Information Retrieval from the Web: an Interactive Paradigm

Massimiliano Albanese, Pasquale Capasso,
Antonio Picariello, and Antonio Maria Rinaldi

Università di Napoli "Federico II", Napoli, Italy,
{malbanes,pacapass,picus,amrinald}@unina.it

Abstract. Information retrieval is moving beyond the stage where users simply type one or more keywords and retrieve a ranked list of documents. In such a scenario users have to go through the returned documents in order to find what they are actually looking for. More often they would like to get targeted answers to their queries without extraneous information, even if their requirements are not well specified. In this paper we propose an approach for designing a web retrieval system able to find the desired information through several interactions with the users. The proposed approach allows to overcome the problems deriving from ambiguous or too vague queries, using semantic search and topic detection techniques. The results of the very first experiments on a prototype system are also reported.

1 Introduction

The challenging task of a web information retrieval system is that of retrieving from the web all the *relevant* documents to a user query: this operation is considered successful if it returns as few as possible of non-relevant documents. The typical steps thus performed by a user are described as in the following: i) a user submits the query – hopefully written in natural language – to the search engine; ii) the system returns a ranked list of documents; iii) the user reads each document until she finds enough relevant information. From a general point of view, the returned list may be viewed as the *engine's model* of the user's idea of what is considered relevant: in a certain way, we can say that the engine considers the first document as the most relevant one, and so on. The main problem that arises in traditional systems is that in most cases user's interests are either poorly defined or inherently broad. The answer to a typical query is thus a very large set of documents, among which the user has to look for what she is actually interested in. In these cases, we can easily identify another usual user behavior: if, at a certain point, she starts to find that most of the retrieved documents are not relevant, she gives up that returned list and tries to refine the query, adding more specific details. Our vision is that of extending this behavior, in order to give the system the capability of *understanding* if the user query is too much general, thus automatically *trying to refine* the queries and *asking the users* more specific information. In other words, we explicitly accept the user's

feedback in order to adjust the *engine model*. Unfortunately, characterization of the user information need is not a trivial task, and, at the same time, the users want targeted answers to their queries without extraneous information.

Imagine, for example, a user who is looking for a new car. She is interested in the information about different cars, with a primary focus on their prices. Our user turns to a web search engine and types in the simple query $q =$ "car": we surely agree that q is a inherently vague query. In our vision, a great advance in the retrieval process may be obtained if the system could recognize that the required information is poorly specified, thus posing some questions to the user in order to narrow the search. In the above example the user may be explicitly asked to clarify which sense of the word "car" she means, since that word can have multiple meanings (see example 1). Once the user has selected a meaning, then the system may decide if the information for solving the query are sufficient or not. If it is the case, then a set of semantically relevant documents is returned to the user and other questions may be asked, on the basis of the major identified topics, such as "car rental", "car parts", "car prices", and so on, asking the user to specify what she is looking for.

In this paper we propose an approach for designing a web information retrieval system, able to retrieve the desired information through several interactions with the users. The proposed approach allows to overcome the problems deriving from ambiguous or too vague queries, using semantic search and topic detection techniques.

The paper is organized as follows. Next section 2 discusses related works. Section 3 introduces the architecture of the system while section 4 briefly describes the overall retrieval process. Sections 4.1, 4.2 and 4.3 respectively describe in details the three main steps of the process, namely the *user driven keyword extraction*, the *semantic search* and the *topic based query refinement*. First experimental results are reported and discussed in section 5, while conclusions are given in section 6.

2 Related Works

The research community has been devoting huge efforts in the field of information retrieval in the last two decades. Investigations range from traditional document retrieval systems to modern question answering systems, from user relevance feedback to document clustering and classification.

Traditional information retrieval systems [2] usually retrieve and rank documents containing the keywords in the query, irrespective of the context in which those keywords are used. Some attempts have been done to take context into account and retrieve documents that are semantically rather than only syntactically related to the user's query [9, 18, 25].

In the latest years, the IR community has focused its attention on question answering (QA) systems, whose goal is that of identifying and presenting to the user a concise and precise answer to a question, rather than identifying documents that may be somehow related to the query. Such systems are designed

to extract answers from structured knowledge bases or directly from the web. Several question answering systems rely on a variety of linguistic resources – dictionaries, part of speech tagging, syntactic parsing, semantic relations, named entity recognition, etc. [1, 5, 21] – while many other simpler solutions rely on pattern matching and the inherent redundancy of large text repositories to facilitate answer mining [7, 22]. Due to the extreme simplicity of the latter approaches, they are usually more efficient, even if sometimes they may be less reliable.

