Hierarchical Dirichlet Model for Document Classification

Sriharsha Veeramachaneni SRIHARSHA@ITC.IT
Diego Sona SONA@ITC.IT
Paolo Avesani AVESANI@ITC.IT
SRA Division, Istituto per la ricerca scientifica e tecnologica (ITC-IRST), Trento, 38050, Italy

Abstract

The proliferation of text documents on the web as well as within institutions necessitates their convenient organization to enable efficient retrieval of information. Although text corpora are frequently organized into concept hierarchies or taxonomies, the classification of the documents into the hierarchy is expensive in terms human effort. We present a novel and simple hierarchical Dirichlet generative model for text corpora and derive an efficient algorithm for the estimation of model parameters and the unsupervised classification of text documents into a given hierarchy. The class conditional feature means are assumed to be inter-related due to the hierarchical Bayesian structure of the model. We show that the algorithm provides robust estimates of the classification parameters by performing smoothing or regularization. We present experimental evidence on real web data that our algorithm achieves significant gains in accuracy over simpler models.

1. Introduction

The organization of text corpora in hierarchies is a convenient approach for the management of the information comprised therein, since it enables increased efficiency of search, and accuracy of retrieval of documents relevant to the needs of the end user. Web directories are well known examples of the scenario, where the most relevant Web pages are classified into a set of categories structured as a hierarchy (e.g., the directories of LookSmart™, Yahoo!™, and Google™ with the Open Directory Project initiative[1]).

[1]An open source initiative aimed to promote a comprehensive Web directory <http://www.dmoz.org>

Appearing in *Proceedings of the 22nd International Conference on Machine Learning*, Bonn, Germany, 2005.

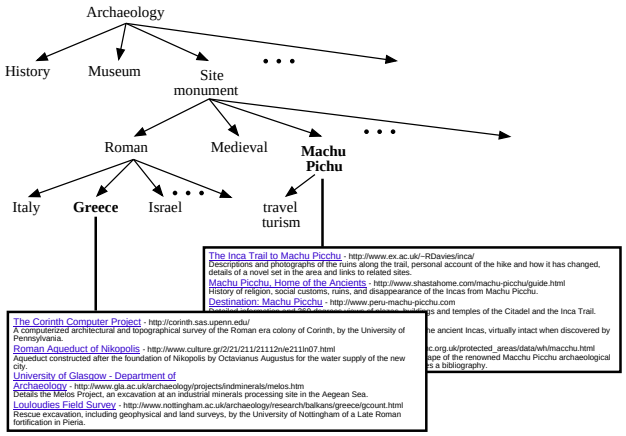

Figure 1. A snapshot of the sub-directory *Archaeology* extracted from Google. Each node is represented by labels and it contains some documents.

A concept taxonomy can be defined as a hierarchy (or tree) of categories. Each category (sometimes referred to as *class* or *concept*) is represented by a node, which is described by a set of linguistic labels (or *keywords*) that denote the "semantic meaning" of the category. In addition the directed edges between nodes may represent the relationships between categories. Ideally, the nodes near the top of the tree represent *general* concepts, while the deepest nodes (i.e., the leaves) represent *specific* concepts. Each object in the corpus belongs to one (in some applications, more than one) category that best describes the object. Figure 1 is a small example of taxonomy, where documents are found in each class.

Although building a domain specific concept hierarchy is in itself a challenging task, manual filtering and classifying documents within these taxonomies are significantly more expensive in terms of time and effort. Consequently there is considerable interest in developing automated tools to organize documents into hierarchies of concepts or taxonomies.

We address the problem of automatic classification of textual information (i.e., documents, web pages,

etc.) into a concept taxonomy defined by an editor to reflect the desired organization of the data. Sebastiani (Sebastiani, 2002) has written an excellent survey on machine learning methods for text categorization. Although there is considerable previous work on hierarchical classification of textual documents using many different approaches (e.g., *Neural Networks* (Ruiz & Srinivasan, 2002), *SVMs* (Dumais & Chen, 2000; Sun & Lim, 2001; Zhang & Lee, 2004; Hofmann et al., 2003), *Probabilistic* (Ceci & Malerba, 2003; Chakrabarti et al., 1997), *Association Rules* (Wang et al., 1999), etc.) most of it is directed towards the construction of document classifiers from labeled training data (i.e., supervised learning).

