A Scalable Topic-Based Open Source Search Engine*

Wray Buntine, Jaakko Löfström, Jukka Perkiö, Sami Perttu,
Vladimir Poroshin, Tomi Silander, Henry Tirri, Antti Tuominen and Ville Tuulos
Complex Systems Computation Group,
Helsinki Institute for Information Technology
P.O. Box 9800, FIN-02015 HUT, Finland.
{firstname}.{lastname}@hiit.fi

Abstract

Site-based or topic-specific search engines work with mixed success because of the general difficulty of the information retrieval task, and the lack of good link information to allow authorities to be identified. We are advocating an open source approach to the problem due to its scope and need for software components. We have adopted a topic-based search engine because it represents the next generation of capability. This paper outlines our scalable system for site-based or topic-specific search, and demonstrates the developing system on a small 250,000 document collection of EU and UN web pages.

1 Introduction

There is a strong commercial market and a good base of freeware software for the task of site search or topic specific search. This is the search engine task when restricted either by domain or by topic or sites crawled. Often, these search engines are packaged with a larger corporate intranet suite. The industry has grown up around the widely published opinion that[1]:

> "Knowledge workers spend 35% of their productive time searching for information online, while 40% of the corporate users report they cannot find the information they need to do their jobs on the internet."

Our research in this area is focused on developing the next generation of site search or topic-specific search engines, capable of handling 1–100 million pages. One branch of research here is the topic-specific crawler [6]. However, we are concerned with improving the *relevance* side of information retrieval, rather than solving problems with document collection. General articles and expert business opinion on the future of search agree on a typically nebulous statement of the form "understanding the user's intent." But what does this mean in practice? We present a first implementation of our general vision for this in the present paper. Our evolving system is freely available[2] under the GNU Public License.

The relevance side of information retrieval is generally considered to be an orthogonal measure to authority. Considerable research has extended the popular link-based authority scores such as Pagerank^{TM} to topic-specific authority measures [10]. Methods for improving the relevance of retrieved documents have been the subject of the TREC tracks organized by NIST. Until recently the dominant paradigm was simple versions of TF-IDF, using for instance pseudo-relevance feedback to incorporate empirically related words [18]. A recent promising area is the language modelling approach to retrieval [14], which is based on the simple idea that retrieval should seek to find a document D that maximizes a probability for the query Q of $p(Q|D, collection)$, rather than the earlier notion of $p(D|Q, collection)$ [12]. From a practical viewpoint, this means a change in emphasis from "model the users intent of Q, and then find matching documents D" to "model the content of each document D and then find queries Q matching the content". For the computational statistician, the difference is stark: discrete statistical modelling of documents is feasible whereas modelling of queries of size 2 or 3 is not. Thus we convert the nebulous business concept of "understanding the user's intent" into a feasible statistical task of modelling documents and generating query words from that model. Language modelling made its first major applied breakthough in the 2003 TREC Web track [7], where it ranked a strong first. In this paper we present an implementation for a new approach to language modelling based on recent developments in discrete principal components ana-

*In *Web Intelligence*, Beijing, China, Sept. 2004.

[1] Working Council of CIOs, Business Wire, February 2001

[2] http://cosco.hiit.fi/search/software.html

lysis.

In Section 2, we present the approach to language modelling and its use in the task of information retrieval. In Section 3, we outline the architecture of our system. In Section 4, we report on some experiments using our system. This demonstration uses a small collection of approximately 250,000 documents crawled from web sites related to the European Union and the United Nations. Finally, we offer some concluding remarks and discuss future directions in Section 5.

2 The Hierarchical Topic Model and Document Retrieval

Multi-aspect topic models are a statistical model for documents that allow multiple topics to co-exist in one document [11, 2, 9], as is the case in most newswire collections. We are scaling up the technique using a hierarchical topic model to allow efficient initialization and easier interpretation. These topic models are directly analogous to the Gaussian model of Principal Component Analysis (PCA), which in its form of Latent Semantic Analysis (LSA) has not proven successful in information retrieval. We demonstrate the successful use of this new version in the experimental section. Recall, the goal of the model is to assist in evaluating queries Q represented as a bag of words with the generative probability model $Pr(Q|D, collection)$, where D is the bag of words representation for a document in the collection. We wish to find a set of documents for which this probability score is the highest.