In most recent works, great importance is also given to the so called 'relevance feedback'. This term is referred to the practice of adapting the retrieval behavior of a search engine, based on some information gathered from the users themselves. The collected information can be used to dynamically refine the current query and speed up future searches. Several ways exist to get feedback from the users. The most common ones rely on asking the user to judge the relevance of each retrieved document w.r.t. her query [23], while many others are based on implicitly inferring users' interests by analyzing browsing logs. Good examples of the latter approach are QueryFind [29] and Takealook [27]: user's interests are captured and document presentation is personalized accordingly.

Independently from the approach, the goal of any feedback strategy is that of overcoming the information overload caused by the great number of documents returned by current web search engines. Search engines themselves commonly provide a simple mechanism to address this issue, ranking documents according to some relevance measure. In order to take into account user requirements, several authors propose document clustering techniques, grouping together similar or related documents into classes. The operation of clustering documents could usually be of two types: global and local. In a global clustering strategy, the entire document collection is clustered off-line and queries are compared with a representation of each cluster instead of a representation of each document. In a local clustering strategy, clustering is applied to the documents that are returned in response to a query (e.g., Carey et al. [4], Wu et al. [30], Zamir and Etzioni [31]). This approach is successfully applied to the web by some search engines, as Northern Light or Vivísimo, to the purpose of making result set browsing easier. Nevertheless, recent works [10, 30] about clustering evaluation involving users show no significant differences in effectiveness between interfaces using classic ranked list and post-retrieval clustering. Among others, authors suggested two main reasons: first, the information accompanying clusters (usually a list of keywords) frequently is not enough to specify the content of the documents in a cluster; second, although the organization in clusters helps users to discover groups with a high density of relevant documents, clustering interface does not give further assistance in identifying particular documents with relevant information. Some authors (Mana-Lopez et al. [16]) propose to add to post-retrieval clustering algorithms, some techniques of multi-document summarization, in order to better show the topic related to each cluster and those related to the different documents in it. Kummamuru et al. [13] propose a hierarchical document clustering algorithm for summarizing and browsing search results, based on automatic taxonomy generation.

Despite the great amount of work done, we can state that, at the best of our knowledge, no system is currently able to retrieve only the really relevant documents for each user query. That is why we propose a system that produces a very small answer set for each query, through repeated interactions with the user. The main goal of the user interaction is that of collecting enough information to manage ambiguous or too vague queries. In particular, we apply some kind of clustering to larger answer sets.

It's worth noting that we do not refer to the term *clustering*, like many other authors do, as "grouping data elements such that the intra-group similarities are high and the inter-group similarities are low" [20]. Instead, we agree with Estivill-Castro [8], who more generally defines *clustering* as a "hypothesis to suggest groupings in the data", thus considering the possibility of overlapping document sets. In fact, as described in details in section 4.3, we identify major topics in a document collection and group documents based on such topics.

3 System Architecture

Figure 1 shows at a glance the overall architecture of the system. From a structural point of view, the system acts as a meta search engine, relying on existing general purpose search engines.

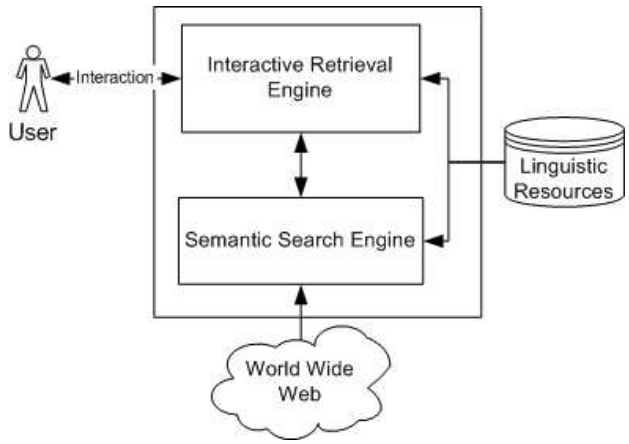


Fig. 1. System architecture

Users submit their queries – sets of keywords or natural language questions – to the *Interactive Retrieval Engine*, whose role is that of analyzing queries, asking appropriate questions to the users if they exhibit some ambiguities and forwarding the clarified queries to the *Semantic Search Engine*, that returns a set of documents semantically matching the queries. The role of the Interactive

Retrieval Engine also consists in asking further questions to the users in order to narrow too wide answer sets.