However, such an approach is often impractical because of the necessity to provide labeled training examples whenever a taxonomy is created or modified. The solution here is to automatically organize a given collection of unlabeled documents according to the new taxonomy specifications (nodes' keywords and the hierarchy) – a process known as *bootstrapping* (McCallum & Nigam, 1999). We can view *bootstrapping* as a preliminary step in a complex process to create and manage a text corpus. This initial step supports a human in labeling and classifying a set of unlabeled examples resulting in a labeled dataset that would eventually be used to train supervised classifiers to classify new documents as the corpus expands.

By the very nature of the problem the data is sparse, owing to the growth of the size of the taxonomy with the number of documents to enable easy management. That is, the number of classes grows with the amount of data in the corpus, resulting in a situation where the number of documents per class is never large enough for accurate parameter estimation. Therefore, simple approaches such as *Naive Bayes* or similarity based classification inevitably have poor accuracy. The effects of sparse data on parameter estimation and consequently on classification accuracy can be alleviated by *regularization* or *smoothing*, which are essentially variance reduction techniques.

This approach is fundamental to the *EM* with *shrinkage* algorithm proposed in (McCallum et al., 1998). They propose a regularization scheme based on a generative model for classifying documents on the leafs of a taxonomy. Each word in a document at the leaf is assumed to be generated by one of the nodes along the path from the root to the leaf. Under the generative model they derive an algorithm to obtain shrinkage estimators for class-conditional word probabilities. The shrinkage estimator derived from our model involves just one smoothing parameter as opposed to several parameters per leaf node in Nigam *et al.* Moreover, although their algorithm can be extendend to a situation where internal nodes of the hierarchy are also valid classes, their generative model is not applicable. We propose a simple generative model directly for the parameters (or, loosely, the reference vectors) of the nodes, which is applicable if the documents are to be classified also into internal nodes.

As an alternative solution to the small-sample problem in text classification, Huang *et al.* populate a topic hierarchy by automatically gathering information from the web by making search queries (Huang et al., 2004). Both the formulation of the query (using local keywords and those of the parent) and the subsequent computation of node reference vectors from the documents collected (the documents at a node comprise all the documents of its subtree) uses the structure of the hierarchy.

Bayesian methods have been extensively used in machine learning because they provide a theoretical framework to analyze and design decision systems as well as a way to incorporate prior knowledge about the problem. A Bayesian model is described by its parameters as well as prior distributions for the parameters. The Bayesian model is learnt by estimating its parameters from the observed data under the assumed prior. Hierarchical[2] Bayesian models are used to define and exploit inter-relationships between the parameters of a Bayesian model, by use of so-called *hyperpriors*, i.e., the prior distributions of the parameters of the priors of the model parameters. Hierarchical Bayesian models have been widely used in the absence of sufficient data because they afford a way to pool data from various sources to obtain robust estimates of the model parameters (Gelman et al., 2003).

We propose a *Hierarchical Dirichlet* generative model for the documents in the corpus and derive an unsupervised classification method based on the *EM* algorithm to classify a set of text documents into a given concept hierarchy defined only by the linguistic labels at the nodes and by the hierarchy of the nodes. Although Dirichlet priors have been extensively used for multinomial data, the novelty of our contribution lies in the specification of a model with dependent Dirichlet priors where the dependence is influenced by the structure of the concept taxonomy. The main contributions of this paper are the general probabilistic generative model for documents in a hierarchy, the algorithm for supervised and unsupervised learning of

[2]We use the word *hierarchical* in the sense of the tree of concepts as well as the statistical model. The context, however, should prevent any confusion.

its parameters, and the empirical evidence that our algorithm achieves a significant improvement in classification accuracy over a *Naive Bayes* and a simple *EM* based classification algorithms.

2. Hierarchical Dirichlet Model

Our generative model for a document in the corpus can be described as follows.

- A document d is a sequence of words drawn from the vocabulary $\mathcal{V}$ with $|\mathcal{V}| = k$.
- The probability of the document d given the class (or node) i in the hierarchy is given by

$$p(d|i) = p(|d|)\frac{(\sum_j d_j)!}{\prod_j d_j!}\prod_{j=1}^{k}(\theta_{ij})^{d_j} \qquad (1)$$

  where d_j is the number of occurrences of the j^{th} word of the vocabulary in document d. That is the parameter vector $\theta_i = (\theta_{i1}, \ldots, \theta_{ik})^T$ describes the parameters for a multinomial distribution (i.e., θ_{ij} is probabilities of the j^{th} word of the vocabulary for class i).

  We note that, contrary to most previous work where documents are classified only on the leaves of the concept hierarchy, we consider every node in the concept hierarchy as a valid class into which some of the documents belong.

