p_j(\boldsymbol{x}|\ell)P_j(\ell)}{\sum_{i=1}^{K-j+1} p_j(\boldsymbol{x}|i)P_j(i)} \tag{5}$$

If clusters $\mathcal{C}_{k_1}^j, \mathcal{C}_{k_2}^j$ have been merged into $\mathcal{C}_k^{j+1}$ at level j, then $P_{j+1}(k|\boldsymbol{x}_n) = P_j(k_1|\boldsymbol{x}_n) + P_j(k_2|\boldsymbol{x}_n)$. Thus, all posterior cluster probabilities are easily computed from the level 1 posteriors $P_1(k|\boldsymbol{x}_n)$.

Once the hierarchy is constructed we want to determine cluster/level membership of new examples. For this purpose we chose the following criterion: If $P_j(k|\boldsymbol{x}) = \arg\max_\ell P_j(\ell|\boldsymbol{x}) > \rho$ then $x \in \mathcal{C}_k^j$, where $\min_k P_1(k) < \rho \leq 1$ is a prescribed threshold, e.g., $\rho = 0.9$. This corresponds to accepting that $\boldsymbol{x}$ is assigned to a wrong cluster in with probability 0.1.

2.3. Interpretation of Clusters

Interpretation of clusters in the hierarchy is important for webmining applications. Suppose that each original example in our database is a set of elements drawn from finite number of possible elements (often large). Each example could for instance be a html-document consisting of a number of elements, i.e., words from a large vocabulary. The set of elements of each example is encoded into the feature vector $\boldsymbol{x}$. Basically two methods exist for a cluster interpretation: The first consist in listing a number of representative examples from the available training data set which are member of the cluster to be interpreted. The second method consists in listing typical elements associated with the cluster.

2.3.1. Prototype Examples

Representative examples of a specific cluster can be defined as the ones which are most probable. Since $p(\boldsymbol{x}|k)$ is a probability density the values are not directly comparable. Instead we compute the probability[4] $Q(t) = \mathrm{Prob}(\boldsymbol{x} \in \mathcal{R})$, $\mathcal{R} = \{\boldsymbol{x} : p(\boldsymbol{x}|k) < t\}$, for all thresholds t. We aim at identifying the t-value corresponding to the most probable example for the major part of the probability mass. This value is found as $t_{\max} = \arg\max_t Q(t) \leq Q_{\max}$, where that $Q_{\max}$ is a high threshold, e.g., 0.9. Practically, $Q(t)$ is computed from the training data assigned to cluster k, say $\mathcal{D}_k = \{\boldsymbol{x}_n \in \mathcal{C}_k\}$, as follows: rank $t_n = p(\boldsymbol{x}_n|k)$, $\boldsymbol{x}_n \in \mathcal{D}_k$ in ascending order, $t_1 \leq t_2 \leq \cdots \leq t_{N_k}$, where $p(\boldsymbol{x}_n|k)$ are model density values, and $N_k = |\mathcal{D}_k|$ is the number of example in $\mathcal{D}_k$. Finally, let $Q(t_n) = n/N_k$. Prototype examples are then a number of high ranked examples having t_n near $t_{\max}$.

[4]This idea relates to the concept of highest probability density regions (Box 92, Ch. 2.8).

2.3.2. Prototype Elements

In order to list representative elements associated with a cluster we start by finding most probable feature vectors from each cluster, basically using the method described in the previous section. An large surrogate data set can be generated by drawing Monte Carlo random samples from the estimated Gaussian mixture. From these data typical feature vectors are those having t-values for which $Q(t)$ is sufficiently high. Finally, the generated feature vectors are back-projected into original element space.

2.3.3. Novelty Detection

When the estimated density model is applied to new data there is a risk that these can not meaningfully be described by the model; in other words, we need to address the novelty problem. In line with recent work (Baker 99, Bishop 94, Nairac 97, Basseville 93), we suggest a novelty detector based on total input density $p(\boldsymbol{x})$. The method described in Section 2.3.1. can be used to form a $Q(t)$-function for $p(\boldsymbol{x})$. We then set a low threshold $Q_{\min}$ and find the corresponding $t_{\min}$ as $t_{\min} = \arg\min_t Q(t) \geq Q_{\min}$. Finally, novel events are detected as those having density values less than $t_{\min}$.

