Using Belief Networks and Fisher Kernels for structured document classification

Ludovic Denoyer[1], Patrick Gallinari[1]

Laboratoire d'Informatique de Paris VI
LIP6
France
{ludovic.denoyer, patrick.gallinari}@lip6.fr

Abstract. We consider the classification of structured (e.g. XML) textual documents. We first propose a generative model based on Belief Networks which allows us to simultaneously take into account structure and content information. We then show how this model can be extended into a more efficient classifier using the Fisher kernel method. In both cases model parameters are learned from a labelled training set of representative documents. We present experiments on two collections of structured documents : WebKB which has become a reference corpus for HTML page classification and the new INEX corpus which has been developed for the evaluation of XML information retrieval systems.
Keywords : textual document classification, structured document, XML corpus, Belief Networks, Fisher Kernel

1 Introduction

The development of large electronic document collections and Web resources has been paralleled by the emergence of structured document format proposals. They are aimed at encoding content information in a suitable form, for a variety of information needs. These document formats allow us to enrich the document content with additional information (document logical structure, meta-data, comments, etc) and to store and access the documents in a more efficient way. Some proposals have already gained some popularity and description languages like XML are already widely used by different communities. For text documents, these representations encode both structural and content information.

With the development of structured collections, there is a need to develop information access methods which may take all the benefit of these richer representations and also allow to answer new information access challenges and new user needs. Current Information Retrieval (IR) methods have mainly been developed for handling flat document representations and cannot be easily adapted to deal with structured representations.

In this paper, we focus on the particular task of structured document categorization. We propose methods for exploiting both the content and the structure information for this task. Our core model is a generative categorization model

based on belief networks (BN). This work offers a natural framework for encoding structured representations and allows us to perform inference both on whole documents and on document subparts. We then show how to turn this generative model into a discriminant classification model using the Fisher kernel trick. Paper is organized as follows : we make in 2 a brief review of previous work on structured document classification, we describe in 3 the type of structured document we are working on, we then introduce in 4 our generative model and the discriminant model in 5. Section 6 presents a series of experiments performed on two textual collections, the WebKB [20] and the INEX Corpus [7].

2 Previous works

Text categorization is a classical information retrieval task which has motivated a large amount of work over the last fews years. Most categorization models have been designed for handling bag of words representations and do not consider word ordering or document structure. Generally speaking, classifiers fall into two categories: generative models which estimate class conditional densities $P(document/Class)$ and discriminant models which directly estimate the posterior probabilities $P(Class/document)$. The naive Bayes model [12] for example is a popular generative categorization model whereas among discriminative techniques support vector machines [10] have been widely used over the last few years. [17] makes a complete review of flat document categorization methods. More recently, models which take into account sequence information have been proposed [3]. Classifying structured document is a new challenge both from IR and machine learning perspectives. For the former, flat text classifiers do not lead to natural extensions for structured documents, however there has been recently some interest in the classification of HTML pages. For the latter, the classification of structured data is an open problem since most classifiers have been designed for vector or sequence representations, and only a few formal frameworks allow to consider simultaneously content and structure informations. We briefly review below recent work in these different areas.

The expansion of the Web has motivated a series of works on Web page categorization - viz. the last two Trec competitions [19]. Web pages are built from different type of information (title, links, text, etc) which play different roles. There has been several attempts to combine these information sources in order to increase page categorization scores ([5],[21]). Chakrabarti ([2]) proposes to use the information contained in neightboring documents of an HTML pages. All these approaches which deal only with HTML, propose simple schemes either for encoding the page structure or for exploiting the different types of information by combining basic classifiers. These models exploit a priori knowledge about the particular semantics of HTML tags, and as such cannot be extended to more complex languages like XML where tags may be defined by the user. We will see that our model does not exploit this type of semantics and is able to learn from data the importance of tag information.