- Furthermore the parameter vectors θ_i themselves have Dirichlet prior distributions[3].

$$\begin{aligned} \theta_i &\sim Dir(1, \ldots, 1) \quad \text{if } i \text{ is the root node} \\ \theta_i|\theta_{\mathrm{pa}(i)} &\sim Dir(\sigma\theta_{\mathrm{pa}(i)}) \quad \text{otherwise} \end{aligned}$$

  where pa(i) identifies the parent of node i and σ is a constant smoothing parameter chosen in advance.

Note that the prior for the parameter vector for a class is parameterized by the *true* parameter vector of its parent class, signifying that our proposed *Hierarchical Dirichlet* model has a hierarchical structure identical to the hierarchy (the tree) of the classes. Another way to view the model is to note that the parameters along a path from the root to a leaf are generated by a random walk mechanism where the parameter σ controls the step size. Furthermore, we use the uniform (i.e., non-informative) prior for the root node. A large value for σ indicates that the parameter vector for a node is *tightly* bound to that of its parent. We also note that for most generative models for document hierarchies the document length is assumed to be generated by a class-independent Poisson distribution. We do not incorporate it into our model because it does not play a role during classification.

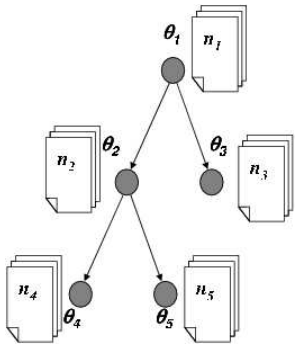

Figure 2. Hierarchical organization of documents.

The intuitive justification for our model is that the concept at a node, being more general, subsumes the concepts of its children. This is implicitly encoded in the model because we have

$$E[\theta_i|\theta_{\mathrm{pa}(i)}] = \theta_{\mathrm{pa}(i)} \qquad (2)$$

2.1. Parameter Estimation

We now describe how the parameters of the model can be learnt from data. First let us consider the case where the data consists of labeled documents for every class. Let $n_i = (n_{i1}, \ldots, n_{ik})^T$ be the vector that describes the number of occurrences of each word of the vocabulary in *all* the documents at class i and let $N_i = \sum_j n_{ij}$. We could say that n_i is a sort of meta-document built merging all the documents in the node i (see Figure 2). We recall that $n_i|\theta_i \sim multinomial(\theta_i)$.

The estimate for the parameter vector θ_i for a node i is updated iteratively given the data at the node and the previous estimates of the parameters at its neighbors. We will now derive the update formulae for the iterative algorithm.

At the leaves of the hierarchy we perform a Bayes Minimum Mean Squared Error Estimate (MMSE) (which is the a posteriori expectation of the random param-

[3]We recall that Dirichlet is the conjugate prior of multinomial distribution. For $X \sim Dir(v)$ i.e., $p(X = (x_1, \ldots, x_n)) = \frac{\prod_i^n \Gamma(v_i)}{\Gamma\left(\sum_i^n v_i\right)} \prod_i^n x_i^{v_i - 1}$, the mean and covariance matrix are given by $E[X] = \frac{1}{\sum_i^n v_i} v$ and $\Sigma(X) = \frac{1}{1+\sum_i^n v_i}(-E[X]E[X]^T + diag(E[X]))$

eter vector) given the data under the Dirichlet prior parameterized by the parameter vector of the parent. This is straightforward since the Dirichlet distribution is a conjugate prior for the parameters of a multinomial distribution. Hence the estimate for the parameters of a leaf node i are:

$$\begin{aligned}\hat{\theta}_i &= \frac{1}{\sum_j n_{ij} + \sigma \sum_j \hat{\theta}_{\mathrm{pa}(i)j}}(n_i + \sigma\hat{\theta}_{\mathrm{pa}(i)}) \\ &= \frac{1}{N_i + \sigma}(n_i + \sigma\hat{\theta}_{\mathrm{pa}(i)}) \qquad (3)\end{aligned}$$

For an internal node its parameter vector must be estimated given the data at the node, the parameter estimate at its parent and the estimates of the parameters at its children. We can view this step as estimating the parameter vector at a node considering the parameters of its children also as part of the data. An MMSE estimate, however, is intractable because the parameters of the children are generated by a Dirichlet distribution and the posterior distribution is no longer Dirichlet. We therefore perform this estimation using a Linear Minimum Mean Squared Error Estimate (LMMSE). The LMMSE is the linear estimate with the lowest mean squared error and depends only on the first and second moments of the joint parameter and data distribution. We do not use a more natural Maximum A Posteriori (MAP) estimate because LMMSE leads to a very simple algorithm where the estimator update equations are the widely used linearly regularized estimates.

