Local search engine with global content based on domain specific knowledge

SANDI POHOREC, MATEJA VERLIČ, MILAN ZORMAN
Laboratory for system design
University of Maribor, Faculty of Electrical Engineering and Computer Science
Smetanova 17, 2000 Maribor
SLOVENIA
sandi.pohorec@uni-mb.si, mateja.verlic@uni-mb.si, milan.zorman@uni-mb.si, http://www.lsd.uni-mb.si

Abstract: - In the growing need for information we have come to rely on search engines. The use of large scale search engines, such as Google, is as common as surfing the World Wide Web. We are impressed with the capabilities of these search engines but still there is a need for improvment. A common problem with searching is the ambiguity of words. Their meaning often depends on the context in which they are used or varies across specific domains. To resolve this we propose a domain specific search engine that is globally oriented. We intend to provide content classification according to the target domain concepts, access to privileged information, personalization and custom ranking functions. Domain specific concepts have been formalized in the form of ontology. The paper describes our approach to a centralized search service for domain specific content. The approach uses automated indexing for various content sources that can be found in the form of a relational database, web service, web portal or page, various document formats and other structured or unstructured data. The gathered data is tagged with various approaches and classified against the domain classification. The indexed data is accessible through a highly optimized and personalized search service.

Key-Words: - information search, personalization, indexes, crawling, domain specific crawling, natural language processing, content tagging, distributed data sources, ranking functions

1 Introduction

We have looked at the problem of developing a search engine through the perspective of one organization. The organization's domain scope is limited with all areas of interests of every single member. The indexed content is global by its location, although the only sites indexed are those that are related to the target domain. The main objective is to provide an optimal search system for a particular organization. The natural language processing being done in the indexing and tagging process is domain oriented. We use ontology of domain concepts to resolve ambiguity of word meaning. The sources that are indexed can be in a variety of forms and formats. The system supports web sites and services, unstructured data (documents), structured data (XML files) and data bases. The system provides personalization of content and automatically uses user credentials to limit access to restricted content. The personalization is important because it improves the quality of users information retrieval, reduces time spend, and automatically classifies new content according to the users profile. The presented system is intended as a practical implementation of several well known methods and approaches in order to evaluate their use in a domain specific environment.

2 Introduction to modern web search engines

Modern web search engines [1] use a loosely coupled architecture which consists in two major modules; crawling and searching. The joining factor between the two is the search index. This allows that the crawling and searching can be relatively independent. Their internal data structures, functions and programming logic can change at any time as long as they still conform to the design of the search index. This independency along with the fact that web crawling is an extremely time consuming task has led to the loosely coupled architecture. The entire process of a web search engine is presented in a vertical hierarchy on Fig. 1. On the bottom is a representation of the content to be searched (most commonly for a large scale engine the content is the entire World Wide Web). The crawlers are transferring the content to the search index, which in turn executes the queries submitted by the users via web interface.

We continue with the examination of the three main components (crawling, indexing, searching) individually and conclude this chapter with a short overview of search engines in the semantic web and an introduction to personalization.

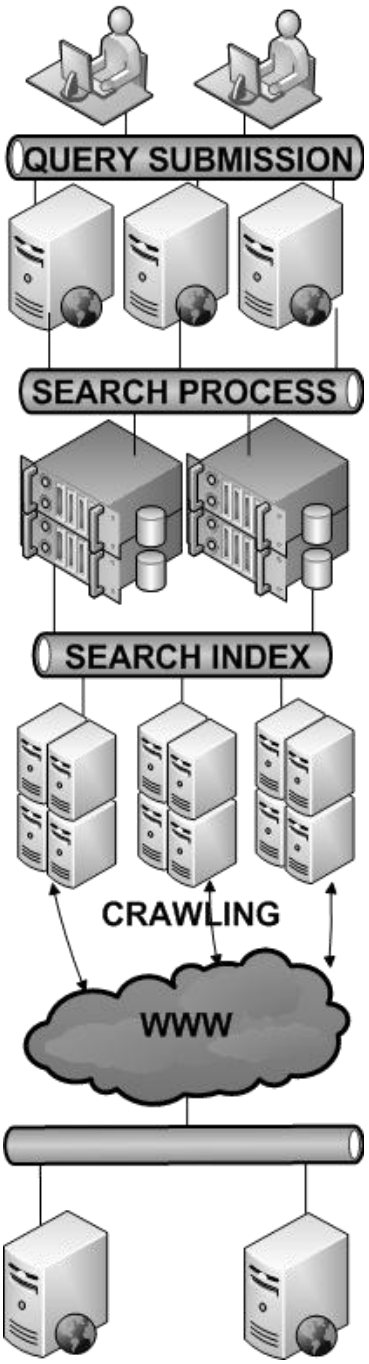

Fig. 1: Vertical architecture of a search engine

2.1 Crawling

Web crawler [2][3] also known as a spider has the task of :

- continuously visiting web pages,
- downloading their content,
- transforming from various formats to plain text and
- performing the search index updates.