We call these models Multinomial PCA. The simplest version consists of a linear admixture of different multinomials, and can be thought of as a generative model for sampling words to make up a bag, for the Bag of Words representation for a document [1].

1. We have a total count L of words to sample.

2. We partition these words into K topics, components or aspects: $c_1, c_2, ...c_K$ where $\sum_{k=1,...,K} c_k = L$. This is done using a hidden proportion vector $\vec{m} = (m_1, m_2, ..., m_K)$. The intention is that, for instance, a sporting article may have 50 general vocabulary words, 40 words relevant to Germany, 50 relevant to football, and 30 relevant to people's opinions. Thus L=170 are in the document and the topic partition is (50,40,50,30). Proportions $\vec{m}$ are generated with a Dirichlet distribution [2].

3. In each partition, we then sample words according to the multinomial for the topic, component or aspect. This is the base model for each component. This then yields a bag of word counts for the k-th partition, $\vec{w}_{k,\cdot} = (w_{k,1}, w_{k,2}, ..., w_{k,J})$. Here J is the dictionary size, the size of the basic multinomials on words. Thus the 50 football words are now sampled into actual dictionary entries, "forward", "kicked", "covered" etc.

4. The partitions are then combined additively, hence the term admixture, to make a distinction with classical mixture models. This yields the final sample of words $\vec{r} = (r_1, r_2, ..., r_J)$ by totalling the corresponding counts in each partition, $r_j = \sum_{k=1,...,K} w_{k,j}$. Thus if an instance of "forward" is sampled twice, as a football word and a general vocabulary word, then we return the count of 2 and its actual topical assignments are lost, they are hidden data.

This is a full generative probability model for the bag of words in a document. The hidden or latent variables are $\vec{m}$ and $\vec{w}$ for each document, whereas $\vec{c}$ is derived. The proportions $\vec{m}$ correspond to the components for a document, and the counts $\vec{w}$ are the original word counts broken out into word counts per component.

2.1 Hierarchical Models

We use two computationally viable schemes for learning these models from data. The mean field approach [2, 4] and Gibbs sampling [15, 9]. Gibbs sampling is usually not considered feasible for large problems, but in this application it can be used to hone the results of faster methods, and also it is moderately fast due to the specifics of the model. The standard technique for scaling up general clustering algorithms is to introduce hierarchies (for instance, top-down partitioning as in [3]). We apply that kind of technology here. Multi-aspect models and Multinomial PCA, however, have not yet been made hierarchical, thus we present a method for doing that here. A second, and perhaps more important reason for building a hierarchy is for interpretability of the model. People cannot understand a large flat structure with 500 components. Instead, we present it as a meaningful hierarchy.

Hierarchies need to be forced into Multinomial PCA because the components naturally seek to become as independent as possible [5]. We do this by making the component proportions $\vec{m}$ correlated in a tree structure corresponding to a hierarchy. Represent the components numbered $1, ..., K$ then as a tree, where each index k has its parents, children, ancestors, etc. Note a component k in this tree can be an internal node or a leaf, but every node has its associated probability m_k.

The root probability m_0 represents the proportion of stop words for the problem, words statistically common in all documents. The children of m_0, say, m_1, m_2, etc. represent the proportion of words common to the top level topics. In a newspaper these would be a sports topic, a finance topic, etc., and m_1 would represent the proportion of common words in general sports document such as "win", "defend", "score", etc. Thus proportions for internal nodes rep-

resent shared/common words to the topic, and proportions at the leaf represent words specific to that low level topic, such as "Hong Kong stock market" or "world cup soccer".

A generating model for this kind of tree is

$$m_k = q_k n_k \prod_{l \in ancestors(k)} n_l(1 - q_l)$$

where q_k is the probability that one will remain at node k and not descend to its children, and n_l is the probability that child l will be chosen. Note $q_k = 1$ for each leaf node k, and for each parent node k, $\sum_{l \in children(k)} n_l = 1$. The probabilities q_k and n_k form a dual representation for m_k and the mapping is invertible. The tree generating probabilities $\vec{q}$ and $\vec{n}$ can be generated with a Dirichlet distribution, just as the original component probabilities $\vec{m}$ are in the non-hierarchical version [15, 2].