For simplicity, instead of a general derivation, we show how the estimate of the parameter vector for node 2 in Figure 2 is updated using LMMSE.

Assume that we have the estimates $\hat{\theta}_1$, $\hat{\theta}_4$ and $\hat{\theta}_5$ from the previous iteration for the parameter vectors for the nodes 1, 4 and 5. We have a prior coming from the parent of node 2 (i.e., node 1) parameterized by $\hat{\theta}_1$. The problem is to update the estimate of θ_2 given $\hat{\theta}_4$, $\hat{\theta}_5$, the data n_2 and the prior $(Dir(\sigma\hat{\theta}_1))$.

Firstly the posterior distribution of θ_2 given the data at node 2 is given by

$$p(\theta_2|n_2) = Dir(\sigma\hat{\theta}_1 + n_2)$$

Now using this as the distribution for θ_2 we estimate (using LMMSE) $\hat{\theta}_2$ from $\hat{\theta}_4$ and $\hat{\theta}_5$.

The estimate $\hat{\theta}_2$ can be computed by an LMMSE estimate as follows (Moon & Stirling, 2000):

$$\begin{aligned}\hat{\theta}_2 &= E[\theta_2] + [\Sigma(\theta_2,\theta_4)\Sigma(\theta_2,\theta_5)] \cdot \\ &\quad \begin{bmatrix} \Sigma(\theta_4) & \Sigma(\theta_4,\theta_5) \\ \Sigma(\theta_5,\theta_4) & \Sigma(\theta_5) \end{bmatrix}^{-1} \begin{bmatrix} \hat{\theta}_4 - E[\theta_4] \\ \hat{\theta}_5 - E[\theta_5] \end{bmatrix} \qquad (4)\end{aligned}$$

Let us look at each of the terms in the above equation.

$$E[\theta_2] = \frac{1}{\sigma + N_2}(\sigma\hat{\theta}_1 + n_2) \qquad (5)$$

$$E[\theta_4] = E_{\theta_2}[E[\theta_4|\theta_2]] = E[\theta_2] \qquad (6)$$

$$\begin{aligned}\Sigma(\theta_2,\theta_5) &= \Sigma(\theta_2,\theta_4) \\ &= E_{\theta_2}[(\theta_2 - E[\theta_2])E[(\theta_4 - E[\theta_4])^T|\theta_2]] \\ &= E_{\theta_2}[(\theta_2 - E[\theta_2])(\theta_2 - E[\theta_2])^T] \\ &= \Sigma(\theta_2) \qquad (7)\end{aligned}$$

$$\begin{aligned}\Sigma(\theta_5,\theta_4) &= \Sigma(\theta_4,\theta_5) \\ &= E_{\theta_2}[E_{\theta_4,\theta_5}[(\theta_4 - E[\theta_4])(\theta_5 - E[\theta_5])^T|\theta_2]] \\ &= E_{\theta_2}[E[(\theta_4 - E[\theta_4])|\theta_2]E[(\theta_5 - E[\theta_5])^T|\theta_2]] \\ &= \Sigma(\theta_2) \qquad (8)\end{aligned}$$

$$\begin{aligned}\Sigma(\theta_4) &= \Sigma(\theta_5) = \Sigma_{\theta_2}(E[\theta_4|\theta_2]) + E_{\theta_2}[\Sigma(\theta_4|\theta_2)] \\ &= \Sigma(\theta_2) + E_{\theta_2}[\Sigma(\theta_4|\theta_2)] \qquad (9)\end{aligned}$$

The above derivations use Equation 2, the properties of conditional expectation, the law of total variance and the fact that the random variables θ_4 and θ_5 are exchangeable.

