Abstract

Standard search engines prove to be very useful if they have access to huge amounts of data. However, a common problem is search over a restricted domain. This paper addresses this problem by indexing the source data in a more elaborate way than in standard search engine technology. This allows us to extract concepts that are used to create a structure for the documents that is similar to that found in classified directories.

Overview

The vast amount of data that can be found on the Web helps to find information on specific topics very quickly. However, a common problem is to find the appropriate documents in well defined subdomains of the Web. For example the search for lecturers in AI on the Essex university Web pages is not successful even though the information is there, but not explicitly on a single page. A related problem is the retrieval of too many documents: searching for lecturers gives us too many matches in an apparently random order.

Our approach aims at creating a similar structure as can be found in classified directories, only that this is done dynamically for a set of Web pages. This involves extracting terms that function as classifications or cross-references. Uncovering this structure is done by indexing the same document in a variety of ways. This gives us a number of index tables which can then be exploited in order to find the important terms. These terms, our concepts, then function as classifications like in classified directories. We select a concept for a document if it is found in:

- the document title and in the meta tags or
- in the document title and in a heading and in a bold font tag or
- in a heading and in a bold font tag and in the text.

Similarly we select concepts for directories in which the documents are found.

In addition we define a second classification level based on links between documents, that are hyperlinks from or to documents but also links based on the file structure where these documents were found, for example two documents are linked if they are found on the same server in the same directory tree. A dialogue system which exploits this knowledge can then guide the user in cases where the queries were not successful.

In case of too many matching documents for a query the system could automatically constrain the query by searching for concepts only. In case of the user request for lecturers this would result in only a few matching documents which relate to the extracted concept (i.e. classification) lecturer.

Alternatively, we can now apply the concepts that were extracted in the indexing process, more specifically the related concepts1. If we define two concepts to be related if a document exists for which both of these were extracted, then we can add each of these related terms to the user query and offer it as one of the options the user can choose from. For example, imagine the term lecturer is related to computer_science, art_history, course etc., then the user could be offered to continue the search by selecting one of these related terms to constrain the query. To compare this with adverts in a classified directory this means you are now looking for those entries that are listed under two different classifications at the same time.

In case of too few matches we can look for those documents that partially match the query and which are related to other documents that match the rest of the query. This relation can be based on related concepts or the link structure.

In the remainder of the paper we will first discuss related work and then describe offline (index construction) and online (dialogue handling) processes.

Related Work

Generally speaking, we look at the problem of intelligent indexing. There are various research perspectives of how this could be tackled. They all differ in some

1comparable to cross-references in classified directories
respect from the assumptions we make about our data: a subdomain of the Web containing documents with no specified structure, typically the Web pages of a company, an organisation or some institution.

Indexing documents has been a research topic in the information retrieval (IR) community for many years. But documents are normally very long, contain little internal structure and collections are “typically measured in gigabytes” (Zhai 1997). The addition of a Web track to the TREC conference series highlights the importance of finding new methods for Web search.

Concerning document size and structure, the same is true for document clustering. A recent example of conceptually indexing a document collection is Keyphind which is described in (Gutwin et al. 1999). Machine learning techniques are applied in order to extract keyphrases from documents in the context of browsing digital libraries. This comes close to our idea of imposing a structure on the collection by extracting “important” phrases from each document, but here the documents are long enough to extract phrases from the raw text and furthermore a manually tagged training corpus is needed to build the classifier. Extractor is a similar system for extracting keyphrases using supervised learning (Turney 1999; 2000).

Clustering is also being used for concept-based relevance feedback for Web information retrieval (Chang & Hsu 1999). Following a user query the retrieved documents are organised into conceptual groups. Unlike in our approach this structure is not extracted for the indexed domain but for the search results.

Ontologies and customised versions of existing language resources like WordNet (Miller 1990) are being successfully employed to search product catalogues and other document collections held in relational databases (Guarino, Masolo, & Vetere 1999; Flank 1998). Part of that research is the actual construction of ontologies (Craven et al. 1998). The cost to create the resources can be enormous and it is difficult to apply these solutions to other domains where the document structure is not known in advance.