Algorithms for learning the component model to match a given tree are variations of the standard Multinomial PCA algorithms, mean field and Gibbs sampling, but have the advantage that the model can be grown top-down. The model can be built rapidly at the top levels because there are less parameters and less data is needed, cycles can be faster. Each level provides a natural and high quality initialization for the next level.

2.2 Relevance Models

The above models by themselves are excellent for organizing a document collection. However, for relevance testing in the language modelling approach to information retrieval [14], the models are too non-specific. A particular query is a highly specific task and a general statistical model lacks the nuances required to do successful relevance evaluation on that unique query.

Thus we instead adopt an approach akin to pseudo-relevance feedback [18]. Our point of departure is an initial set of 300 top documents as ranked by TF-IDF. The specific variation of TF-IDF we employ assigns the score

$$s(D, Q) = \sum_j \frac{d_j}{d_j + 1} \frac{q_j}{q_j + 1} \log \frac{N}{n_j}$$

to query bag Q with word counts q_j and document bag D with word counts d_j. n_j is the number of documents the word j appears in in the corpus. There are an endless number of variations of the TF-IDF formula; we have simply chosen a reasonable default, similar to the one used in the Lemur Toolkit[3].

We take the set of initial documents and build a specific Multinomial PCA model for it. We then evaluate the formula $p(Q|D, sub\text{-}collection)$ for the query Q for each of the top 300 documents D, and the sub-collection of 300 documents providing the Multinomial PCA components and their word distributions. Techniques for doing this are given in [5]. Thus we build a query specific model and use it.

[3] http://www-2.cs.cmu.edu/~lemur/

3 Architecture

The architecture of our system is fairly typical for an intranet engine. The search engine runs on a cluster of Linux machines in a fast local network. The overall architecture is illustrated in Figure 1. Rounded boxes represent algorithmic components, while cylinders represent components mainly concerned with data storage. The arrows between the subsystems show the flow of data from the corpus to more refined forms, finally leading to the capability to answer user queries.

The glue between the various components of the system, which are implemented in a variety of languages, is a simple socket protocol built on top of TCP/IP, which allows maximal modularity in a Linux environment. The protocol is message-oriented: a session between a client and a server consists of an arbitrary number of messages initiated by either side. In a typical setup, the server passively waits for requests and then responds to them.

Each of the subsystems, which we have divided into configuration, crawling, language processing, document modelling and query subsystems, has its own distribution strategy for scaling up. Next we describe the functions of the individual subsystems.

3.1 Configuration

The configuration subsystem is actually a "supersystem" that connects the subsystems together. The goal of modularity dictates that all subsystems be as independent from each other as possible. This architecture makes it possible to launch and kill individual subsystems without affecting the others, resulting in increased stability in the whole system.

In addition to parametrization and configuration services, the configuration subsystem is responsible for distributing the other subsystems to multiple machines. Real-time statistics and diagnostics about system status and performance are also collected here.

3.2 Crawling

The crawler subsystem is responsible for crawling web pages. Crawling is distributed by host: each crawler machine is allocated a portion of the space of host names. The allocations are obtained from hash codes computed from host names. Non-standard or even malicious web servers, broken HTML and the increasing proportion of dynamically generated content are everyday issues which make writing a robust crawler a challenge. Another key design goal is