Some authors have proposed more principled approaches to deal with the general problem of structured document categorization. These models are not specific to HTML even when they are tested on HTML databases due to the lack of a reference XML corpus. [4] for example propose the Hidden Tree Markov Model (HTMM) which is an extension of HMMs to a structured representation. They consider tree structured documents where in each node (structural element), terms are generated by a node specific HMM. [16] have proposed a Bayesian network for classifying structured documents. This is a discriminative model which directly computes the posterior probability corresponding to the document relevance for each class. [22] present an extension of the Naive Bayes model to semi-structured documents where essentially global word frequencies estimators are replaced with local estimators computed for each path element. [18] propose to use Probabilistic Relationnal Models to classify structured document and more precisely Web pages.

For the ad-hoc IR task, Bayesian networks (BN) have been used for information retrieval for some time. Inquery [1] retrieval engine operates on flat text while more recent proposal handle structured documents,e.g. [14], [15]. Outside the field of information retrieval, some models have been proposed to handle structured data. The hierarchical HMM (HHMM) [6] is a generalization of HMMs to structured data, it has been tested on handwriting recognition and on the analysis of English sentences, similar HMM extensions have been used for multi-agent modeling [13]. However, inference and learning algorithms in these models are too computationally demanding for handling large IR tasks. The inference complexity for HHMM is $O(NT^3)$ where N is the number of states in their HMM and T the length of the text in words, for comparison our model is more like $O(N+T)$ as will be seen later.

The core model we propose is a generative model which has been developed for the categorization of any tree like document structure (typically XML documents). This model bears some similarities with the one in [4], however, their model is adapted to the semantic of HTML documents and considers only the inclusion relation between two document parts. Ours is generic and can be used for any type of structured document. Even when tags do not convey semantic information, it allows considering different types of relations between structured elements: inclusion, depth in the hierarchical document, etc. This model could be considered as a special case of the HHMM [6] since it is simpler and since HHMM can be represented as particular BNs [13]. It is computationally much less demanding and has been designed for handling large document collections. This generative model is then extended into a discriminant one using the method of the Fisher Kernel. For that, we extend to the case of structured data the ideas initially proposed by [9] for sequences.

Our main contributions are a new generative model for the categorization of large collections of structured documents and its extension via the use of Fisher kernels into a discriminant model. We also describe for the first time to our knowledge experiments on a large corpus of structured XML documents (INEX) developed in 2002 for ad-hoc retrieval tasks.

3 Document structure

We represent a structured document d as a Directed Acyclic Graph (DAG). Each node of the graph represents a structural entity of the document, and each edge represents a hierarchical relation between two entities (for example, a paragraph is included in a section, two paragraphs are on the same level of the hierarchy, etc). For keeping inference complexity to a reasonable level, we do not consider circular relations which might appear in some documents (e.g. Web sites), this restriction is not too severe since this definition already encompasses many different types of structured documents.
Each node of the DAG is composed of **a label** (for example, labels can be *section*, *paragraph*, *title* and represent the structural semantic of a document) and **a textual information** (which is the textual content associated to this node if any).

A structured document then contains three types of information: **the logical structure information** represented by the edges of the DAG (the position of the tag in an XML document), **the label information** (the name of the tag in an XML document) and **the textual information**. Figure 1 gives a simple example of structured document.

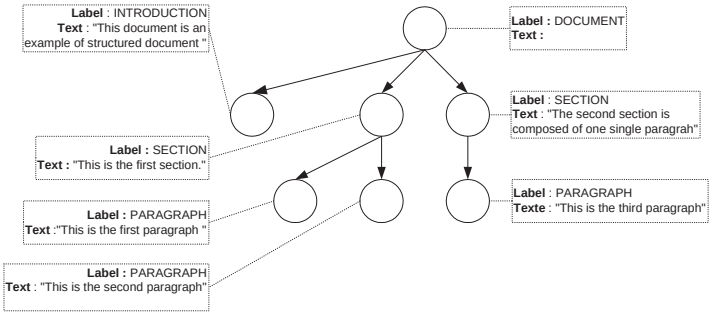

Fig. 1. An example of structured document represented as a Direct Acyclic Graph. This document is composed of an introduction and two sections. Each part of the document is represented by a node with a **label** and a **textual information**

4 A generative model for structured documents

We now present a generative model for structured documents. It is based on BNs and allows to handle these 3 types of information. This model can be used with any XML document without using *a priori* informations about the semantic of the structure. We first briefly introduce BNs and then describe the different elements of the model.