3. Generalizable Gaussian Mixture Classifier

If the feature vectors $\boldsymbol{x}$ are annotated by providing class labels, we are able to perform supervised learning using the GGM model. Consider a data set $\mathcal{D} = \{(\boldsymbol{x}_n, c_n) \mid n = 1, 2, \cdots, N\}$ where $c_n \in \{1, 2, \cdots, C\}$ is the class associated with example n. The joint density of feature vectors $\boldsymbol{x}$ and class labels c is $p(\boldsymbol{x}, c) = p(\boldsymbol{x}|c)P(c)$, where $p(\boldsymbol{x}|c)$ is the class conditioned density and $P(c)$ is the marginal class probabilities. The classifier is designed by adapting GGM's to each class separately. Hence, the class conditional density can be written as $p(\boldsymbol{x}|c) = \sum_{k=1}^{K_c} p(\boldsymbol{x}|k, c)P(k|c)$, where $P(k|c)$ and K_c are the mixture component probabilities and number components used for class c, respectively.

Labels are assigned to a new data point in accordance with the optimal Bayes classification (under the 0/1 loss) rule by selecting the maximum posterior probability, $P(c|\boldsymbol{x}) = p(\boldsymbol{x}|c)P(c)/\sum_{c=1}^{C} p(\boldsymbol{x}|c)P(c)$.

3.1. Unsupervised-then-Supervised Gaussian Mixture Model

In (Nigam 00) the interplay between supervised and unsupervised learning was discussed. To estimate the role of the labels for the GGM model first perform an GGM input density estimate $p(\boldsymbol{x}) = \sum_{k=1}^{K} P(k)p(\boldsymbol{x}|k)$. Next estimate $P(c|k)$ for each component k from the joint feature/label training data set as N_{ck}/N_k, where N_{ck} is the number of data samples of component k assigned class label c, and N_k is the number of data samples of component k. Finally, estimate the conditional class probability by

$$P(c|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|c)P(c)}{p(\boldsymbol{x})} = \frac{\sum_{k=1}^{K} p(\boldsymbol{x}|k,c)P(c|k)P(k)}{p(\boldsymbol{x})} = \frac{\sum_{k=1}^{K} p(\boldsymbol{x}|k)P(c|k)P(k)}{p(\boldsymbol{x})}. \quad (6)$$

The classification of examples using Eq. (6) can be compared to that of the supervised GGM classifier, illustrating the role of labels during training.

4. Experiments

4.1. Classification of Web Pages

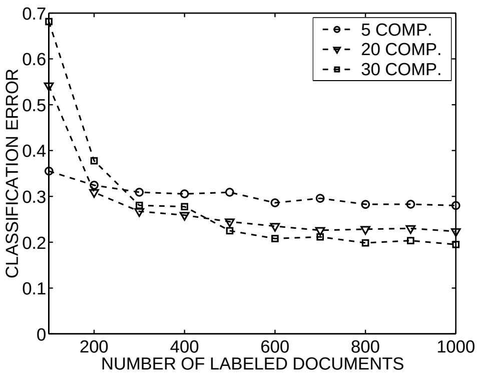
The focus is on understanding the textual content of a web page based on statistical features. Here we consider the single word statistics; frequency of word occurrence, hence disregarding order and association. Word frequencies have been used in the vector space model (Luhn 58, Salton 89) for decades. In practice words which high and low frequencies have little discriminative power. High frequency words are typically function words, e.g., is and the. Such words are removed by comparing the document with a list of stop words, i.e., a dictionary of common words. Also low frequency words are removed since they do not represent any common meaning among a number of web pages. In addition, we will consider to remove words with common stem, i.e., words like worked and working are represented by their stem work. Typically the number of words/terms after such parsing is still a very large compared to the number of documents available for learning. Since learning algorithms often fail to generalize in high dimensions there is a need for efficient and robust means for data reduction and feature extraction. Latent Semantic Indexing (LSI) (Deerwester 90) is a method to generate a reasonable low dimensional feature vector, and is further believed to handle polysemy and synonomy problems. Polysemy refers to the problem that words often have more than one meaning, whereas synonomy refers to the problem of different words with similar meaning.

LSI is based on the $T \times N$ term-document matrix, $\boldsymbol{Z} = [\boldsymbol{z}_1, \cdots, \boldsymbol{z}_N]$, where $\boldsymbol{z}_n$ represent term frequency of document n, i.e., z_{in} is the probability of term i in document n.[5] The term frequencies are projected on a orthogonal set of eigen-histograms found by singular value decomposition (SVD). LSI can aid interpretation by visualizing group structure in the set of documents, typically by scatter plots of the term histograms on a reduced set of salient eigen-histograms. Another virtue of this representation is that it can be used as a dimensionality reduction scheme. First we remove the mean value $\bar{\boldsymbol{z}}_n = \boldsymbol{z}_n - \widehat{\boldsymbol{\mu}}$, where $\widehat{\boldsymbol{\mu}} = N^{-1}\sum_{n=1}^{N} \boldsymbol{z}_n$. Then the SVD is given by $\bar{\boldsymbol{Z}} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^\top = \sum_{i=1}^{R} \boldsymbol{u}_i D_{i,i} \boldsymbol{v}_i^\top$, where the $T \times R$ matrix $\boldsymbol{U} = \{u_{ti}\} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_R]$, with R being the rank[6] of $\boldsymbol{Z}$, and the $N \times R$ matrix $\boldsymbol{V} = \{v_{ni}\} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_R]$

[5]The probabilities as normalized so that $\sum_i z_{in} = 1$.