Three important properties of the World Wide Web that have a significant impact on crawling are:

- extremely large quantities of content,
- fast rate of content changes, and
- dynamic web pages.

Especially the ever increasing number of dynamic web pages makes the efficient crawling problem exponentially harder. Because of dynamic content there is a significant probability that different combinations of HTTP GET parameters will result in retrieval of the same content. This reduces the efficiency of the crawler and is a major obstacle because neither the bandwidth nor the time are available in unlimited quantities. The generalized high level architecture of crawlers is presented in Fig. 2. As we can see a crawler maintains a list of sites to be crawled (queue) and the process scheduler component schedules the execution of every job (site to be crawled). When a site is scheduled to be crawled, a multithreaded component opens multiple connections simultaneously. It crawls multiple pages at the same time to limit the number of requests to a particular site and at the same time to increase the efficiency of crawling. The most important factor in understanding the crawler is to know its scope. A crawler can be global, such as "GoogleBot", or local (only indexes certain domains, only sites in a certain language or even just a single web site).
The strategies that define and limit the crawler's behavior during crawling are the following:

- limitation to link following (the crawler follows only links that link to a certain site or domain or it can follow all links),
- technique of priority crawling, either depth-first (only one link on any depth is crawled) or breadth-first (every link is followed before descending a level),
- use of narrow crawling (only certain content is crawled) and
- crawling the deep part of internet (content that is hidden and access is restricted to authorized users only).

The policy on revisiting pages can be either:

- a uniform policy (same for all pages) or
- a proportional policy (more frequent visits to pages with more changes).

Whether the search index is up to date and to what degree depends on what time interval of sequential visits was selected for the crawler. An important factor that describes a crawler is its behavior during the actual crawling of a particular web site. A crawler should conform to rules provided in the robots.txt file. The file

restricts which areas a particular crawler can visit and provides guidelines for general crawling behavior. The speed of crawling is essential so that the crawler does not overload the targeted web site. Certain sites such as Wikipedia provide data dumps of their content in order to avoid extensive crawling. The final important factor of a crawler is the degree of parallelism. This determines whether only one crawler operates on any given time or are there multiple crawlers that operate simultaneously. If there is more than one then the workload allocation algorithm, that distributes the workload evenly among the crawlers, is also an important factor.

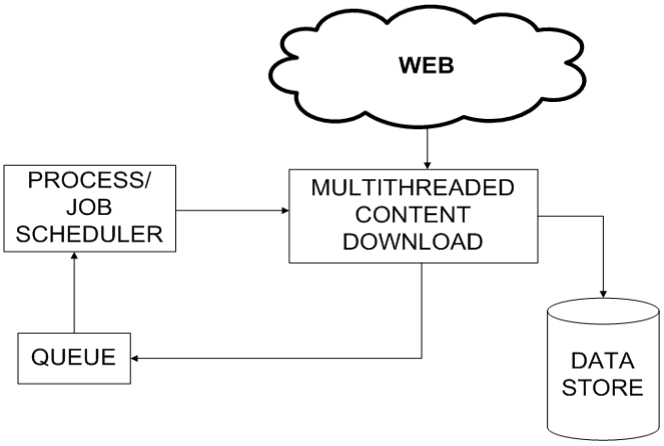

Fig. 2: Architecture of a crawler

2.2 Search index

The sole purpose of a search index is to diminish the time consumption of searches. If the search engine would not have an index, the search would consist of consecutive reading and searching on every document in the target data. Obviously that would take a considerable amount of time. For a target set of thousands of documents the search on an index would be completed within milliseconds, while the sequential reading of every document would be completed in no less than a couple of hours.

The crucial factors in index designs are:

- the way content is added to the index (multiple crawlers are supported or not),
- the physical storage technique (large scale engine have their own file system such as Google's Big Tables),
- expected size of the index (whether the content that is crawled is global or not),
- search time,
- the ratio between search time and time needed to add content to the index,
- the maintenance of the index for an extended time period and
- the error handling capabilities.

The most common data structures that are used to store the index are: suffix trees, trees, inverse index (an example is shown in Table 1), forward index (an example is shown in Table 2), citation indexes, document-term matrices and many others.

Table 1: Example of an inverted index

Word	Documents
name	Document 1
informatics	Document 3, Document 5
phenomena	Document 1, Document 4
in	Document 1, Document 2, Document 3

Table 2: Example of a forward index

Document	Words
Document 1	name, phenomena, in
Document 2	in
Document 3	informatics, in
Document 4	phenomena
Document 5	informatics

2.3 Search

Search is conducted when a user submits the search string. The search string is a formalization of a user's need for information. It is typical that the search strings are not structured and can have ambiguous interpretations and meanings.

Three most common search query types are: information query (thousands of possible results), navigation query (the user is trying to navigate to a known site, such as www.wseas.us) and a transaction query (where the search query is used to conduct a transaction, such as the purchase of an automobile. A crucial factor for understanding search queries is the experience of the user [4].