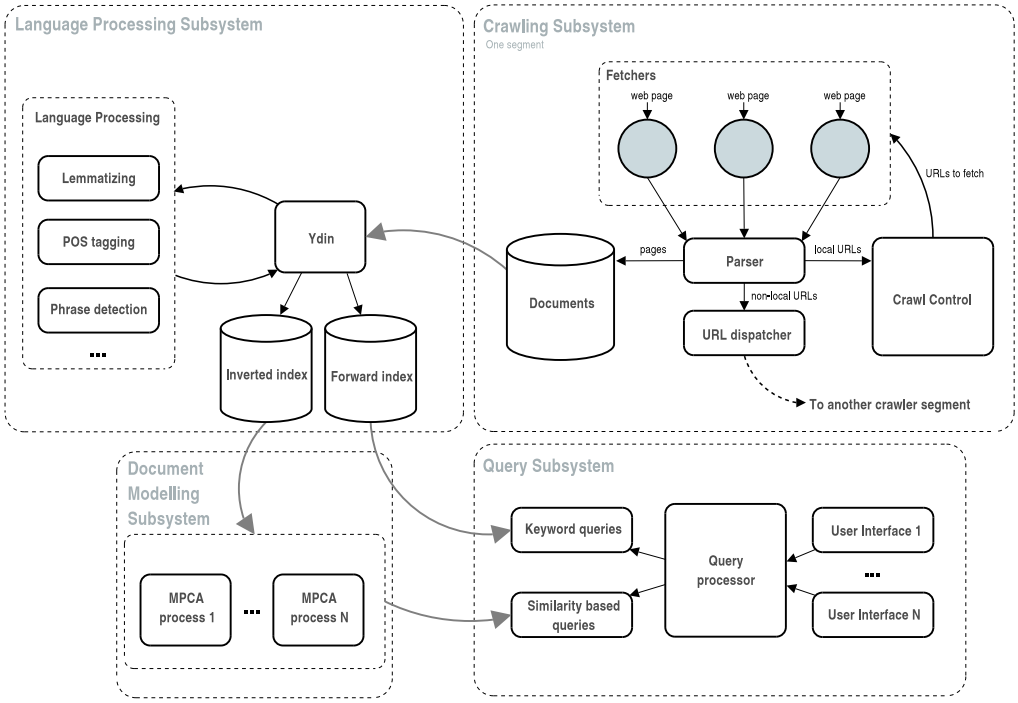

Figure 1. Search Engine Architecture.

the massive parallelism needed to achieve a good throughput without hammering a single host. This parallelism must also be turned into efficient local I/O. Writing a web crawler is something of a black art and we do not go into details here.

3.3 Language Processing

The language processing subsystem parses the raw documents obtained by the crawler and builds the lexicon and the forward and inverted indexes. The core, a program called Ydin, is written in C++ to be as efficient as possible.

A custom HTML parser scans the documents and converts them into an internal representation. Content is then part-of-speech (POS) tagged and lemmatized. The results are inserted into a database as document bags and inverted indexes. We are using a custom hash table based database system here, but are also investigating the use of standard solutions such as Berkeley DB.

Part-of-speech tagging is currently performed by Connexor[4], a commercial Functional Dependency Grammar (FDG) parser. In the future an open source alternative to Connexor will also be provided.

Lemmatization, i.e., normalization of words to their root forms, is more accurate than simple stemming, such as that performed by the classical Porter stemmer algorithm. While word form normalization is important when dealing with morphologically rich languages such as Finnish, we argue that document modelling benefits from it as well, as there are less words to model – computational costs of running MPCA are high – and the words that are left have more comprehensive statistics. Part-of-speech information, on the other hand, enables word class specific document modelling, as explained in the previous section.

The document collection is trivially distributed by allocating each language processing machine a set of documents. The machines then build their own lexicons and indexes. The only global statistics needed is n_j, word frequencies for TF-IDF. They have to be computed separately and the distribution there is by word[13].

3.4 Document Modelling

The document modelling subsystem builds a topic model from a set of document bags. This is the most computationally intensive task in the whole process of setting up a search engine for a document collection. The process naturally distributes in the same manner as statistical clustering. The hierarchical approach adds some structure to the problem and allows better scaling. Using hierarchical topics, the top levels of the hierarchy can be built efficiently on subsets of the data. These top levels then provide a good initialization for processing on the full document collection.

[4] http://www.connexor.com/

3.5 Queries

The query subsystem answers user queries. At the moment, we support TF-IDF queries, as a baseline, and topic queries. The query subsystem is distributed in the same fashion as the language processing subsystem - the space of documents is divided into disjoint sets, with a separate inverted index for each set. The topic model, which is built globally at first, is distributed in the same fashion. For more speed, many copies could be run in parallel on the same index or model.

4 Experiments

For the experiments in this paper, we crawled a small collection of approximately 230,000 HTML and 15,000 PDF documents from 28 EU and UN related sites. Linguistic preprocessing was as follows. The 50 major stop words (including all word classes except nouns) were eliminated. Only the top 3000 numbers where included (tokens such as "1996", "third", "20/20", etc.). Words were split into nouns and abbreviations (96957), verbs (3720), adverbs (1596), adjectives (13019), participles (5571) and other (connectives, determiners, prepositions, etc., 348 in total). Words with less than 5 instances or occurring in less than 3 documents were removed. This left 121211 lexemes broken up into 6 categories each modelled with separate multinomials.