4.1 Notations

We will use the following notations, let :

- d : be a structured document
- s_d : be the structure of document d. $s_d = (\{s_d^i\}, pa(s_d^i))$ where $\{s_d^i\}_{i \in [1..|s_d|]}$ is the set of node labels ($|s_d|$ is the number of structured nodes for d), $s_d^i \in \Lambda$ with Λ the set of possible labels. $pa(s_d^i)$ are the parents of node $\{s_d^i\}$ in the structured document and describe the **logical structure information**.
- t_d : be the textual information in d. t_d is a set $\{t_d^i\}_{i \in [1..|s_d|]}$ of textual elements for each node i of the structured document.
- $\{w_{d,k}^i\}_{k \in [1..|t_d^i|]}$ is the set of words of part t_d^i in document d ($|t_d^i|$ is the number of words of part t_d^i) and $w_{d,k}^i$ is the kth word in t_d^i. $w_{d,k}^i \in V$ where V is the set of indexing terms in the corpus.

4.2 Base model using belief networks

Belief networks [11] are stochastic models for computing the joint probability distribution over a set of random variables. A BN is a DAG whose nodes are the random variables and whose edges correspond to probabilistic dependence relations between 2 such variables. More precisely, the DAG reflects conditional independence properties between variables, the joint probability of a set of variables writes :

$$P(x_1, ..., x_n) = \prod_{i=1..n} P(x_i / pa(x_i)) \text{ where } pa(x_i) \text{ denotes the parents of } x_i.$$

4.3 Generative model components

We consider a structured document as **the realization of random variables** T and S corresponding respectively to textual and structural information. For simplicity, we will denote $P(T = t_d, S = s_d/\theta)$ as $P(t_d, s_d/\theta)$. Let θ denotes the parameters of our document model, the probability of a document writes :

$$P(d|\theta) = P((t_d, s_d)|\theta) = P(s_d|\theta)P(t_d|s_d, \theta) \quad (1)$$

$P(s_d|\theta)$ is the **structural probability** of d and $P(t_d/\theta)$ is the **textual probability** of d given its structure s. Each document will be modeled via a BN, whose nodes correspond either to tag or textual information and whose directed edges encode the relations between the document elements. The whole corpus will then be represented as a series of BN models, 1 per document. The BN model of a document can be thought of as a model of the structured document generation, where the generation process goes as follows : someone who wants to create a document about a specific topic will sequentially and recursively create the document organization and then fill the corresponding nodes with text. For example he first creates sections after what, for each section, he creates subsections etc... recursively. At the end, in each "terminal" node, he will create

the textual information of this part as a succession of words. This is a typical generative approach which extends to structured information the classical HMM approach for modeling sequences. The two components -structure and content- of the model are detailed below.

Structural probability
The structural information of a document is encoded into the edges of the BN. Under the conditional independence assumption of the BN document model, the structural density of document d writes:

$$P(s_d|\theta) = \prod_{i=1}^{|s_d|} P(s_d^i|pa(s_d^i)) \quad (2)$$

The BN structural parameters are then the $\{P(s_d^i|pa(s_d^i))\}$ which are the probabilities to observe s_d^i given its parents $pa(s_d^i)$ in the BN.

In order to have a robust estimation of the BN parameters, we will share sets of parameters among all the document models. We will make the hypothesis that the $\{P(s_d^i|pa(s_d^i))\}$ only depend on the labels of nodes s_d^i and $pa(s_d^i)$, i.e. two nodes in two different document models which share the same label and whose parents also share the same labels will have the same conditional probability $P(s_d^i|pa(s_d^i))$.