Both the Interactive Retrieval Engine and the Semantic Search Engine rely on a set of linguistic resources. In our implementation we adopt WordNet [17], since it offers both a dictionary functionality and semantic network capabilities. The latter one are fundamental for the design of a semantic search engine.

During the entire retrieval process, the Interactive Retrieval Engine keeps track of the answers given by the users and updates an internal cache, in order to simplify and speed up future searches.

4 The Retrieval Process

In this section we briefly introduce all the steps of the interactive retrieval process. Figure 2 shows a data flow diagram of the process.

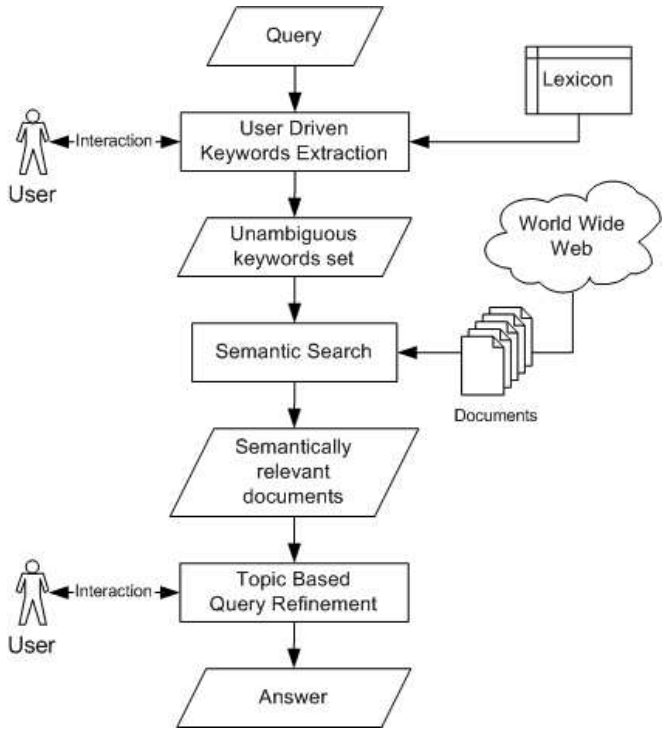


Fig. 2. The retrieval process

First of all a user submits a query to the system expressing her information needs. In its simplest form, a query is composed of keywords, but in our framework we allow more general queries, in the form of natural language phrases or questions (e.g. "The capital of France").

Given the query, a set of keywords must be derived from the query string, as described in section 4.1. This step may require interaction with the user in order to clarify ambiguities.

The unambiguous keyword set built in the previous step is then used in the semantic search phase for retrieving documents that are semantically relevant to the user query, as described in details in section 4.2.

The answer set returned by the Semantic Search Engine may be still too wide to be directly presented to the user for manual inspection and contain non relevant information. Thus the documents in the answer set are further analyzed and main topics are identified, as described in section 4.3. The user is then asked to select which of them she is interested in. At this point the documents covering the selected topic are presented to the user or the latter step is repeated if the the answer set is still too wide.

4.1 User Driven Keyword Extraction

This section describes the steps needed to extract a set of unambiguous keywords from the original user query in a user driven fashion.

As we have previously seen, a query is usually composed of keywords: anyway, we want to allow more general queries, such as natural language sentences. A user who is interested in retrieving information about the capital of France, may just type the sentence "The capital of France", from which the system should identify "capital" and "France" as keywords.

In the simplest cases elimination of stopwords may be enough to achieve the goal. We propose a more general approach based on a preliminary syntactical analysis of the text. To this aim, part of speech tagging is applied to the query text. The tagger is based on a dictionary and a set of heuristic rules that allows to resolve ambiguous part of speech assignment: most words usually have more than a single part of speech, but most ambiguities may be simply resolved (e.g. the word after an article can be a noun but not a verb). In addition to that we also apply *Named Entity Recognition* [3] in order to identify and properly tag the names of entities – people, places, organizations. Based on the part of speech tagging, we can select the keywords by considering named entities and other noun phrases only.

At this point, words that admits more than a single meaning need to be disambiguated. An effective way to address this issue is that of asking the user which sense she intended for each of the ambiguous words. In order to perform this interaction with the user, the system extracts from the dictionary the glosses for the several senses of the ambiguous words and asks the user to select one of them.