The only two terms left to compute are $\Sigma(\theta_2)$ and $E_{\theta_2}[\Sigma(\theta_4|\theta_2)]$ which are given by

$$\Sigma(\theta_2) = \frac{1}{\sigma + N_2 + 1}(-E[\theta_2]E[\theta_2]^T + diag(E[\theta_2])) \qquad (10)$$

$$\begin{aligned}E_{\theta_2}[\Sigma(\theta_4|\theta_2)] &= \frac{1}{\sigma+1}E_{\theta_2}[-(\theta_2\theta_2^T) + diag(\theta_2)] \\ &= \frac{1}{\sigma+1}(-\Sigma(\theta_2) - E[\theta_2]E[\theta_2]^T + \\ &\qquad + diag(E[\theta_2])) \qquad (11)\end{aligned}$$

The LMMSE estimation step (Eq. 4) seems computationally expensive due to the $mk \times mk$ matrix inversion (where m is the number of children of the node whose parameters are being estimated). However owing to the fact that the parameter vectors for the children of a node are exchangeable, it can be shown that the

directory: cooking/soups_and_stews/fish_and_seafood
pre-processed node's keywords: fish, seafood

url: http://www.fish2go.com/rec_0120.htm
site's description: Finnan Haddie and Watercress Soup: made with smoked haddock, potatoes, watercress, and milk.
pre-processed description: smoke, watercress, make, haddock, milk, potato, soup.

url: http://www.bettycrocker.com/default.asp
site's description: Crunchy Snacks from Betty Crocker: collection of sweet and savory snack recipes which pack a crunch, from healthy vegetables to s'mores.
pre-processed description: snack, collection, recipe, healthy, savoury, vegetable, sweet

Figure 3. Example of two "documents" in a node of Google. Pre-processed node's keywords and description's words were used to generate the vocabulary by which descriptions were then encoded.

general update equation for node i with m children simplifies to

$$\begin{aligned}\hat{\theta}_i &= E[\theta_i] + \\ &+ m \cdot (E_{\theta_i}[\Sigma(\theta_{\mathrm{ch}(i)}|\theta_i)]\Sigma(\theta_i)^{-1} + mI)^{-1} \cdot \\ &\cdot (\bar{\hat{\theta}}_c - E[\theta_i]) \end{aligned} \quad (12)$$

where $\bar{\hat{\theta}}_{\mathrm{ch(i)}} = \frac{1}{m}\sum_{j\in \mathrm{ch}(i)} \hat{\theta}_j$, $\mathrm{ch}(i)$ represents the children of node i and I is the identity matrix.

Moreover, due to the structure of the matrices $E_{\theta_i}[\Sigma(\theta_{\mathrm{ch}(i)}|\theta_i]$ and $\Sigma(\theta_i)$ (cf. Equations 10 and 11), Equation 12 reduces to

$$\hat{\theta}_i = \frac{\sigma\hat{\theta}_{\mathrm{pa}(i)} + m(\sigma+1)\bar{\hat{\theta}}_{\mathrm{ch(i)}} + n_i}{\sigma + m(\sigma+1) + N_i} \quad (13)$$

Thus starting from the initial guesses for the parameters, the estimates are updated (Equation 13) iteratively until some convergence criterion is met. We note that for the root node we have the uniform prior instead of the prior coming from the parent.

From Equation 13 we note that the estimate of the parameter vector of particular node is just a weighted average of the maximum-likelihood estimate from the data and the corresponding estimates from the parent and children of the node. We also note from the above equation, that under our model the estimates of the parameters are very similar to the James-Stein shrinkage estimators (Efron & Morris, 1977).

2.2. Document Clustering

The above equations for parameter estimation require class-labeled documents, therefore they can only be used for a supervised classification task. However, our goal is to classify a set of unlabeled documents into the classes which are labeled only by a small set of keywords. We propose to address this problem by clustering the documents using the well-known *EM* algorithm for multinomial mixtures. The *EM* algorithm is initialized by using the keywords at each node. Actually, the prototype vectors made from the node labels are used as the initial seeds:

$$\theta_{ij} = \begin{cases} 0.9 & \text{if word } j \text{ occurs as a keyword for node } i \\ 0.1 & \text{otherwise} \end{cases} \quad (14)$$

At each iteration of the *EM* algorithm we compute membership of every document to each class using the parameter estimates from the previous iteration (the E-step). We then update the parameters with Equation 12 using the weighted set of documents at each node (the M-step).

3. Experimental Evaluation

3.1. Datasets

We evaluated the approach on eight benchmark datasets (taxonomies) extracted from two well known Web directories – Google and LookSmart. These eight datasets correspond to different sub-directories (concept hierarchies) selected to represent a variety in the dimensionality. All the linguistic descriptions, i.e., keywords for nodes, and documents' content (short descriptions of the linked sites) are made by the editors of the directories (see an example in Figure 3).