Within this framework, several BN models may be associated to a document d. Figure 2 illustrates two of the models we have been working with. The DAG structure of Model 2 is copied from the tree structure of the document and reflects only the inclusion relation. The same type of relation is used in [4]. Model 1 contains both inclusion information (vertical edges) and sequence information (horizontal edges). Both models are an overly simplified representation of the real dependencies between document parts. This allows to keep the complexity of learning and inference algorithms low. Statistical models that work best are often very simple compared to the underlying phenomenon (e.g. naive Bayes in text classification or Hidden Markov Models in speech recognition), practioners of BNs have experienced the same phenomenon. Note that other instances of our generic model could have also been used here.

Textual probability
For modeling the textual content of a structured document, we make the following hypothesis :

- the probability of a word only depends on the label of the node that contains this word (first order dependency assumption).
- in a node, words are independent (Naive Bayes assumption)

The naive Bayes hypothesis is not mandatory here and any other term generative model (e.g. HMM) could be used instead, however this hypothesis allows for a robust density estimation and in our experiments more sophisticated models did not led to any performance improvement.

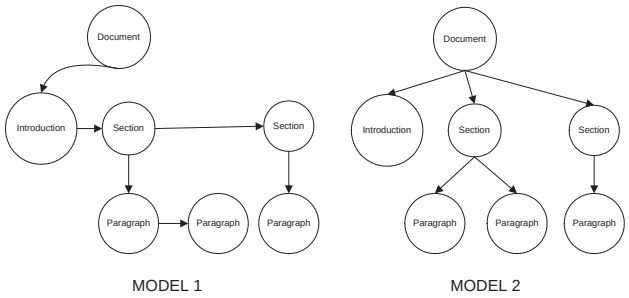

Fig. 2. Two possible structural belief networks constructed for the document presented in figure 1.

For a particular part t_d^i of document d, we then have :

$$P(t_d^i/s_d, \theta) = P(t_d^i/s_d^i, \theta) = \prod_{k=1}^{|t_d^i|} P(w_{d,k}^i|s_d^i, \theta) \tag{3}$$

And for the entire document, we have :

$$P(t_d|s_d, \theta) = \prod_{i=1}^{i=|s_d|} \prod_{k=1}^{|t_d^i|} P(w_{d,k}^i|s_d^i, \theta) \tag{4}$$

Final belief network

Combining equations 2 and 4, we get a generative structured document model

$$P(d|\theta) = \prod_{i=1}^{|s_d|} \left(P(s_d^i|pa(s_d^i), \theta) \prod_{k=1}^{|t_d^i|} P(w_{d,k}^i|s_d^i, \theta) \right) \tag{5}$$

Equation 5 describes the contribution of structural and textual information in our model.

4.4 Learning

This model is completely defined by two sets of parameters, *transition* and *emission* probabilities respectively denoted by $P(s_i|s_j)$ and $P(w_i|s_j)$:

$$\theta = \{P(s_i|s_j)\}_{s_i,s_j \in \Lambda} \bigcup \{P(w_i|s_j)\}_{w_i \in V, s_j \in \Lambda}$$

In order to learn the θ, we use the Expectation Maximization (EM) algorithm for optimizing the maximum likehood of the data. Since evidence is available for any variable in the BN model of a document, this simply amounts to a count for

each possible value of the random variables.

Using equation 5, the log-likehood for all documents in the corpus D is :

$$L = \sum_{d \in D} \sum_{i=1}^{|s_d|} \left(\log P(s_d^i | pa(s_d^i), \theta) + \sum_{k=1}^{|t_d^i|} \log P(w_{d,k}^i | s_d^i), \theta) \right) \quad (6)$$

Let us denote the model parameters $P(s_d^i|pa(s_d^i))$ and $P(w_{d,k}^i|s_d^i)$ by $\theta_{s_d^i, pa(s_d^i)}$ and $\theta_{w_{d,k}^i, s_d^i}$. Equation 6 then writes :

$$L = \sum_{d \in D} \left(\left(\sum_{i=1}^{|s_d|} \log \theta_{s_d^i, pa(s_d^i)} \right) + \left(\sum_{i=1}^{|s_d|} \sum_{k=1}^{|t_d^i|} \log \theta_{w_{d,k}^i, s_d^i)} \right) \right) \quad (7)$$

The learning algorithm then solves $\frac{\partial L}{\partial \theta_{n,m}} = 0$ with the constraint $\sum_n \theta_{n,m} = 1$.