[6]Since $T \gg N$, then for independent documents the rank is $R = N$.

Figure 2: *Learning curves for supervised learning of the generalizable Gaussian mixture classifier using WebKB data set.*



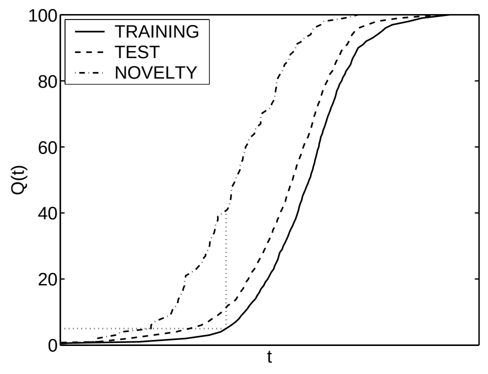
Confusion Matrix
$d = 30$ and $N_{\text{train}} = 1000$

	True			
Estimated	Course	Fac.	Proj.	Stud.
Course	0.92	0.03	0.05	0.02
Fac.	0.04	0.64	0.10	0.13
Proj.	0.03	0.09	0.75	0.02
Stud.	0.01	0.24	0.10	0.83

represent the orthonormal basis vectors (i.e., eigenvectors of the symmetric matrices $\boldsymbol{X}\boldsymbol{X}^\top$ and $\boldsymbol{X}^\top\boldsymbol{X}$, respectively). $\boldsymbol{\Lambda} = \{\lambda_{ii}\}$ is a $R \times R$ diagonal matrix of singular values ranked in decreasing order. Many singular values will be small and are regarded as artifacts or noise. Consequently, the subspace associated with these should be omitted while maintaining the latent semantic structure. The projection onto the d dimensional latent subspace is given by $\boldsymbol{X} = \widetilde{\boldsymbol{U}}^\top \bar{\boldsymbol{Z}}$, $\widetilde{\boldsymbol{U}} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_d]$.

The CMU WebKB repository (CMU homepage) consist of 2240 web pages labeled according to the following categories: Course (24.7%), Faculty (21.6 %) Project (15.7%), Student (38.0%). A term list of 13071 words that occurred in two or more documents was defined without screening for stop-words. Latent semantic analysis is performed using feature dimensions of $d = 5, 20, 30$. In Figure 2 learning curves for the GGM classifier Section 3. were estimated by cross-validation. Data are randomly split 10 times into a test set of ($N_{\text{test}} = 1240$) and training sets of increasing sizes, $N_{\text{train}} = 100$–1000. Learning curves were estimated as the averaged test error as a function of d. A generalization cross-over, as function of the dimension, is noticed, i.e., the larger dimensional representations requires more samples to generalize. The proposed GGM classifier achieves classification rates and learning curves comparable to those found in (Nigam 00). The GGM model, however, achieves this performance based on the full 13071 dimensional term-frequency showing the strength of Latent Semantic Analysis representation. This allows for handling more complex webmining problems and also avoiding the selection of terms as in (Nigam 00). The interplay between supervised and unsupervised learning was further addressed in (Nigam 00). To estimate the role of the labels for the GGM model, we have carried out a similar learning curve experiment for the unsupervised-then-supervised Gaussian mixture model Section 3.1. It turns out that learning is much less efficient for the unsupervised-then-supervised procedure indicating significant class overlap.

Figure 3: *Novelty detection using web 173 pages from the* Department *group of the WebKB data set. The model has $d = 30$ dimensions and both the training and test sets contained 1120 documents. Threshold t for $p(\boldsymbol{x})$ is selected for $Q = 5\%$.*



4.1.1. Novelty Detection

Since the GGM classifier produces conditional probabilities we obtain in this way a clue to the "internal" confidence. The magnitude of the probabilities is determined by proximity of the decision boundary of the closest competing class. The overall test error rate give a clue to our confidence in the probabilities obtained from the system. However, when applied to new data the possibility exist, of course, that the new data can not in a meaningful way be assigned to any of the classes in the training data. In other words we need to address the novelty problem by identifying outliers in $p(\boldsymbol{x})$ as described in Section 2.3.3. Figure 3 shows $Q(t)$ based on training and a test set gathered from the documents above. We note that the test data are not rejected at reasonable Q-levels. The third curve is obtained from a third independent set of documents Department not related in an obvious way to the training and test sets. This data is declared novelty at levels below $Q_{\text{min}} = 5\%$.