The data were preprocessed by both removing stopwords and stemming the words to common roots. The feature space (i.e., the vocabulary) was separately determined for each taxonomy by a process of feature selection by eliminating words that were extremely frequent or very rare. We used this approach because the task we address is unsupervised learning and therefore a sophisticated feature selection method that requires class labels cannot be used. Furthermore the keywords occurring at all the nodes were also added to the vocabulary. Each dataset was further processed to remove all documents having less than four words and all subtrees having their root node without documents. The statistics of the datasets used for the experiments are presented in Table 1.

Since our algorithm performs clustering as a means for classification of the documents, there was no need to partition the data into training and test sets.

Evidently, the classification task is quite hard, since, after preprocessing, the document summaries are made up of only a few words (see Table 1). Due to the scarcity of repeating words in the preprocessed sum-

	Hierarchies				Documents	
	Max depth	Nodes	Docs	Average lab/node	Vocab. size	Average w/doc
Google						
Archaeology	5	122	1204	1.91	649	10.92
Language	8	514	4138	1.40	866	8.94
Neuro. Disord.	4	210	2443	1.80	632	9.68
News Media	5	29	549	1.38	586	10.69
LookSmart						
Archaeology	5	51	609	1.53	673	8.58
Common Lang.	4	139	1794	1.45	529	8.75
Movies	4	34	623	2.21	524	9.39
Peripherals	5	175	3925	1.73	538	11.33

Table 1. Statistics of the eight concept hierarchies used to evaluate the model. The columns present respectively the maximum depth of the hierarchy, the number of classes, total number of documents, the average number of keywords per node, the number of words in the vocabulary (after feature selection) and the average number of words per document.

maries, documents were encoded as *set-of-words*, i.e., binary vectors indicating the presence or absence of the corresponding word in the document.

3.2. *Set-of-words* Model

In the *set-of-words* framework, given a vocabulary of length k, the bits in a document vector at a certain node are assumed to be generated independently by k binomial distributions with Beta priors. The Beta distribution is just a one-dimensional Dirichlet and therefore the update equation (from Equation 13) for the j^{th} parameter at node i (the probability θ_{ij} of appearance of word j in a document at node i) is

$$\hat{\theta}_{ij} = \frac{\sigma\hat{\theta}_{\mathrm{pa}(i)j} + m(\sigma+1)\bar{\hat{\theta}}_{\mathrm{ch}(i)j} + n_{ij}}{\sigma + m(\sigma+1) + N_i} \qquad (15)$$

where m is the number of children of node i, N_i is the number of documents at node i, n_{ij} is the number of documents at node i with word j present, $\hat{\theta}_{\mathrm{pa}(i)j}$ is the estimate of the parameter for the j^{th} word for the parent of node i and $\bar{\hat{\theta}}_{\mathrm{ch}(i)j}$ is the mean of the estimates of the parameter for j^{th} word for all the children of i.

The above equation is for the estimation of the classification parameters from a labeled data set. On the contrary, for clustering documents, the corresponding *EM* update formula is given by

$$\hat{\theta}_{ij} = \frac{\sigma\hat{\theta}_{\mathrm{pa}(i)j} + m(\sigma+1)\bar{\hat{\theta}}_{\mathrm{ch}(i)j} + \sum_d d_j p(i|d)}{\sigma + m(\sigma+1) + \sum_d p(i|d)} \qquad (16)$$

where

$$p(i|d) \propto p(i) \cdot \prod_{j=1}^{k} (\theta_{ij})^{d_j} \cdot (1-\theta_{ij})^{(1-d_j)} \qquad (17)$$

3.3. Choice of σ

When class labeled examples are available, the value of the smoothing parameter σ can be optimized by cross-validation. However for the unsupervised classification of documents the choice of σ is more complicated. In such a situation we may require a small training set that can be used to optimize σ.

At each iteration of our *EM*-based clustering algorithm we tried to estimate σ using method-of-moments. The estimates for σ we obtained were extremely large owing to the fact that for most neighboring class pairs many words never occur and the corresponding word probabilities are very close to zero for both classes. Another approach to estimating σ may be along the line used in (McCallum et al., 1998). They optimize their smoothing parameters by choosing a value that maximizes the likelihood of some held-out (unlabeled) data. We intend to pursue this approach in the future. However, we provide evidence below that smoothing even with an incorrect value of σ is often better than no smoothing at all.

3.4. Discussion

We used a maximum a posteriori classifier that classified each document into the class with the highest posterior probability. We reject a document when there was more than one class with the highest probability. The efficacy of the proposed algorithms was evaluated by comparing the classification result of each document with its original label. To evaluate the model we adopted the standard information retrieval measure *micro-F*1 (Baeza-Yates & Ribeiro-Neto, 1999), that combines *precision* and *recall* of a model on a given dataset. At zero percent reject rate the *micro-F*1 value equals the classification accuracy. For each experiment our *EM*-based classification algorithm that performs estimation and classification iteratively was terminated when the number of documents classified differently from the previous iteration is less then 1% or after 10 iterations, which ever occurs first.