The most important factors on search are:

- queries (number of search terms, use of logic and modifiers),
- sessions (types of queries in a session, number of pages visited),
- terms (rank/frequency) distribution and
- most common search terms) [5].

Most commercial large scale search engines do not share the data from their logs.

But a study [6], that analyzed the Excite engine revealed these characteristics of web search:

- average query length is 2,4 words,
- almost 50% of users examines only the first and second results page (10 results per page),
- less than 5% of users use advanced search features,
- 19% of the queries contained a geographical term.

2.4 Search in the semantic web

Semantic web and semantic technologies in general provide a new way of managing data, which is based on the creating of semantic meta-data. The meta-data is used in two different levels. In the first the data describes a document (web page) or a section of a document (paragraph, table). In the second it describes the entities within the document (persons, organizations). Regardless of the level it is important that the meta-data provides an additional description of the meaning of a document or entity. The description actually provides information on the content of the document (topic or relations to other documents) or entities. In most pages on the web today the metadata is encoded within the HTML and they only provide information on the representation format (design). With HTML the content can be formatted, but there is no mechanism to tag a string with additional information (price of a product, or a name of an author) on its meaning.

If the semantic metadata is available for hypertext documents an array of exciting services becomes available:

- Information retrieval that is based on the meaning. This includes automated resolving of the problem of synonyms, antonyms and context based meaning of ambiguous words.
- Improved presentation of information retrieved, for instance the results are separated into groups with regard of their target domain.
- Information exchange between different organizations.

We see the foremost advantage of the semantic web in the elimination of deficiencies in the current generation of search engines. The most important issue is the ambiguity of search strings. Current search engines cannon determine the correct meaning of the search string although the ambiguity can be resolved to some extent with additional keywords, the majority of users do not use them [6]. Ambiguity is a problem because search engines are basically executing an index scan. The words in the search string can be different from those in the index and still have the same meaning (synonyms). Current search engines have no means to determine semantic relations between concepts. For instance if we take into consideration the following search string:

"carrier Europe John Smith management".

The user is looking for information on a manager of a carrier (telecommunications company) in Europe, whose name is John Smith. The search engine would not retrieve (or would retrieve and rank it as not very important in relevance to the query) a document titled "The new CEO of Vodafone in the UK is John Smith".

In order to retrieve the document and rank it accordingly the search engine would have to be aware of the following semantic relations:

- Vodafone Live is a telecommunications company in other words a **carrier**.
- UK is an abbreviation for United Kingdom which is a country in **Europe**.
- CEO stands for "chief executive officer", one of the highest-ranking corporate officers (executives) or administrators in charge of **management**.

2.5 Personalization

Personalization is an iterative process [7]. The first step consists of gathering user data and classification of the provided content. In the next (2) step the user profiles are created. Then the content (in our case search results) is customized to the user profiles and an evaluation process follows to determine the success rate of the personalization. If the rate is too low it is an indicator to review the methods and approaches that were selected in steps 1 and 2. Fig. 3 presents the personalization process as viewed from the perspective of a search engine.

In order to have a successful personalization solution we have to gather as much data on the users as possible. The data can be gathered implicitly (from his/hers activity) or explicitly where the user expresses his own interests. Implicitly gathered data most commonly involves: users IP address, timestamp, HTTP requests, data returned, browser information and the referral. The user can be identified with one of these methods:

- IP address,
- URI (the server automatically adds strings to the URI that identify the current user and session),
- hidden fields (parts of the web page that are invisible are used to store identification data),
- HTPP authentication (authentication procedure included in the HTTP protocol) or
- cookies (stored on the user's computer).

With explicit data gathering it is up to the users to state their needs, interest areas or properties. Two most common approaches are the use of questionnaires and evaluation procedures.