Example 1. Consider a query containing the word "car", that is selected as a keyword. Since the word "car", according to WordNet, has five different senses, the user is presented with the following question.

What do you mean by "car"?

1. "4-wheeled motor vehicle; usually propelled by an internal combustion engine"
2. "a wheeled vehicle adapted to the rails of railroad"
3. "a conveyance for passengers or freight on a cable railway"
4. "car suspended from an airship and carrying personnel and cargo and power plant"
5. "where passengers ride up and down"

Similar questions are posed for each ambiguous word in the keyword set.

4.2 Semantic Search

Several techniques have been proposed in the last years [2, 6] in order to overcome the limitations of traditional search engines, that are mainly keyword based, and evolve towards semantic search engines. In this work we use a technique based on *ontologies*. A formal definition of ontology can be found in [11] and [19]. According to this definition an ontology can be seen as a set of "terms" and "relations" among them, denoting the concepts that are used in a specific domain.

In our framework the implementation of the ontology is realized through a semantic network that is dynamically built around a term or a set of terms that are central to a specific context. We call this kind of ontology *Dynamic Semantic Network* (DSN) due to the way it is built. Technically speaking the network is built by extracting a subgraph from the complete graph of WordNet. WordNet organizes words into synsets, that are set of synonyms; words having more than a single meaning appear into as many synsets, each representing a different concept. Given a word and chosen a sense, i.e. the synset that represents the concept which the user is interested in, the DSN is built starting from that synset.

In the construction of the DSN we consider all the terms (synonyms) in the synset. Beyond the synonymy, we consider other semantic and lexical proprieties, namely holonymy, meronymy, hyponymy, hypernymy, coordinate terms and domain terms. WordNet manages all the properties we are interested in for building the DSN.

Figure 3 shows an example of DSN built around the synset corresponding to the first sense of the word "car".

After considering the synonyms, we build a hierarchy, only based on the hyponymy property; its last level corresponds to the last level of WordNet hierarchy. After this first step we enrich our hierarchy considering all the other kinds of relationships in WordNet. Based on these relations we can add other terms obtaining an highly connected semantic network.

The lexical and semantic properties are represented by arcs between the nodes of the DSN and are assigned a weight σ_i, in order to express the strength of the relation. The concept of strength of a relation is presented in [28] and,

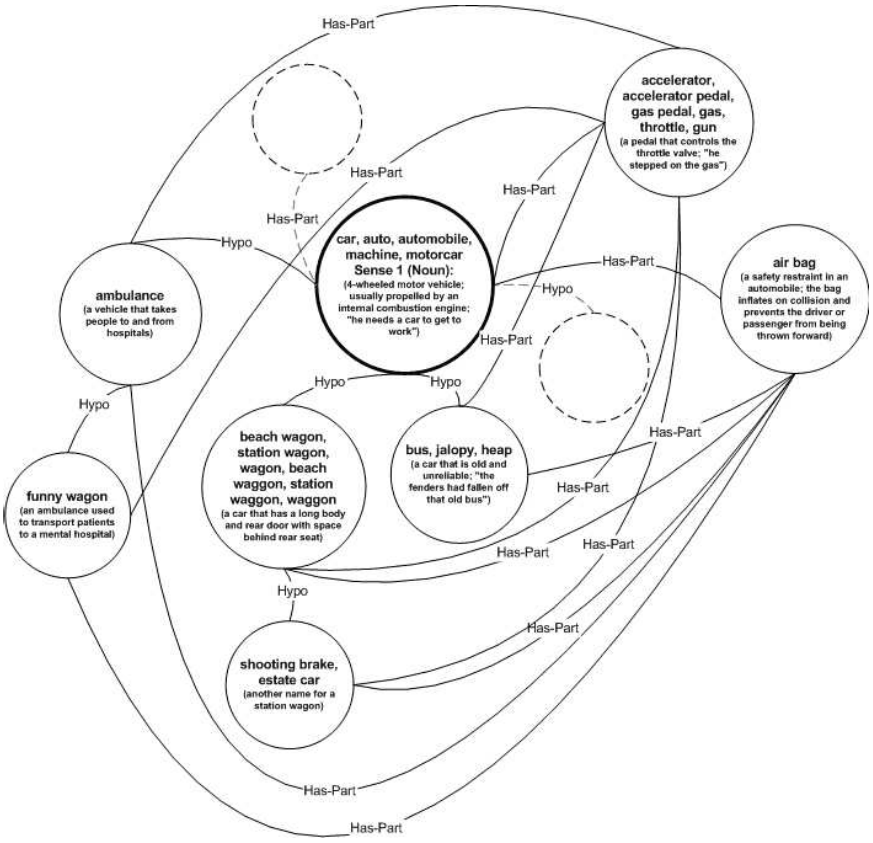


Fig. 3. An example of DSN

following its example, the values of our weights, defined in the [0,1] interval, are set by experiments.