4.1.2. Web Navigation

A possible application is a navigation tool that can assist the user by combining the supervised and unsupervised classification schemes. At first the supervised part uses a list of labeled web pages, as typically can be found in a bookmark/favorite list ordered in folders for which the folder name serves as label for the underlying web pages (links). The GGM classifier classifies new pages into known bookmark labels. Documents not qualifying w.r.t. the current list of topics are detected as novel and using unsupervised GGM clustering of the pages and evaluating representative keywords for each mixture component, we are able to get an overall description of the document. Keywords are generated by back-projecting cluster centers into term-frequency space and then selecting most probable terms. Using e.g., Other/Misc pages of the WebKB data set 40% of the pages in this group are detected as novel, and these were subsequently clustered into 4 new groups. Keywords suggested the 4 groups could be interpreted as: Places, Spare time, Computer systems and Multimedia as indicated by Table 1.

Table 1: *Keywords associated with novel WebKB group* Other/Misc.

Multimedia	Computer sys.	Spare time	Places
eros	up	page	mississippi
random	readme	webteam	detroit
np	cache	visits	university
u	incoming	funny	military
player	msdos	uva	saint
ramifications	directory	today	macon
gif	windows	museum	williamsburg
format	mac	totals	rolla
slide	unix	robins	aeronautical
modulo	wie	total	louis

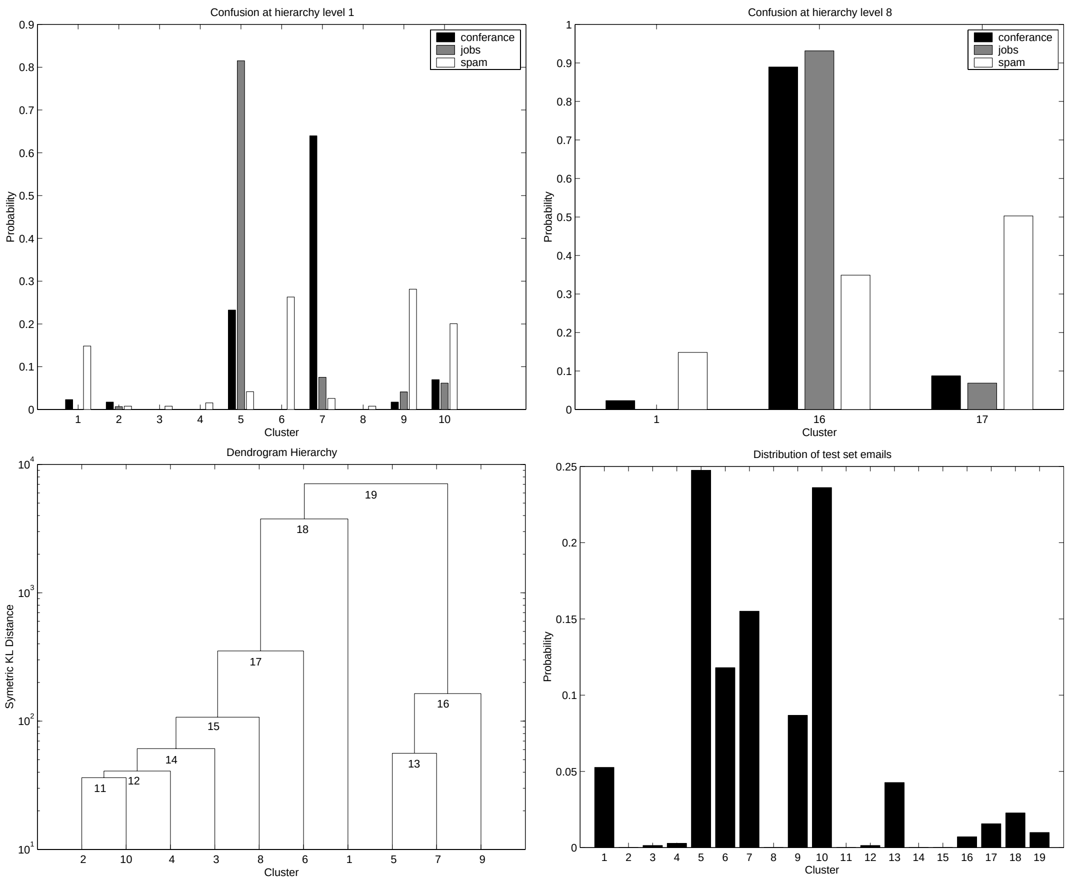
4.2. Email Segmentation using Hierarchical Probabilistic Clustering