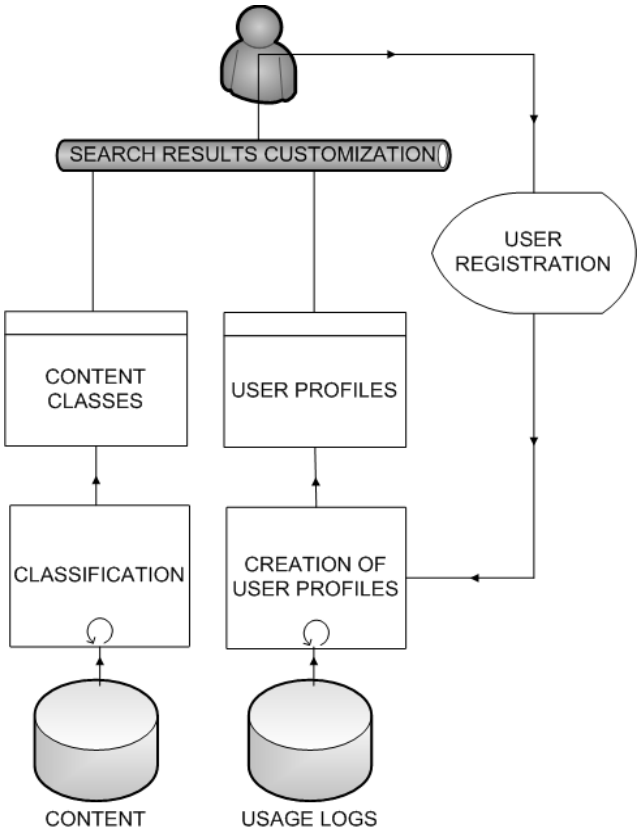

Fig. 3: Personalization process

The content profiles (Fig. 3) are used to classify the content according to the interest areas of users. The profiles are limited to typical properties. Since the content is mostly textual data, the typical properties are expressed in the form of words. The content classes are then weighted differently for each user, expressing his interest in a particular content class (topic).
Ontology is a formal representation of concepts and relations between them. It can be used to represent knowledge on a particular domain. The building blocks of ontology are classes, attributes and relations. The most important relation type is the hierarchy type. It is represented with relations such as super class (parent) and subclass (child). Ontological profiles require domain ontology. The content is then classified against the ontology classes (represented by word vectors). The user profile is extended with the entire domain ontology and we store the observed interests of a user for individual classes.

Usage profiles are the opposite of content profiles. They are used to store links to the content that was interesting to a particular user. The links can be entered explicitly by the user or implicitly with an analysis of user activity data. The value of usage profiles is the ability to predict which new (previously unknown for the user) content would be of interest to him.

Search result customization can be done with search result grouping, which is a classical technique, more generally known as content classification. The results are grouped according to their mutual similarity. A typical approach is to use the Euclidean distance defined as follows (1):

$$E = \sqrt{\sum_{i=1}^{N}(u(i) - d(i))^2} \qquad (1)$$

In the equation N stands for the number of elements in a vector, u is the classification vector of the first element (search result) and d is the classification vector of the second element (search result).

The main methods of personalization (for a particular web site) in general are:

- "Pull".

  Personalization is activated only on user demand, while he is actively using the web site.
- "Push".

  Personalization when the user is not browsing. It is based on notifications to the user about changes or new content on the site.
- "Passive".

  The personalization is autonomous and is actively used with normal usage of the target site. This is the method of choice for personalization of search engines since it incorporates content classification during browsing and searching.

The personalization evaluation provides the information on the quality of our approach. It can have a decisive influence on the decisions regarding the entire personalization process (outlined in table 3). The evaluation is based on precision measurements. It evaluates the distance between the assessments of user's interest and his actual interests. The three main metrics are: forecasting precision, classification precision and ranking precision.
Forecasting precision metrics compare the degree of interest (forecast) generated by the system with the actual interest degree of the user.
Usually the precision is calculated as the mean absolute error (2):

$$MAE = \frac{\sum_{i=1}^{N} |p_i - r_i|}{N} \qquad (2)$$

In the equation N stands for the number of all elements, p_i is the system generated degree of interest and r_i is the actual users' interest.

3 Overview of the presented system

The system we developed has many of the components basically the same as any of the existing search engines. Unfortunately this cannot be avoided since the development of a customized search engine brings with itself the heavy burden of having to develop and test every component of the search engine not just the ones we wish to upgrade or modify. As we did in section 2 for the general search engine architecture we will examine major components of our system in the next subsections. We will focus only on the major differences between the standard implementation and the one we have developed.

3.1 Specific features

We felt that there is a need to develop a custom search engine intended to provide its users with unique personalization capabilities and result ranking features. The main initiative for this is because the leading search engines cannot provide those capabilities. They are globally oriented and use ranking factors and functions that are intended to work on a global scale. Their crawlers cannot access content beyond the publicly indexable web barrier. The presented search engine is oriented locally although it does index global content. The content indexed is limited only by its inclusion in the search domain, not geographical or language based considerations. The target domain is everything that is connected with the faculty including the courses taught. Therefore for example some parts of the well known W3CSchools site are indexed because web development is taught on multiple courses. The crawler is composed of multiple applications that run concurrently, so it is an example of a parallel crawler. The indexed content is tagged with part-of-speech (POS) [8] tags. Also an ensemble [9] of specialized taggers is used to tag the meaning of words in both the general and domain specific sense. As a main feature the personalization component features a time component and groups of interest areas. In the following subsection we present the individual components of the system in greater detail.

3.2 Data gathering

Any major search engine has only one way to access the documents, web pages or anything else it indexes. The crawler crawls through the public available content and transfers that data to the indexing server.

In our system there are multiple possible sources of data and therefore we use a number of different approaches to collect it. Fig. 4 shows the data sources that the crawler can index: web content, structured data, unstructured data and data from various applications and databases.