4.1 Building a Hierarchy

A first experiment we performed was to build a topic hierarchy using the method described in the previous section. This is the largest Multinomial PCA model published to date by an order of magnitude. The model was built in phases: (1) the top level of 10 nodes and the root, (2) the second level of 10 sets of 10 nodes for the above, (3) and then free floating expansion of subsequent nodes using a branching factor of 5 once the parent node had stabilized. Thus the top two levels are balanced with a branching factor of 10, and the subsequent levels are unbalanced with a branching factor of 5. The final model had 511 components in total. This took 50 hours of time on a dual CPU with 3GHz processors and 2GB of memory. About 5GB of disk was used.

Some detail of the topic hierarchy are given in the tables below. The phrase summaries for these topics have been entirely automatically generated by looking for distinctive nominal phrases appearing in documents associated with a topic. Full automatic naming of the topics is not feasible: the meaning of a topic is essentially its documents and summarization of documents is a hard task, requiring a deep understanding of the semantics of text. Thus we use phrase summaries instead, which provide good overviews of the topics.

The hierarchy has two top levels with a branching factor of 10, resulting in 111 top nodes, and subsequent nodes have a branching factor of 5. Shown are the top level Nodes 1–10, the children of Node 3, 3.1–3.10, the children of node 3.1, 3.1.1–3.1.5, and the children of node 3.2, 3.2.1–3.2.5. In most cases, the phrase summaries are quite clear, though they might be confusing to the general public. Nevertheless, this demonstrates our model building technology, and we believe these would make a strong topic hierarchy for browsing and indexing documents.

1	programme; rights; people; States; Conference; world; Nations; Council; women; region;
2	Council; Europe; groups; Commission; European Union; Council of Europe; European Parliament; drugs; European Agency; European Convention;
3	countries; development; people; policies; world; society; population; Office; study; Union;
4	States; members; services; Union; rights; community; Member States; EU; European Union; case;
5	system; activities; project; network; sustainable development; water; European Environment Agency; European Topic Centre; Research Networks;
6	information; products; site; section; documents; list; United Nations; staff; Information Services; web site;
7	Agency; Phone; environment; Denmark Phone; Environment Agency; European Environment Agency; industry; production; report; companies;
8	development; information; programme; project; issues; technology; partners; trade; investment; Institute;
9	years; data; Article; agreement; persons; rate; education; Government; $; Act;
10	development; States; policies; years; report; meeting; Commission; Committee; action; services;

Table 1. Top Level Topics

4.2 Evaluating Queries and Results

A second experiment we performed is to search using the modelling technology of Multinomial PCA as the relevance method, as described in the previous section. We took EU and UN relevant queries from the TREC *ad hoc* query set. We used queries 401, 404, 407, 409, 410 and 412 as the first 6 queries in the 401-450 set relevant to EU or UN. Queries cover topics such as poaching on wildlife reserves, the Ireland peace problem, and the Schengen agreement. We used the title and description fields as the query words. We ran these queries through our standard Lemur-like TF-IDF evaluation, and using the Multinomial PCA relevance evaluation. The top ranked 25 results from each were then rated together in a blind test so the rater had no knowledge of the method. Rating used the scale 1-5, with higher being better.

Comparative results are given in Figures 2-4. The bars show the average relevance of both methods at ranks 5, 10, 15, 20 and 25, i.e., average relevance at rank 5 is the average relevance of the top 5 documents as rated by the method. The topic model is noticably better than TF-IDF in 3 queries (404, 409, 410); the methods are about equal in queries 407 and 412; and TF-IDF is better in query 401.

3.1	project; Republic; assistance; funds; monuments; contribution; programme; donors; building; disasters;
3.2	schools; students; study; pupils; University; book; primary schools; films; secondary schools; grade;
3.3	cities; Asia; areas; development; town; settlement; authorities; habitats; local authorities; region;
3.4	European Commission; Delegation; Union; European Union; EU; European Commission's Delegation; Europe; relations; co-operation; Member States;
3.5	America; agriculture; countries; Latin America; developing countries; economy; farmers; Caribbean; world; system;
3.6	population; families; Centre; poverty; education; family planning; Philippines; Conference on Population;
3.7	century; links; Africa; media; site; Partner Institutions; Links with Partner Institutions; UNESCO; journalists; Biosphere reserves;
3.8	Delegation; children; people; Head; Chairman; President; elections; room; young people; parties;
3.9	University; Science; Office; Director; team; technology; Professor; Box; UNEP; Library;
3.10	per cent; China; development; goods; period; services; training; administration; economic growth;