Consider hierarchical segmentation of emails. A database of 1443 English emails categorized in three groups conference, jobs, and spam were collected. Only the text contained in subject and body was considered. As in Section 4.1. we performed LSI using a stop word list of 577 words, removed words which occurred less than 4 times, and finally we discarded emails which contained less than 2 words. Only one word for words with a common stem was maintained by discarding 13 different endings. After preprocessing we had 1405 emails divided into 702 for training and 703 for testing. Each email was represented by it's term-histogram of 7798 terms. Using a latent subspace of $d = 30$ components[7] resulted in GGM models with optimal number of clusters in level 1 in the range 6–10. We chose to illustrate a model consisting of $K = 10$ clusters. Performing hierarchical clustering on top of the GGM, as described in Section 2.2., results in a dendrogram hierarchy depicted in lower left panel of Figure 4. Numbers refer to cluster numbers, e.g., 12 is the merging of clusters 4 and 11. The confusion matrices computed from training examples for hierarchy levels 1 and 8 are shown in the upper panels of Figure 4. It is noted that at level 1 the conference category is mainly represented by cluster 7 and 5, jobs by cluster 5, and spam by clusters 9, 6, 10 and 1. At level 8, corresponding to three clusters, clusters 1 and 17 mainly represent spam whereas cluster 16 mainly represents both conference and jobs. Consequently, the unsupervised hierarchical clustering is not able to distinguish these categories. Also notice that cluster 5 and 7 which largely represent these categories are merged at an early level into cluster 13. For comparison, supervised learning was also implemented. As expected, it performs much better regarding cluster separation. Then confusion on the first level of hierarchical clustering

[7] A method for selecting the subspace dimension based on generalization error in described in (Szymkowiak 01).

was much smaller comparing to the unsupervised. However, since the goal of the algorithm is to extract hidden common sense in the text documents, the exact classification can be misleading. In the tested database the clustering algorithm seems to confuse big parts of the conference and jobs group. This happens both for the unsupervised and supervised learning algorithm. The KL divergence measure, Eq. (4), indicates a small distance in the probabilistic space between these two clusters, and the generated keywords (see Table 2) are closely related which explains the small distance. When filtering test set

Figure 4: *Dendrogram for hierarchical email clustering and distribution of test set emails among clusters.*



emails through the hierarchy we assign a specific email to the cluster at which the posterior probability is above 0.9, according to Section 2.2. The right lower panel of Figure 4 shows the fraction of test set emails ending up in different clusters. We notice that several email first obtain a meaningful interpretation at high level in the hierarchy (i.e., cluster number larger than 10).

Keywords are generated by back-projecting most probable features from each cluster at any level in the hierarchy as outlined in Section 2.3. The back-projection intro term-frequency space is given by $\bar{\boldsymbol{z}} = \widetilde{\boldsymbol{U}}\boldsymbol{x}$, where $\boldsymbol{x}$ is a probable feature vector and $\widetilde{\boldsymbol{U}}$ is the 1405×30 projection matrix. The keywords are then found as the most likely terms, i.e., highest values[8] of $\bar{\boldsymbol{z}}$.

Table 2: *Keywords for email cluster hierarchy in Figure 4.*

Cluster	Keywords
1	free address government
2	fax
3	subject future remove computer
4	adult free
5	fax interest computation computer web science position research university
6	good mac
7	fax year message call conference information computer address
8	call girl
9	good year receive product special make month day future mail friend quick line state send offer
10	government remove adult
11	action address check hottest click site creativity call website government web free remove mac adult
12	hey jessica site remove call government creativity web website mac adult fax free
13	conference send information application computation year interest science address call fax computer
14	website mac adult remove fax free computer
15	website revolution remove fax free call adult girl computer
16	year interest computation fax address science web computer position university research
17	mail government subject creativity call website free future fax food adult remove computer
18	food web free mac government adult
19	message fax computer science list call subject university money position information address dear

4.3. User behavior modeling

User behavior modeling is an important aspect of e-commerce systems. The current examples is based on our work reported in (Christiansen 01) which studied an e-commerce company selling articles via the web. Web log-data was recorded for half a year and resulted in 31700 sessions for which all user actions where mapped into 60 unique events. Events could be pressing a buy button, selecting a certain group of articles, or following a link to a another web page. Each session is thus a variable length sequence of events from the 60 element event-alphabet $\mathcal{B} = \{1, 2, \cdots, 60\}$, $B = |\mathcal{B}| = 60$. In general, it might be difficult to map the details of the web server log file into a unique event space unless the logging has been designed with this purpose in mind.