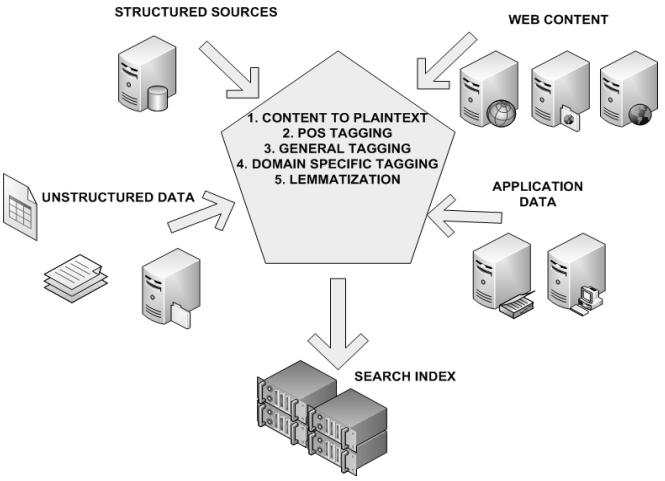

Fig. 4: Crawler architecture

A specialized component is dedicated to the complex task of scheduling the indexing jobs. The indexing intervals are source dependent and occur very often for some sources. The interval is based on the probability of observed content change described in [10]. The metric is calculated with the following equation:

$$Probability\ (content\ changed) = 1 - e^{-RT} \qquad (3)$$

In the equation R stands for rate of change of every data source and T stands for the time span for which the probability is being calculated.

Each content type has a dedicated indexer and the individual indexers run concurrently.

Web content is indexed with a traditional web spider although it is indexing only targeted sites. The spider can be described with the following attributes:

- it only follows links to sites that can be classified as domain relevant,
- it uses breadth-first technique (every site on a level is examined before descending a level),
- crawling is limited to textual data (no video or audio data is included) and
- it crawls on some sites that are not publicly accessible (those that provided us with credentials).

As we have said the system is intended to provide a domain specific information retrieval system on a potentially global scale. So the indexed content is distributed across the web. We use a technique similar to the focused crawling approach [11]. We use global search engines (Google) to provide the list of sites that are potentially interested to the users of our system. The sites are retrieved with executing consecutive searches with search strings from the formal description (ontology of domain concepts) of the target domain. The sites are then stored in a list of potential sites to crawl. The crawlers then visit each site and determine if the content

is appropriate for our domain. Each link is reevaluated so that only certain parts of large web portals are included. For instance we are crawling the W3Schools site but only the sections that are taught in a course at the faculty. Fig. 5 shows a crawler on a hypothetical site. The dots represent web pages; hyperlinks between them are represented with arrows. The dark dots (pages) are those classified to be of interest to the domain users while the light ones represent pages that are not in the domain area. The areas that are classified as uninteresting are not crawled (beyond the initial download for the classification procedure) and their links are not followed.

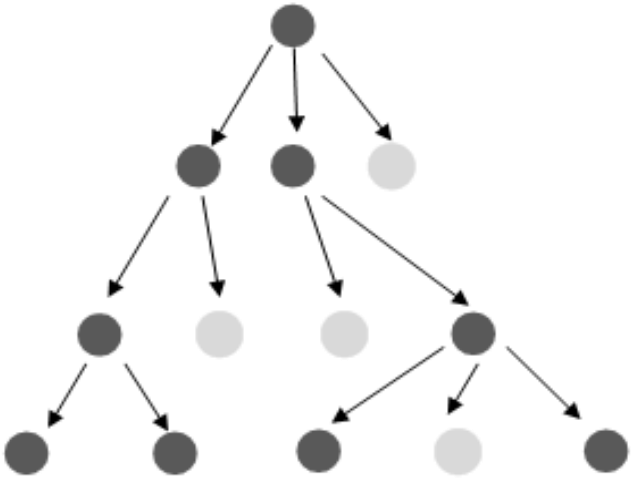

Fig. 5: Domain specific crawling

The approach is similar to [12] in the way that we are also forming a weighted directed graph although we are using it only to determine the content to crawl (index) and not during the actual search process.
The spider runs with multiple simultaneous connections. Every connection is opened to a different web site. This is to assure that the indexer does not overload the targeted sites. The content indexed is the HTML content of the sites as well as various attachments (mostly documents). The crawler respects the robots.txt standard for robots exclusion [13].
The database indexing component operates in two possible modes: (i) fully automated indexing or (ii) indexing according to a provided schema for a particular database. The fully automated indexing relies on the schema of the database. In a relational database the schema describes the data structure and their relations and is an implementation of the conceptual model (ER). We take full use of the fact that the schema gives a detailed description of domain concepts. For the automated indexing the following elements of the schema are important: table relations, attribute data types, attribute constraints (unique) and of course the primary and foreign table keys.
Structured data indexer automatically tags content according to the structure of the document. The database, structured and unstructured data indexers all work in their respective tasks on the local server therefore they share a common component that provides data transfer capabilities. The data transfer is done via web service that is compliant with the MTOM mechanism [14].
All content that is not in plain-text form is transformed to it. The transformation is done with a number of specialized plugins that are compliant with the iFilter interface standard. The iFilter interface is used by the Windows Indexing Service and the newer Windows Desktop Search to index various file formats. We have implemented this component in order to make the task of supporting new file types a simple one. The architecture of the component allows registering of a new iFilter plugin during normal system runtime, therefore any new content type can be indexed without the need to stop normal operations.