Table 2. Mid Level Topics under 3 "Countries, Development, People, Policies"

3.1.1	Republic; monuments; Yugoslav Republic; former Yugoslav Republic; phase; People's Republic; Democratic Republic; cultural heritage; Islamic Republic; Republic of Macedonia;
3.1.2	funds; contribution; income; ECHO; total cost; Trust Fund; credit; volunteers; region; Development Fund;
3.1.3	donors; countries; disasters; Iran; cooperation; natural disasters; Democratic Republic; Democratic People's Republic;
3.1.4	building; community; programme; Department; latest major documents; emergency; UNICEF; Emergency Report; WFP Emergency Report;
3.1.5	resources; Coordinator; assessment; contribution; forestry; consortium; technical assistance; preparation; June; Burundi;

Table 3. Low Level Topics 3.1 "Projects"

3.2.1	mission; University; programme; activities; Yearbook; High Representative; rehabilitation; programs; crafts; higher education;
3.2.2	supply; images; electricity; water supply; Office; metric tons; cereals; food supplies; urban areas; energy supply;
3.2.3	students; book; project; minorities; training; Association; young people; national minorities; English; members;
3.2.4	TV; audience; TV channels; TV equipment transmissions facilities; audience market share Volume; TV production volume; TV programming; satellite TV channels; TV fiction; Social Council;
3.2.5	study; degree; publication; case studies; grade; population; cities; rural areas; comparative study; Arabic version;

Table 4. Low Level Topics 3.2 "Schools"

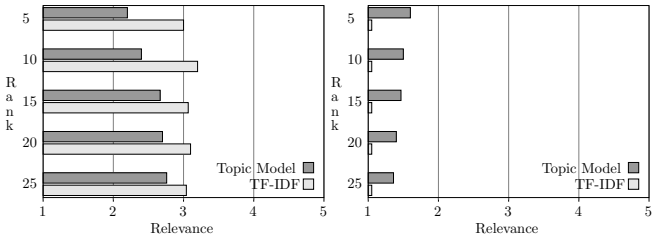

Figure 2. Query 401 and 404 comparisons.

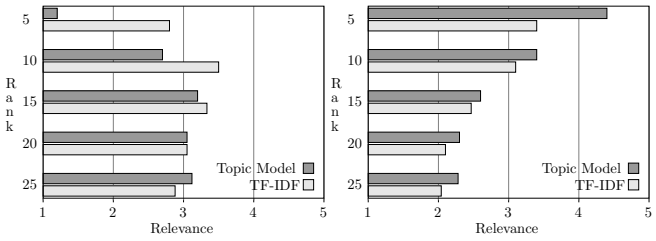

Figure 3. Query 407 and 409 comparisons.

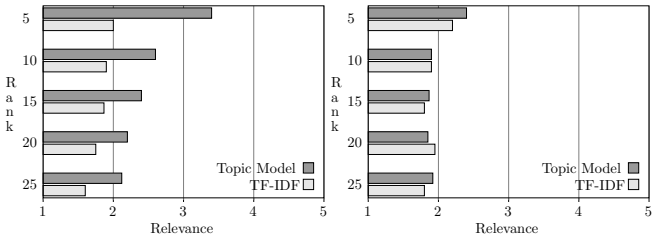

Figure 4. Query 410 and 412 comparison.

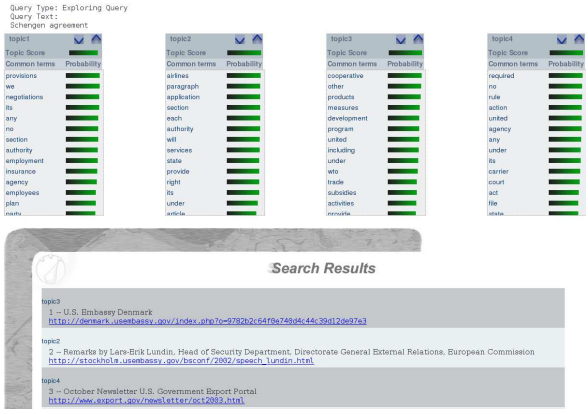

Figure 5. A screenshot of the user interface.