The results of the proposed model were then compared to those of a standard *EM* algorithm without any regularization and a standard *Naive Bayes* classifier. The word probabilities used by the standard *Naive Bayes* classifier are obtained from the keywords at the nodes as described by Equation 14. These probability vec-

	NB	EM	HD best (σ)	HD $\sigma = 2$
Google				
Archaeology	25.63	0.42	27.74 (1)	25.83
Language	24.67	30.69	27.31 (1)	26.21
Neuro. Disord.	37.78	0.61	42.24 (1)	41.55
News Media	31.66	37.86	41.71 (5)	39.53
LookSmart				
Archaeology	18.54	5.75	34.48 (10)	29.56
Common Lang.	11.25	1.06	29.93 (7)	28.09
Movies	26.83	33.71	42.70 (5)	41.09
Peripherals	10.96	0.05	20.43 (7)	19.13

Table 2. $F1$ measures for all models (*Naive Bayes*, *EM*, and *Hierarchical Dirichlet*) on the eight benchmark taxonomies. The *Hierarchical Dirichlet* results are presented both for the best smoothing parameter σ, and for a fixed σ for all datasets.

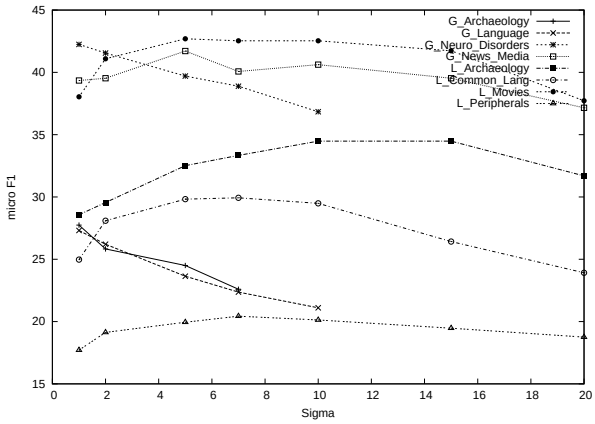

Figure 4. $F1$ results of *Hierarchical Dirichlet* model for various smoothing parameters σ on all eight taxonomies.

tors were also used to initialize both the standard *EM* and our regularized *EM* algorithms.

In Table 2 we present the classification results of the different approaches on the eight datasets. We observe that the standard *EM* algorithm performs poorly on several datasets. This can be attributed to the fact that without regularization the algorithm often results in an incorrect identification of cluster labels to class labels. Our smoothed variant of the *EM* algorithm performs significantly better than both the standard *EM* and the *Naive Bayes* classifier on almost all datasets. In Table 2 we present the classification results for the best value of σ on each dataset as well as the results with a fixed $\sigma = 2$. We observe that even when the chosen σ is not the best for the dataset, smoothed estimates of the classification parameters result in considerably better accuracy than without smoothing.

Figure 4 shows the plot of $F1$ values against the smoothing parameter (σ) for all the eight benchmark taxonomies. We observe that for the experimental datasets the classification accuracy does not degrade drastically with an incorrect choice of σ. In fact for many of the taxonomies the classification accuracy stays the same over a wide range of σ values.

4. Conclusion and Future Work

We presented a simple generative model for the generation of documents in a concept hierarchy. The model encodes our intuition about the relationships between neighboring nodes in the hierarchy by means of hierarchically dependent Dirichlet priors for the class parameter vectors. Under this model we derived formulae to estimate the parameters in a supervised as well as an unsupervised setting. We have shown that under our model the parameter estimates closely resemble the shrinkage estimates used in statistics.

The major drawback of our approach is the a priori choice of smoothing parameter σ. We intend to design methods to automatically choose the σ from data. We plan to find improved methods for the initialization of the clustering algorithm, since it is critical for the classifiation accuracy. One way to improve the initialization is to use the method proposed in (Huang et al., 2004). We also plan on experimenting with datasets with more repeating words per document in order to test the more general Dirichlet model and to evaluate the efficacy of the model on semi-supervised classification tasks.