3.3 Tagging of content

The tagging is in two phases. In the first the POS (part-of-speech) tag is assigned to every word. In the second an ensemble of specialized taggers is used for a more detailed tagging focusing on the meaning of particular words rather than POS.
The POS tagging is done with an optimized variation of the Hidden Markov Model Approach (HMM) [15]. The HMM is a statistical parser. Statistical parsers work by assigning probabilities to possible parses of a sentence locating the most probable parse. We have implemented a variation that uses string matching techniques to determine which of the sentences in the learning corpora are similar to the one being tagged. The technique works in three steps. In step one we tag known words (with the use of dictionary and corpora). Then the sentence is tagged with a pattern. The pattern is a single string (word) in which every character represents the part-of-speech of a single word. The words with unknown POS tags are represented by a question mark. For an example the pattern “NVN?” represents a sentence where the first word is a Noun (N), the second one is a Verb (V), the third again a Noun (N) and the POS of the fourth is unknown (?). By using full-text search operators in step two similar sentences are selected from the learning set (examples of grammatically correct sentences). Then in step three the most probable POS tags are assigned to the unknown words. As an additional factor in the probability calculation the ensemble of specialized taggers is used (the same one used in the second phase of content tagging). An example of this is the use of a name tagger that specializes in the recognition of person names. If the name tagger tags an unknown word as a

person name then obviously the POS of that word is a Noun. The degree of influence of the taggers on the probabilities depends on the reliability of a particular tagger. The taggers are also using a specialized component for automated declination of proper nouns [16].

The ensemble uses the same basic idea of operation as the ensemble data mining method of boosting. The method is simply explained with the following example. Suppose you as a patient have certain symptoms. Instead of consulting one doctor, you choose to consult several. Suppose you assign weights to the value or worth of each doctor's diagnosis, based on the accuracies of previous diagnoses they have made. The final diagnosis is then a combination of the weighted diagnoses. This is the essence behind boosting. We have thought at our ensemble in a similar way with the difference that the doctors from the example are each a specialist for a particular area. Their worth is based both on accuracies of previous tagging and the classification of the tagger.

The specialized ensemble of taggers consist of four classes of taggers: (I) tagger that uses a database of examples (II) tagger that generates examples according to some logic (III) tagger that uses regular expressions and (IV) a tagger that uses a combination of taggers from other classes. Some of the taggers implemented include: a name tagger (class I), abbreviation tagger (class I), tagger based on generated values (class II), web and email address tagger (class III) and others.

3.4 Personalization

We will provide an examination of the personalization features by viewing the most important aspects; context, interest areas and content classification. We should note that as with any personalization there is a delicate question of handling personal data (that is data by which the user could be identified) [17]. Before any work can progress some important decisions are to be made. They are depicted in table 3.

We have decided to avoid user interaction as much as possible. Instead we have tried to gather information needed from various sources automatically. This of course still requires a written permission from the user to start collection information about him, since some of the data may not be publicly available or is of classified nature. It was decided that user profiles would be of a user type.

The nature of our environment makes the use of group profiles unnecessary and impractical. Also a well known fact is that individual (user) profiles are used when the user can be positively identified, usually with registration and login procedures. The personalization is focused on search results (content) and it will include some personalization of result visualization, such as grouping results by their respective topics. In this sense the approach selected is obviously a passive one.

Besides these considerations there is also a large question on determining the user intention. Without knowing the actual intent of a query it is impossible to provide relevant results. Since most of the queries are short, they are ambiguous. To solve this problem, there have been many different approaches. We believe that Relevance Feedback [18] is the main method for improving the accuracy for a particular user.

Table 3: Table of decisions with personalization of search results, blank fields represent decisions to be made

Data gathering method	User profile type	Personalization involves		
		Content	Visualization	Structure
Implicit	User			
	Group			
Explicit	User			
	Group			

Our approach is a combination of personalization based on ontology [19] and ensemble of content classifiers with an inferring component. The personalization component provides services that enable it to use data from other application, web services and data stores in various formats. Every data provider is a member of an ensemble. An inferring component (»conductor«) has the task of selecting appropriate ensemble members for a particular query. Every component is associated with semantic information that allows the automata to infer where, when and how it should be used. The conductor uses the ensemble member's semantic description with the data available for the query to select members that are relevant to the query. The relevant members are then called upon to retrieve the data at their disposal and submit it to the inferring component. The inferring component then provides a ranked list of relevant key words that the search engine uses to personalize the ranking function. Fig. 6 gives an overview of the process. We can describe this process as being a combination of implicit and explicit data gathering. It is implicit because there is no user interaction required. And it is explicit because the data is gathered from reliable sources (faculty classes schedule, domain login information…) about a particular user. The various formal sources are used as a substitute for user input. This can be done because the data they contain is regularly maintained and supervised. This process eliminates user interaction while maintaining a high degree of data reliability.