Because the system built a component model on the 300 documents, we can display these components and allow the user to specialize their query to one component or another. An example of the interface used here is in Figure 5. This turns out to be very interesting. For the poaching query, only one component in the 10 generated corresponds to relevance to the query, whereas in the Schengen agreement query, several components correspond. The TREC queries are very specific. In more general queries, multiple components indicate the query appears to have multiple responses and should be specialized.

5 Conclusions

First, we have demonstrated that coherent hierarchical topic models can be built automatically on large document collections. The topic models provide a content analysis of documents rather than a partitioning of document space achieved by traditional clustering methods [3], thus they are more suitable for supporting user navigation since any one document can have multiple topics. Our next task is to integrate these topic models into the business of search, so that users can key on topics of interest, and have displayed the topical content of results found. We can achieve topical specialization during search.

Second, we have demonstrated our relevance method and shown that it returns results that are very much different from standard TF-IDF, and somewhat better in the small test set used. More extensive testing needs to be performed in the future. Because of the number of opportunities for improving the method demonstrated, and the interaction modes available, we believe significant improvements should be obtainable. For instance, the initial query could be expanded with related words to get a richer document set for subsequent modelling.

A second aspect of relevance for retrieval is allowing the user to provide feedback. Folklore from the IR community is that more sophisticated user feedback such as Scatter-Gather for results clustering [8] improves the user experience but has little real improvement in terms of quality of results (e.g., they feel "empowered"). Recent experience in the TREC HARD Track goes against this: clustering of results is useful in interactive search tasks [17], although so-called statistically optimized systems for presenting candidate documents to users for relevance checking have not yet proven useful [16]. Since our relevance method has produced a clustering, we also have the option of displaying this to the user. Our informal experience is that Multinomial PCA does a good job of teasing out the components in the results, and may well support interactive search as well.

References

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[3] D. Boley, M. Gini, R. Gross, E.-H. Han, G. Karypis, V. Kumar, B. Mobasher, J. Moore, and K. Hastings. Partitioning-based clustering for web document categorization. *Decis. Support Syst.*, 27(3):329–341, 1999.

[4] W. Buntine. Variational extensions to EM and multinomial PCA. In *ECML 2002*, 2002.

[5] W. Buntine and A. Jakulin. Applying discrete pca in data analysis. 2004.

[6] S. Chakrabarti, M. van den Berg, and B. Dom. Focused crawling: A new approach to topic-specific web resource discovery. In *8th World Wide Web*, Toronto, 1999.

[7] N. Craswell and D. Hawking. Overview of the TREC 2003 web track. In *Proc. TREC 2003*, 2003.

[8] D. R. Cutting, J. O. Pedersen, D. Karger, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proc. of the 15th Annual International ACM SIGIR Conference*, pages 318–329, 1992.

[9] T. Griffiths and M. Steyvers. Finding scientific topics. *PNAS Colloquium*, 2004.

[10] T. Haveliwala. Topic-specific pagerank. In *11th World Wide Web*, 2002.

[11] T. Hofmann. Probabilistic latent semantic indexing. In *Research and Development in Information Retrieval*, pages 50–57, 1999.

[12] K. S. Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments - part 2. *Information Processing and Management*, 36(6):809–840, 2000.

[13] S. Melnik, S. Raghavan, B. Yang, and H. Garcia-Molina. Building a distributed full-text index for the web. In *10th World Wide Web*, pages 396–406, 2001.

[14] J. Ponte and W. Croft. A language modeling approach to information retrieval. In *Research and Development in Information Retrieval*, pages 275–281, 1998.

[15] J. Pritchard, M. Stephens, and P. Donnelly. Inference of population structure using multilocus genotype data. *Genetics*, 155:945–959, 2000.

[16] S. E. Robertson, H. Zaragoza, and M. Taylor. Microsoft cambridge at TREC-12: HARD track. In *Proc. TREC 2003*, 2003.

[17] J. Shanahan, J. Bennett, D. Evans, D. Hull, and J. Montgomery. Clairvoyance Corporation experiments in the TREC 2003 high accuracy retrieval from documents (HARD) track. In *Proc. TREC 2003*, 2003.

[18] A. Singhal and M. Kaszkiel. A case study in web search using TREC algorithms. In *Proc. of WWW10*, 2001.