Quite a different way of dealing with the semi-structured information on the Web is to retain the structure and store the data in graph-structured data models by means of wrappers (Mattoo, Seligman, & Smith 1999; Sahuguet & Aznavant 1999; Thomas 1999). It is not just retaining but also capturing the structure, for example to transform HTML documents into XML or other formats. Databases and query languages have been developed for this, the most prominent database system is Stanford’s Lore (McHugh et al. 1997). But the drawback is that the indexing depends very much on a formally defined structure of the expected input. There has so far also been little experience using semi-structured databases for substantial applications (Seligman et al. 1998).

In the Clever Project (Chakrabarti et al. 1999) no assumptions are made about the documents. The domain is the complete Internet and the problem to be solved is filtering out those pages which are truly relevant for a specific topic, i.e. the problem of too many matches. Authorities and hubs are distinguished, places that are either relevant or are collections of links to those pages, respectively. Authorities and hubs are found by purely analysing the connections between Web pages.

Hyperlink Vector Voting is introduced in (Li 1998). Rather than depending on the words appearing in the documents themselves it uses the content of hyperlinks to a document to rank its relevance to the query terms. That overcomes the problem of spamming within Web pages and seems appropriate for Internet wide search but would cause problems in subdomains where the number of links between documents is much smaller and certainly problems will occur for those pages which are referred to by a small number of documents only or no documents at all. One can go further by not just using the anchor text but also additional structural information found in the context of the hyperlink as explained in (Fürnkranz 1999). For the task of classifying pages using a given set of classes it is reported that it is possible to classify documents more reliably with information originating from pages that point to the document than with features that are derived from the document text itself.

The Cha-Cha system has been developed for intranets. It imposes an organisation on search results by recording the shortest paths to the root node in terms of hyperlinks (Chen et al. 1999). But this is only applied once results could be found by using the search engine. It exploits hyperlinks but ignores the internal structure of the indexed Web pages.

With the YPA we implemented a system that addresses a similar problem as described here where a user is looking for advertisers that could provide certain goods or services (De Roeck et al. 1998; 2000). The documents are not Web pages but advertisements from BT’s Yellow Pages and Talking Pages. But rather than having to build classifications and cross-references these were already an implicit part of the input data sources.

Extracting Concepts

Once a database of index tables exists this can be applied in an online search as long as the search engine knows how to usefully combine different tables. This is actually part of UKSearch, but we can do better than just that. In an offline process following the indexing we extract concepts from the index tables which will allow us to distinguish important and not-so-important keywords in a document or directory.

Purely looking at meta tags does not help. Part of the problem is that these tags are optional and often

\[3\text{Yellow Pages® and Talking Pages® are registered trade marks of British Telecommunications plc in the United Kingdom.}\]
used very infrequently. Considering those meta tags that contain keywords or a description we detected that in our sample domain they are used in little more than a third of all pages. In a subdomain of that (Computer Science Department Web server) this is even worse: less than 10% of all pages contain this sort of tags. Instead, for each document we look for occurrences of an index term in different index tables. This is how we extract our concepts. Currently we say an index term is selected as a concept for a document if it is found in:

- the document title and in the meta tags or
- in the document title and in a heading and in a bold font tag or
- in a heading and in a bold font tag and in the text.

We can extend the selection criteria by combining these index tables in other ways. A positive side effect is that we are resistant to spamming of Web pages in many cases, i.e. multiple entries of a keyword in a meta tag, because this is being ignored unless the keyword shows up somewhere else as well. However, at the same time we relax this definition of concepts by defining soft concepts which are actually selected by using less strict extraction rules. Soft concepts for a document are those index terms which were found in:

- the document title or
- the meta tags or
- in any two different index tables.

Since not all documents can be classified by the initial concept extraction process, we now have the basis for a sensible fallback strategy.

What we described is the process of classifying documents. This is also being applied to directories. A simple heuristic approach is to assign those concepts to a directory which turn up most frequently for the documents found in it. Not only do we get a directory classification but a hierarchy of classifications based on the directory tree structure. Figure 1 gives an example of directory names and some corresponding concepts.

**User Dialogue**

The actual user search is performed in a dialogue system which for most of the cases just accepts a query and returns the results. As long as we find a good number of matching documents (i.e. not too many or too few) for a request UKSearch works like most other search engines, only that various index tables are being combined.