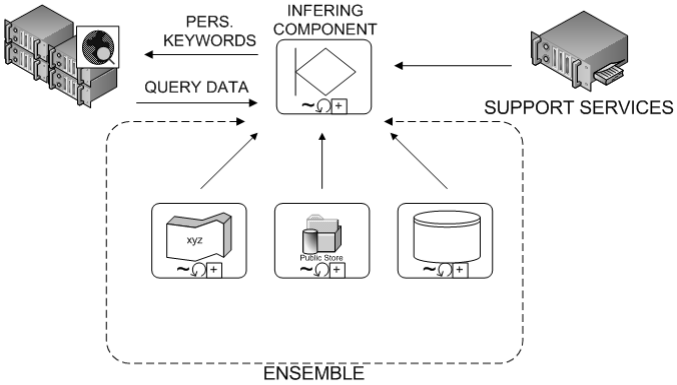

Fig. 6: Personalization process

Fig. 7 gives a simple overview of the user profile. It is essentially consistent of static properties connected with content and usage data gathered during system usage, a classic approach.

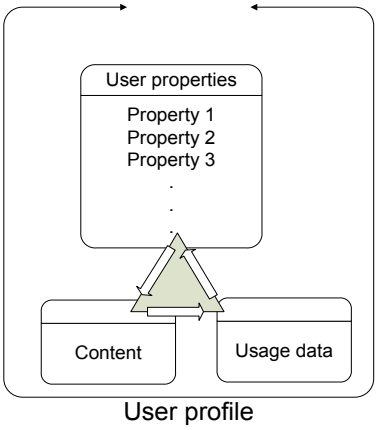

Fig. 7: User profile

3.4.1 Context

The personalization is based on the search history and the information gathered from the user profile. Also various other sources are incorporated.

Again we turn to our academic environment for usage examples. The information for staff can be automatically gathered from their bibliography and current research projects. For students the curriculum and course timetables can be used to determine the context of a given search. For instance the student »John Doe« logs in a computer terminal at 9:00. At 9:15 he starts a search on the faculty web page. From the IP address and the timestamp the personalization system can easily infer who the user (logon information from the domain server provided by the supporting services) is and what course he is taking currently (course timetable). This automatically determines the search context [20] (by using the key words from the course descriptions) and the search can use the key words from the course description as the context guidelines.

3.4.2 Interest areas

When the current context cannot be determined with a high enough confidence the personalization is based on the users interest areas. The interest areas are a generalization of the current context. For instance if a user is looking for information on loops in the C++ programming language, this would be generalized as the interest area of »programming languages«. This generalization is possible because of the domain specific ontology that was created in order to be able to make this type of generalization procedure. The ontology consists of semantic information and relationships on every course taught at the faculty, general information on the faculty, its staff and students. Every course has a few keywords associated with every single topic it covers. The topics are linked to various research areas. For instance the hypothetic course »Introduction to algorithms« that lectures on »Statistical taggers« would be linked to the »natural language processing« research area.

3.4.3 Content classification

The first step in content classification is the transformation to plain-text. The classes in which the content is classified are determined by the domain specific ontology. The main properties of documents are obtained with the use of the TFIDF (Term Frequency Inverse Document Frequency) metric.

The metric is defined as follows:

$$d^{(i)} = TF(W_i, d)IDF(W_i) \quad (4).$$

The IDF is defined: $IDF(W_i) = \log\frac{D}{DF(W_i)}$ and D is the number of documents, DF(W) is the number of documents in which the word (W) occurs at least once and TF(W, d) is the number of word W occurrences in the document d. Additionally the metric can be normalized so that the TFIDF of individual words is divided by the square root of the sum of all TFIDF word frequencies as follows:

$$nTFIDF = \frac{TFIDF_{i,j}}{\sqrt{\sum_i TFIDF_{i,j}^2}} \quad (5)$$

Documents are assigned to their respective classes with the comparison of the class description from the ontology and the main features of the document. For documents that cannot be automatically classified with this method we use the cosine similarity. The cosine similarity is used to find *k* documents that are most similar to the new one. The classes attached to those

documents are then used for the new one. Cosine similarity between two documents (d_i and d_j) is defined as follows (6):

$$\cos(d_i, d_j) = \frac{\sum_k d_{ik} d_{jk}}{\sqrt{\sum_l d_{il}^2 \sum_m d_{jm}^2}} \quad (6).$$

4 Conclusion

The presented system provides features that cannot be provided by global search engines. The indexed content is tagged with an optimized version of the well-known HMM model that uses patterns in order to properly tag words on the fly, with almost no data preparation procedure necessary. As a key feature it uses domain specific ontology and an ensemble of specialized taggers to tag the indexed content with the meaning the words have in the target domain. The specialized ensemble of taggers is designed in such a way that they acquire their data autonomously from their respective sources. This is important because it eliminates the need for manual work and oversight.