5. Other Related Work

Blei *et al.* propose *latent Dirichlet allocation* as a very general generative hierarchical model for a collection of text documents where every document is generated by assuming that for each word a topic is chosen according to a Dirichlet distribution and the word by a multinomial distribution given the topic and repeating the process. The parameters are estimated by variational methods (Blei et al., 2003). In contrast to our approach, they do not model the relationships between classes.

Vaithyanathan *et al.* propose a Bayesian approach for supervised text classification by using an integrated probability of a document given a class instead of using the point estimates of the probability vectors. They also propose ways to learn the parameters for the priors from the training data (Vaithyanathan et al., 2000).

Taskar *et al.* propose *probabilistic relational models* as a means to exploit relationships between data points to improve the accuracy of the models (Taskar et al.,

2001). Our approach in contrast exploits the prior knowledge about the relationships between class labels.

Hooper constructed a family of dependent Dirichlet priors and derived optimal linear estimators for the rows of the conditional probability matrix at the nodes of a Bayesian network (Hooper, 2004). The dependence structure between the Dirichlet random vectors is very different from the one we proposed.

References

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. Addison Wesley.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, *3*, 993–1022.

Ceci, M., & Malerba, D. (2003). Hierarchical classification of html documents with WebClassII. *Proc. of the 25th European Conf. on Information Retrieval (ECIR'03)* (pp. 57–72).

Chakrabarti, S., Dom, B., Agrawal, R., & Raghavan, P. (1997). Using taxonomy, discriminants, and signatures for navigating in text databases. *VLDB'97, Proc. of 23rd Int. Conf. on Very Large Data Bases* (pp. 446–455). Morgan Kaufmann.

Dumais, S., & Chen, H. (2000). Hierarchical classification of web document. *Proc. of the 23rd ACM Int. Conf. on Research and Development in Information Retrieval (SIGIR'00)*.

Efron, B., & Morris, C. (1977). Stein's paradox in statistics. *Scientific American*, *236*, 119–127.

Gelman, A., Carlin, J. B., Stern, H. S., & Rubin, D. B. (2003). *Bayesian data analysis, second edition*. Chapman & Hall/CRC.

Hofmann, T., Cai, L., & Ciaramita, M. (2003). Learning with taxonomies: Classifying documents and words. *Workshop on Syntax, Semantics, and Statistics, at NIPS'03*.

Hooper, P. (2004). Dependent dirichlet priors and optimal linear estimators for belief net parameters. *Proceedings of the 20th Annual Conference on Uncertainty in Artificial Intelligence (UAI-04)* (pp. 251–259). Arlington, Virginia: AUAI Press.

Huang, C.-C., Chuang, S.-L., & Chien, L.-F. (2004). Liveclassifier: creating hierarchical text classifiers through web corpora. *WWW '04: Proc. of the 13th Int. Conf. on World Wide Web* (pp. 184–192). New York, NY, USA: ACM Press.

McCallum, A., & Nigam, K. (1999). Text classification by bootstrapping with keywords. *ACL99 - Workshop for Unsupervised Learning in Natural Language Processing*.

McCallum, A., Rosenfeld, R., Mitchel, T., & Ng, A. (1998). Improving text classification by shrinkage in a hierarchy of classes. *ICML98 - Proc. of 15th Int. Conf. on Machine Learning* (pp. 358–367).

Moon, T. K., & Stirling, W. C. (2000). *Mathematical methods and algorithms for signal processing*. Englewood Cliffs, New Jersey: Prentice-Hall.

Ruiz, M., & Srinivasan, P. (2002). Hierarchical text categorization using neural networks. *Information Retrieval*, *5*, 87–118.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, *34*, 1–47.

Sun, A., & Lim, E. (2001). Hierarchical text classification and evaluation. *ICDM 2001 - Proc. of the 2001 IEEE Int. Conf. on Data Mining* (pp. 521–528). IEEE Computer Society.

Taskar, B., Segal, E., & Koller, D. (2001). Probabilistic classification and clustering in relational data. *Proceeding of IJCAI-01, 17th International Joint Conference on Artificial Intelligence* (pp. 870–878). Seattle, US.

Vaithyanathan, S., Mao, J.-C., & Dom, B. (2000). Hierarchical bayes for text classification. *PRICAI Workshop on Text and Web Mining* (pp. 36–43).

Wang, K., Zhou, S., & Liew, S. (1999). Building hierarchical classifiers using class proximity. *Proc. of the 25th VLDB Conference*.

Zhang, D., & Lee, W. S. (2004). Web taxonomy integration using support vector machines. *WWW '04: Proc. of Int. Conf. on World Wide Web* (pp. 472–481). ACM Press.