Let $N_{n,m}^d$ be the number of times a part with label n has his parent with label m in document d or respectively the number of times a word with value n is in a part with label m for document d, the solution of the learning problem is :

$$\theta_{n,m} = \frac{\sum_{d \in D} N_{n,m}^d}{\sum_i \sum_{d \in D} N_{i,m}^d} \quad (8)$$

The complexity of the algorithm is $O(\sum_{d \in D} |s_d| + |t_d|)$. In a classical structured document, the number of node of the structural network is smaller than the number of words of the document. So the complexity is equivalent to $O(\sum_{d \in D} |t_d|)$ which is the classical learning complexity of the Naive Bayes algorithm. Note that, in the case of flat documents, our model is strictly equivalent to the classical Naive Bayes model.

5 Discriminant Approach

The above model could be used for different tasks, e.g. document classification or clustering or even for performing more sophisticated inferences on document parts, e.g. deciding which part of a document is relevant for a specific topic. For classification of whole documents which is the focus of this paper, discriminant approaches are most often preferred to generative ones since they usually score higher. We then propose below to derive from the generative model of structured document a discriminant model. For that, we follow the line of [9] who proposed to build a discriminant model from a generative sequence model. We show how this idea could be extended to our generative structured document model

5.1 Fisher Kernel

Given a generative model with parameters θ for sequences, [9] propose to compute for each sequence x the Fisher score $U_d = \nabla_\theta logP(x/\theta)$ of the model for the sequence, i.e. the gradient of the log likehood of x for model θ. For each sequence sample, this score is a vector of fixed dimensionality which explains how each parameter of the generative model contributes to generate the sequence.

Using this score, they then define a distance between two examples x and y as a kernel function :

$$K(x,y) = U_x^T M^{-1} U_y \text{ with } M = E_X[U_X^T U_X] \tag{9}$$

This kernel can then be used with any kernel classifier, (e.g. SVM) in order to classify the examples. The key idea here is to map the sequence information onto a vector of scores. This allows to make use of any classical vector discriminant classifier on this new representation and therefore to use well known and efficient vector classifiers for sequence classification. We show below that this idea may be naturally adapted to our structured generative model.

5.2 Fisher Kernel for the structured document model

For our model, the Fisher Kernel can be easily computed. Using 7 we get :

$$\frac{\partial P(d/\theta)}{\partial \theta_{n,m}} = \frac{N_{n,m}^d}{\theta_{n,m}} \tag{10}$$

The Fisher kernel idea initially proposed for HMMs, naturally carries over to our structured data model. However, in practice, using the Fisher Kernel method is not straightforward. In order to make the method work, one must make different approximations, especially when the number of parameters of the generative model is high which is the case here. In our implementation, we make the following approximations :

- we first approximate the M matrix using the identity matrix like in [9]
- we then compute the gradient of the log likehood wrt $2\sqrt{\theta_{n,m}}$ like in [8].

Let $\rho_{n,m} = 2\sqrt{\theta_{n,m}}$, we have : $\frac{\partial P(d/\theta)}{\partial \rho_{n,m}} = 2\frac{N_{n,m}^d}{\rho_{n,m}} = \frac{N_{n,m}^d}{\sqrt{\theta_{n,m}}}$

We use this last formula to compute the vector corresponding to each structured document d.

6 Experiments

6.1 Corpora

We use two corpora in our experiments.