The log-on to the site could be done in two ways, either as member login with personal password, or as a guest assigned a pseudo user-id. Each session was numbered in succession, i.e., repeated log-on from the same user is mapped to different session numbers. Using the industry standard, sessions are interrupted and the user automatically logged-off after 30 minutes of no activity.

Too short sessions will not reflect a real interest in the web site (Yan 96). Hence, the minimum session length was set to four events, corresponding to the shortest way into the "shopping area" from the opening site. A total of 4339 sessions remained of which 1089 randomly was selected as test set, leaving

[8]Due to using a low-dimensional subspace of $d = 30$, $\bar{\boldsymbol{z}} + \widehat{\boldsymbol{u}}$ typically does take values in the range $[0; 1]$ nor is $\sum_i \bar{z}_i + \widehat{u}_i = 1$. In principle, we could feed the values trough a softmax-function (Ripley 96), which, however, will not change the ranking.

3250 sessions for training.

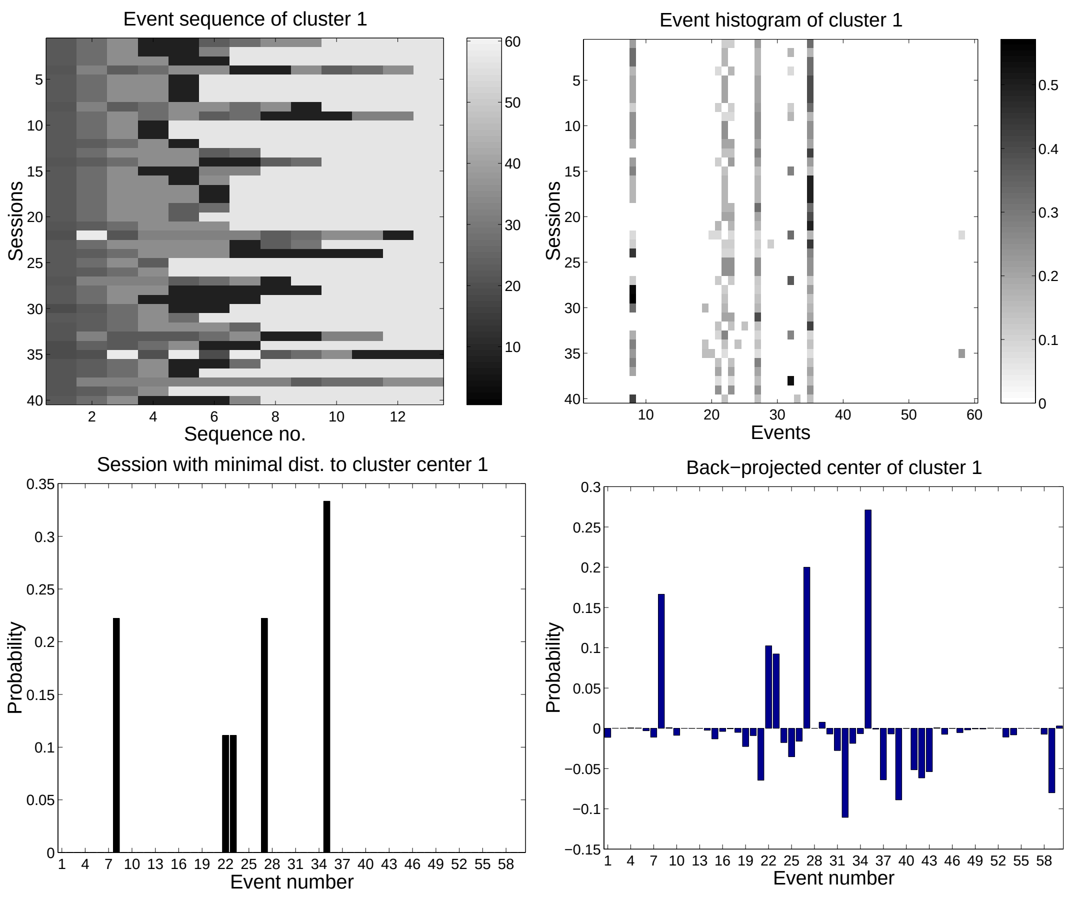
Let $s_{\ell n} \in \mathcal{B}$ represent session n consisting of L_n events, $\ell = [1; L_n]$. As in (Yan 96), we deploy histogram statistics representation of the sessions by computing the frequency of events: $z_{in} = L_n^{-1} \sum_{\ell=1}^{L_n} \delta(i - s_{\ell n})$, where $i \in \mathcal{B}$, $\delta(\cdot)$ is the Kronecker delta-function, and $\boldsymbol{Z} = [\boldsymbol{z}_1, \cdots, \boldsymbol{z}_n]$ is denoted the histogram matrix. It is possible to use second order statistics, i.e., co-occurrence matrices. The $B \times B$ co-occurrence matrix for session n and displacement τ is defined as, $c_{ij}(n, \tau) = (L_n - 1)^{-1} \sum_{\ell=1}^{L_n - 1} \delta(i - s_{\ell,n}) \cdot \delta(j - s_{\ell+\tau,n})$, $\forall i, j \in \mathcal{B}$, and expresses the frequency of events i and j in distance τ of the sequence. Co-occurrence features have be used in e.g., (Faisal 99) and will be further addressed in (Christiansen 01). In this study we merely address the use of the histogram and also neglect to include the duration of a session as a feature. In order to obtain a compact feature space we apply singular value decomposition (see p. 7) of the zero mean $B \times N$ histogram matrix $\bar{\boldsymbol{Z}} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^\top$ defined by $\bar{\boldsymbol{z}}_n = \boldsymbol{z}_n - \widehat{\boldsymbol{u}}$, where $\widehat{\boldsymbol{u}} = N^{-1} \sum_{n=1}^{N} \boldsymbol{z}_n$. Then we project onto the d-dimensional latent subspace spanned by the largest singular values as given by $\boldsymbol{X} = \widetilde{\boldsymbol{U}}^\top \bar{\boldsymbol{Z}}$, where $\widetilde{\boldsymbol{U}} = [\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_d]$.