But in any other case we either automatically relax/constrain or ask the user to decide what is the most appropriate way to continue. Let us come back to the original example: lecturers in AI. Assuming in a certain setup of the system we would not get any answers (because automatic relaxation is switched off). In this case the dialogue manager should give the user as many choices as appropriate in order to finally retrieve documents which could be relevant. For example, the system would ask the user to relax the query (in an input form provided) or to choose between the following options:

- search for partial matches
- display documents found for AI with links to lecturers
- display documents found for lecturers with links to AI
- find directories which contains matches (but not in a single document)
- do query expansion by adding related terms.

A word about the links, these are not necessarily hyperlinks but any sort of link that was detected in the index construction process as explained earlier. For example we can set up the dialogue system that it only expands hyperlinks or connections between documents in the same directory or one level up or down.

In case of too many matching documents for a query the system's response now depends on the setup of the system or the chosen user options. Which of the following strategies is used depends entirely on that setup:

1. The system can automatically constrain the query by searching for the best matches, for example looking for matches in classifications only, i.e. retrieving those documents that denote to concepts matching the query. If this fails, then soft concepts can be searched. In case of the user request for lecturers this would result in only a few matching documents which relate to the extracted concept (i.e. classification) lecturer.

4Our experience with the YPA is that automatic query term expansion can worsen precision quite dramatically, which is why a user option seems more appropriate (see (Kruschwitz et al. 2000)).
2. In a very similar way we could search for those documents which are classified under certain concepts and where the same holds for the directories in which they are stored.

3. We can apply the concepts that were extracted in the indexing process, more specifically the related concepts (cross-references). If we define two concepts to be related if a document exists for which both of these were extracted, then we can add each of these related terms to the user query and offer it as one of the options the user can choose from. For example, imagine the term lecturer is related to computer_science, art_history, course etc., then the user could be offered to continue the search by selecting one of these related terms to constrain the query. To compare this with adverts in a classified directory this means you are now looking for those entries that are listed under two different classifications at the same time.

This needs of course some fine tuning in the concept extraction rules but in the end it works without manual customisation. Initial results demonstrate this, though more tests are needed. For the Computer Science Web pages we found for example the following related concepts without any fine tuning tool, web, network, guide, terena and in a second set essex, computer_science, university, essex.university. When it comes to the classification of directories we can say that generally more specific concepts turn up further down in the directory tree.

Plans are to expand these sets and hierarchies of concepts by incorporating other sources, e.g. domain independent sources like WordNet (Miller 1990) or further exploitation of the directory as well as hyperlink structure. We will also have to evaluate properly which tags are most useful to extract the concepts, for example we currently ignore HTML tags like <i> and <em> completely.

**Implementational Issues**

The robot as well as most of the index construction programming is done in Perl making use of the existing modules (LWP, HTML, URI etc.). In the indexing process we make also use of the Brill part-of-speech tagger (Brill 1992) and WordNet. For the online dialogue system we could have reused parts of the YPA, but to avoid intellectual property conflicts this is now being completely rewritten. Like the YPA it is based on a Sicstus Prolog executable which is accessed via sockets.

Online and offline processes all run on a Sparcstation Ultra 10 with 128 MB working memory.

We focus on indexing a sample subdomain of the Web, the Essex university Web pages. This is an arbitrary choice and once the framework has been fully implemented we plan to validate the approach using different domains. However, as stated earlier we do not aim at searching the Web in general, because for this purpose standard search engines are usually sufficient and more efficient. Looking at subdomains of the Web also allows more flexibility concerning performance issues, naturally sophisticated indexing is computationally much more expensive than simple keyword indexing.

Evaluation will be a major task to see how the techniques actually compare to other approaches. It is yet to early to do this.

**Acknowledgements**

Thanks to Sam Steel for discussing earlier versions of the paper (even though he had enough other things to do). Thanks to the helpful comments of the anonymous reviewers.

**References**


Chang, C. H., and Hsu, C. C. 1999. Enabling Concept-Based Relevance Feedback for Information Retrieval on the WWW. IEEE Transactions on Knowledge and Data Engineering July/August.


Fürnkranz, J. 1999. Exploiting Structural Information for Text Classification on the WWW. In Proceedings of