WebKB corpus [20] is composed of 8282 HTML documents from computer science departments web sites. This is a reference corpus in the machine learning

community for classifying HTML pages. It is composed of 7 topics (classes) : *student, faculty, course, project, department, staff, other*. *Other* is a trash topic, and has been ignored here as it is usually done. We are then left with 4520 documents. We used Porter Stemming and pruned all words that appear in less than 5 documents. The size of the vocabulary V is 8038 terms. We only keep the tags with the higher frequency (*H1, H2,H3, TITLE,B,I,A*). We made a 5-fold cross-validation (80% on train and 20% on test).

INEX corpus [7] is the new reference corpus for Information Retrieval with XML documents. It was designed for ad-hoc retrieval. It is made up of articles from journals and proceedings of the IEEE Computer Society. All articles are XML documents. The collection contains approximately 15 000 articles from over 20 different journals or proceedings. We used Porter Stemming and pruned the words which appear in less than 50 documents. The final size of the vocabulary is about 50 000 terms and the number of tags is about 100. We made a random split using 50% for training and 50% for testing. The task was to classify articles into the right journal or proceedings (20 classes).

6.2 Results

	course	**department**	**staff**	**faculty**	**student**	**project**	**Macro**	**Micro**
Naive Bayes	0.96	0.93	0.07	0.67	0.91	0.65	0.70	0.81
BN Model	0.96	0.82	0.03	0.72	0.93	0.76	0.70	0.83
SVM	0.90	0.79	0.17	0.85	0.91	0.77	0.73	0.85
Naive Bayes Fisher	0.95	0.77	0.17	0.82	0.91	0.71	0.72	0.85
BN Model Fisher	0.95	0.83	0.14	0.84	0.94	0.72	0.73	0.87

Fig. 3. Performance of 5 classifiers on WebKB corpus

	Macro	**Micro**
Naive Bayes	0.61	0.64
BN Model	0.67	0.66
SVM	0.71	0.70
Naive Bayes Fisher	0.69	0.69
BN Model Fisher	0.72	0.71

Fig. 4. Performance of 5 classifiers on INEX corpus

We have used a Naive Bayes classifier as a baseline generative classifier and SVM ([10]) as a baseline discriminant model. Results appear in figures 4 and 3. *Macro-average* is obtained by averaging the percentage of correct classification

for every class considered. *Micro-average* is obtained by weighting the average by the relative size of each class.

Let us consider the micro-average. On WebKB, the BN model achieves a mean 3 % improvement with regard to Naive Bayes. This is encouraging and superior to already published results on this dataset [4]. The Fisher model still rises this score by 4%. This corresponds to 2% more than the baseline discriminant SVM. The structured generative document model is clearly superior to the flat Naive Bayes classifier, and the Fisher Kernel operating on the structured generative models compares well to the baseline SVM.

On the much larger INEX database, our generative model achieves about 2% micro-average improvement with regard to Naive Bayes and the Fisher Kernel method increases the BN score by about 6%, but only 1% compared to the baseline SVM. This confirms the good results obtained on WebKB. Note that, to our knowledge, these are the first classification results obtained on a real world large XML corpus.

These experiments show that it is important to take simultaneously into account structure and content information in HTML or XML documents. The proposed methods allow to model and combine the two types of information. Both the generative and discriminant models for structured documents offer a complexity similar to that of the baseline flat classification models while increasing the performances.

7 Conclusion and perspectives

We have proposed a new generative model for structured textual documents representation. This model offers a general framework for handling different tasks like classification, clustering or more specialized structured document access problems. We focused here on the classification of whole documents and described how to extend this generative model into a more efficient discriminant classifier using the Fisher kernel idea. Experiments performed on two databases show that the proposed methods are indeed able to take simultaneously into account the structure and content informations and offer good performances compared to baseline classifiers.