The domain specific search system we have presented features true personalization and custom ranking to provide the user with effortless information retrieval capabilities. However, as we have come to realize, users expect consistency in their search results. Since the results are personalized they are not consistent over time. The search results of a personalized search engine are inherently dependent on the profile of the user. The better the profile the more the results deviate from the standard (result set without personalization included) result set. It is very important to provide the user with the ability to influence the use of personalization (or to choose not to use it at all). This provides the features of the engine to those users that are willing to use them or find them useful. For future work we will focus on allowing the user to create his custom domain. The definition of the domain will be a group of search sources (web pages, structured, unstructured data and documents) that the user will select either manually or on the recommendation of the search system. Basically this will result in personal search "agents" that are fully personalized because they are essentially "built" by the users themselves. The users will be able to specify which content to index, provide custom ranking of sites and use search results bookmarks that will be stored for them on the search server, making them accessible anywhere. Also we have a need for a better, more automated evaluation procedure of the user satisfaction with the system. Additional metrics will have to be employed with the final goal of a full automated system for personalization features improvement.

For the improved performance we are considering the research on query term frequency distributions. It is suggested that they conform to the power law, or long tail distribution curves. This is expressed in a way that a small number of unique queries are used most often. Traditionally this fact is used for various optimization techniques. We are still researching if the power law is valid in our domain specific variation of a search engine.

References:

[1] S. Lawrence, L. C. Giles, Searching the World Wide Web, *Science*, Vol. 280, No. 5360, 1998, pp. 98-100.

[2] S. Brin, L. Page, The Anatomy of a Large-Scale Hypertextual Web Search Engine, *Proceedings of the Seventh World-Wide Web Conference*, 1998.

[3] A. Heydon, M. Najork, Mercator: A scalable, extensible web crawler, *Proceedings of the Eight Word Wide Web Conference*, 1999, pp. 219–229.

[4] I. Hsieh-Yee, Effects of search experience and subject knowledge on the search tactics of novice and experienced searchers, *Journal of the American Society for Information Science*, Vol. 44, No. 3, 1993, pp. 161-174.

[5] B. J. Jansen, A. Spink, J. Bateman, T. Saracevic, Real life information retrieval: a study of user queries on the Web, *ACM SIGIR Forum*, Vol. 32, No. 1, 1998, pp. 5-17.

[6] A. Spink, D. Wolfram, B. J. Jansen, T. Saracevic Searching the web: The public and their queries, *Journal of the American Society for Information Science and Technology*, Vol. 52, Issue 3, pp. 226–234.

[7] G. Adomavicius, A. Tuzhilin, Personalization technologies: a process-oriented perspective, *Communications of the ACM*, Vol. 48, No. 10, 2005, pp. 83-90.

[8] C. D. Manning, H. Schuetze, *Foundations of Statistical Natural Language Processing*, MIT Press, 1999.

[9] T. G. Dietterich, Ensemble Methods in Machine Learning, *Proceedings of the First International Workshop on Multiple Classifier Systems*, 2000, pp.1-15.

[10] B. E. Brewington, G. Cybenko, How Dynamic is the Web?, *Proceedings of the Ninth International World Wide Web Conference* (WWW9), 2000.

[11] S. Chakrabarti, M. van den Berg, B. Dom, Focused crawling, A new approach to topic-specific web resource discovery, *The 8th International World Wide Web Conference*, 1999.

[12] D. Yadav, A. K. Sharma, J. P. Gupta, Topical Web Crawling Using Weighted Anchor Text and Web Page Change Detection Techniques, *WSEAS*

Transactions on Information Science & Applications, Vol. 6, 2009, pp. 263-275.

[13] M. Koster, A Standard for Robots Exclusion, http://www.robotstxt.org/wc/robots.html, 1994.

[14] W3C, SOAP Message Transmission Optimization Mechanism, http://www.w3.org/TR/soap12-mtom/, 2005.

[15] E. Charniak, Statistical techniques for natural language parsing. *AI Magazine*, Vol. 18, No. 4, 1997.

[16] I. Čeh, M. Ojsteršek, Slovene language question answering system, *Recent advances in computers : proceedings of the 13th WSEAS international conference on computers (part of the 13th WSEAS CSCC multiconference)*, 2009, pp. 502-508.

[17] K. Heikkinen, J. Eerola, P. Jappinen, J. Porras, Personal view of personal information, *WSEAS Transactions on Communications*, Volume 2, 2004, pp. 50-55.

[18] C. Buckley, G. Salton, J. Allan, The effect of adding relevance information in a relevance feedback environment, *Proceedings of the 17th Annual international ACM SIGIR Conference on Research and Development in information Retrieval*, 1994.

[19] M. Adda, W. Lei, Y. Feng, Personalization Based on Domain Ontology, *WSEAS Transactions on Computers*, Issue 3, Vol. 5, 2006, pp. 598-603.

[20] Steve Lawrence, Context in Web Search, *IEEE Data Engineering Bulletin*, Vol. 23, No. 3, 2000.