Repeated training of the unsupervised GGM model using $d = 30$ features resulted in that the most generalizable model contained $K = 17$ components (clusters). Figure 5 shows the obtained analysis of cluster 1. The upper left panel shows the event sequences of the 40 sessions belonging to cluster 1, and are quite similar for the first few instances in the sequence. The upper right panel shows event histograms, and obviously most sessions use a rather limited number of events. In the lower panel the interpretation of cluster 1 is illustrated. The lower left panel shows the histogram of most the probable session, whereas the lower right panel shows the back-projection of the cluster center to histogram space. There is a significant resemblance indicating that the cluster can be interpreted by events (ordered in decreasing importance) as: $35, 27, 8, 22, 23$. From the actions associated with these events it seems that the cluster represents users attempting to register as a new members, while none of the users are able to get to the shopping web page. Other clusters can be interpreted using this technique. For instance, cluster 3 represents members who first login as guests, secondly choose a goods pick-up store, and then browse for while. However, almost 200 out of 708 in this cluster decide to quit after having watched the entry shopping web page. Cluster 15 represents a group of users which are not able to use the site correctly. They try use a search function before selecting preferred goods pick-up store, which turns out to be impossible. This way cluster 15 reveals a simple bug in the web site design.

5. Conclusion

This paper discussed the use of unsupervised and supervised methods for analysis and interpretation of world wide web data. A hierarchical probabilistic

Figure 5: *User behavior modeling. Analysis of cluster 1.*



clustering scheme based on the generalizable Gaussian mixture (GGM) model was described. In addition, methods for interpretation of the identified clusters were presented. The use of the GGM for supervised and unsupervised-then-supervised classification was also discussed. We successfully applied supervised GGM for classification of web pages. The unsupervised GGM was applied for hierarchical probabilistic clustering of emails and segmentation of user's behavior when shopping on a web site.

References

Akaike H. (1969) Fitting Autoregressive Models for Prediction, *Ann. of the Inst. of Stat. Math.*, vol. 21, pp. 243–247.

Baker L.D., Hofmann T., Maccallum A.K., Yang Y. (1999) A Hierarchical Probabilistic Model for Novelty Detection in Text, *CMU techincal report*, `http://www.cs.cmu.edu/People/mccallum/papers/tdt-nips99s.ps.gz`

Basseville M., Nikiforov I.V. (1993) *Detection of Abrupt Changes: Theory and Application*, Prentice-Hall.

Bishop C.M. (1994) Novelty Detection and Neural Network Validation, *IEE Proceedings - Vision Image and Signal Processing*, vol. 141, no. 4, pp. 217–222.

Box G.E.P., Tiao G.C. 1992 *Bayesian Inference in Statistical Analysis*, John Wiley & Sons.

Cooley R., Srivastava J., Mobasher, B. (1999) Data Preparation for Mining World Wide Web Browsing Patterns, *Journal of Know. and Inf. Sys.*, vol. 1, no. 1, pp. 5–32.

Christiansen T., Larsen J., Hansen L.K., Hørlück J. (2001) Understanding User Behavior on Web Sites. Forthcoming publication.