References

1. Jamie P. Callan, W. Bruce Croft, and Stephen M. Harding. The INQUERY Retrieval System. In A. Min Tjoa and Isidro Ramos, editors, *Database and Expert Systems Applications*, pages 78–83, Valencia, Spain, 1992. Springer-Verlag.
2. Soumen Chakrabarti, Byron E. Dom, and Piotr Indyk. Enhanced hypertext categorization using hyperlinks. In Laura M. Haas and Ashutosh Tiwary, editors, *Proceedings of SIGMOD-98, ACM International Conference on Management of Data*, pages 307–318, Seattle, US, 1998. ACM Press, New York, US.
3. Ludovic Denoyer, Hugo Zaragoza, and Patrick Gallinari. HMM-based passage models for document classification and ranking. In *Proceedings of ECIR-01*, pages 126–135, Darmstadt, DE, 2001.

4. M. Diligenti, M. Gori, M. Maggini, and F. Scarselli. Classification of html documents by hidden tree-markov models. In *Proceedings of ICDAR*, pages 849–853, Seatle, 2001. WA (USA).
5. Susan T. Dumais and Hao Chen. Hierarchical classification of Web content. In Nicholas J. Belkin, Peter Ingwersen, and Mun-Kew Leong, editors, *Proceedings of SIGIR-00*, pages 256–263. ACM Press, 2000.
6. Shai Fine, Yoram Singer, and Naftali Tishby. The hierarchical hidden markov model: Analysis and applications. *Machine Learning*, 32(1):41–62, 1998.
7. Norbert Fuhr, Norbert Govert, Gabriella Kazai, and Mounia Lalmas. INEX: Initiative for the Evaluation of XML Retrieval. In *Proceedings ACM SIGIR 2002 Workshop on XML and Information Retrieval*, 2002.
8. Thomas Hofmann. Learning the similarity of documents : An information-geometric approach to document retrieval and categorization. In *Research and Development in Information Retrieval*, pages 369–371, 2000.
9. Tommi S. Jaakkola, Mark Diekhans, and David Haussler. Using the Fisher kernel method to detect remote protein homologies. In *Intelligent Systems for Molecular Biology Conference (ISMB'99)*, Heidelberg, Germany, August 1999. AAAI.
10. Thorsten Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of ECML-98*, pages 137–142, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.
11. Jin H. Kim and Judea Pearl. A Computational Model for Causal and Diagnostic Reasoning in Inference Systems. In Alan Bundy, editor, *Proceedings of the 8th IJCAI*, Karlsruhe, Germany, August 1983. William Kaufmann.
12. David D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In *Proceedings of ECML-98*,, pages 4–15, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.
13. K. Murphy and M. Paskin. Linear time inference in hierarchical hmms, 2001.
14. Sung Hyon Myaeng, Dong-Hyun Jang, Mun-Seok Kim, and Zong-Cheol Zhoo. A Flexible Model for Retrieval of SGML documents. In *Proceedings of the 21st Annual International ACM SIGIR*, pages 138–140, Melbourne, Australia, August 1998. ACM Press, New York.
15. Benjamin Piwowarki and Patrick Gallinari. A Bayesian Network Model for Page Retrieval in a Hierarchically Structured Collection. In *XML Workshop of the 25th ACM SIGIR Conference*, Tampere, Finland, 2002.
16. B. Piwowarski, L. Denoyer, and P. Gallinari. Un modele pour la recherche d'informations sur les documents structures. In *Proceedings of the 6emes journees Internationales d'Analyse Statistique des Donnees Textuelles (JADT2002)*.
17. Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.
18. Ben Taskar, Peter Abbeel, and Daphne Koller. Discriminative probabilistic models for relationnal data. In *Eighteenh Conference on Uncertainty in Artificail Intelligence (UAI02)*, Edmonton, Canada, 2002.
19. Trec. Text REtrieval Conference (trec 2001), National Institute of Standards and Technology (NIST).
20. webKB. http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/, 1999.
21. Yiming Yang, Seán Slattery, and Rayid Ghani. A study of approaches to hypertext categorization. *Journal of Intelligent Information Systems*, 18(2/3):219–241, 2002.
22. Jeonghee Yi and Neel Sundaresan. A classifier for semi-structured documents. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 340–344. ACM Press, 2000.