Carnegie Mellon University homepage, `http://www.cs.cmu.edu/~textlearning`

Deerwester S., Dumais S.T., Furnas G.W., Landauer T.K., Harshman R. (1990) Indexing by Latent Semantic Analysis, *Journ. Amer. Soc. for Inf. Science.*, vol. 41, pp. 391–407.

Dempster A.P., Laird N.M., Rubin D.B. (1977) Maximum likelihood from incomplete data via the EM algorithmm, *Jour. R. Stat. Soc. B*, vol. 39, pp. 1–38.

Faisal A., Shahabi C., McLaughlin M., Betz F. (1999) A Generic Paradigm for Interpreting User-Web Space Interaction, in *Proc. of Web Inf. and Data Management (WIDM'99)*, Kansas City, USA, pp. 53–58.

Hansen L.K., Sigurdsson S., Kolenda T., Nielsen F.Å., Kjems U., Larsen J. (2000) Modeling Text with Generalizable Gaussian Mixtures, *Proeedings of IEEE ICASSP'2000*, Istanbul, Turkey, vol. VI, pp. 3494–3497.

Hansen L.K., Larsen J. (1996) Unsupervised Learning and Generalization, *Proc. of the IEEE Int. Conf. on Neural Networks 1996*, vol. 1, pp. 25–30.

Hastie T., Tibshirani R. (1996) Discriminant Analysis by Gaussian Mixtures, *Jour. Royal Stat. Society - Series B*, vol. 58, no. 1, pp. 155–176.

Landauer T.K., Laham D., Foltz P. (1998) Learning Human-like Knowledge by Singular Value Decomposition: A Progress Report, *Advances in NIPS 10*, MIT Press, pp. 45–51.

Luhn H.P. (1958) The Automatic Creation of Literature Abstracts, *IBM Journal of Research and Development*, vol. 2, no. 2, pp. 159–165 and 317.

Mobasher B., Cooley R., Srivastava J. (1999) Creating Adaptive Web Sites Through Usage-Based Clustering of URLs, in *Proceedings KDEX'99.*

Nairac A., Corbett-Clark T., Ripley R., Townsend N., Tarassenko L. (1997) Choosing An Appropriate Model for Novelty Detection, *IEE 5th Int. Conf. on Art. NNs*, pp. 117–122.

Nigam K., McCallum A.K., Thrun S., Mitchell T. (2000) Text Classification from Labeled and Unlabeled Documents using EM, *Machine Learning*, vol. 39, no. 2–3, pp. 103–134.

Pei J., Han J., Mortazavi-Asl B., Zhu H. (2000) Mining Access Pattern Efficiently from Web Logs, *Proc. 2000 Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD'00)*, Kyoto, Japan, pp. 396-407.

Rasmussen, C.E. (2000) The Infinite Gaussian Mixture Model, in S.A. Solla, T.K. Leen K.-R. Müller (eds.) *Advances in NIPS 12*, MIT Press, pp. 554–560.

Ripley B.D. (1996) *Pattern Recognition and Neural Networks*, Cambridge University Press.

Salton G. (1989) *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer* Addison-Wesley.

Shahabi C., Zarkesh A.M., Adibi J., Shah V. (1997) Knowledge Discovery from Users Web-page Navigation, *Proceedings Seventh International Workshop on Research Issues in Data Engineering*, (cat.no. 97TB100122), pp. 20-29.

Spiliopoulou M., Faulstich L.C., Winkler K. (1999) A Data Miner Analyzing the Navigational Behaviour of Web Users, *Proc. of the Workshop on Machine Learning in User Modelling of the ACAI'99 Int. Conf.*, Creta, Greece.

Szymkowiak A., Larsen J., Hansen L.K. (2001) Hierarchical Clustering for Datamining. Forthcoming publication.

Yan T.W., Jacobsen M., Garcia-Molina H., Dayal U. (1997) ¿From User Access Patterns to Dynamic Hypertext Linking, *Computer Networks and ISDN Systems*, vol. 28, no. 11.

Vasconcelos N., Lippmann A. (1999) Learning Mixture Hierarchies, in M. Kearns, S. Solla & D.A. Cohn (eds.) *Advances in NIPS 11*, pp. 606–612.

Weigend A.S., Wiener E.D., Pedersen J.O. (1999) Exploiting Hierarchy in Text Categorization, *Information Retrieval*, vol. 1, pp. 193–216.

Williams C. (2000) A MCMC Approach to Hierarchical Mixture Modelling, in T. Leen, S. Solla & K.R. Müller (eds.) *Advances in NIPS 12*, pp. 680–686.