In the worst case, the above described way of building the network leads to a number of completely disconnected subnets equal to the number of keywords in the query. This may be a problem, due the lack of a path between some pairs of nodes. We address this issue by taking advantage of the linguistic proprieties of WordNet. In particular we consider the single subnets and their roots, representing the keywords in the query, and connect them by finding the first common subsumer and adding all the synsets along the path. We thus obtain a totally connected DSN. We remark that this operation is always possible, given the hierarchical structure of WordNet: in fact each synset has the concept *entity* among its ancestors.

Example 2. Let us consider a user interested in renting a car and suppose she types in the query "rent a car". The user driven keyword extraction step selects as keywords the words "rent" and "car" and attaches to them the desired sense.

The system thus builds two DSNs and tries to find some common synset. In the example of figure 4 the module finds as connecting term the word "rental".

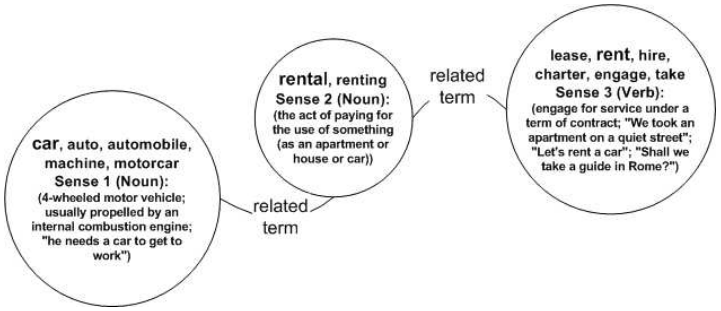


Fig. 4. Connection of two DSNs

At this point we retrieve web pages containing the keywords in the query by means of a traditional search engine and then build a lexical chain [12, 26] on them using the DSN; each word in the page which matches any of the terms in the DSN is a lexical chain component and the references between the components are the relations in the DSN.

Using these words and an appropriate metric described in the following, we can measure the semantic relatedness between words in a given context represented by a DSN.

In order to discriminate interesting pages from the others, we propose a re-ranking strategy that takes into account the measure of similarity among words in a given domain. It is expressed by a combination of the length (**l**) of the path between the terms and the depth of their first subsumer (**d**). The correlation between the terms is the sematic relatedness and it is computed through a nonlinear function. The choice of a nonlinear function to express the semantic relatedness between terms derives from several considerations. The values of path length and depth may range from 0 to infinity, while relatedness between two terms should be expressed as a number in the [0,1] interval.

Furthermore when the path length decrease towards 0, the relatedness should monotonically increase towards 1, while it should monotonically decrease towards 0 when path length goes to infinity. We need a scaling effect w.r.t. the depth, because words in the upper levels of a semantic hierarchic express more general concepts than the words in a lower level. We use a non linear function for scaling down the contribution of subsumers in a upper level and scaling up those in a lower one.

Given the above considerations, we selected an exponential function, that satisfies the constraints previously discussed; our choice is also supported by the studies of Shepard et al. [24], who demonstrated that exponential-decay functions are a universal law in psychological science.

Let us give the following preliminary definition.

Definition 1 (Path Length).
The length l of the path between two terms is defined as follows:

$$l = \min_{j} \sum_{i=1}^{h_j} \frac{1}{\sigma_j} \tag{1}$$

where j spans over all the paths between the two considered terms, h_j is the number of hops in the j-th path and σ_j is the weight assigned to the type of relations in the j-th path. In this way we consider the best path between terms.

We are now in a position to define a semantic relatedness metric extending the one proposed in [14].

Definition 2 (Semantic Relatedness Metric).
The Semantic Relatedness of two terms is defined as follows:

$$W = e^{-\alpha l} \frac{e^{\beta d} - e^{-\beta d}}{e^{\beta d} + e^{\beta d}} \tag{2}$$

where l is the length of the path between the terms, d is the depth of their subsumer, $\alpha \geq 0$ and $\beta > 0$ are two scaling parameters whose values have been defined by experiments.

Example 3. Let us consider two terms X and Y, as in figure 5.

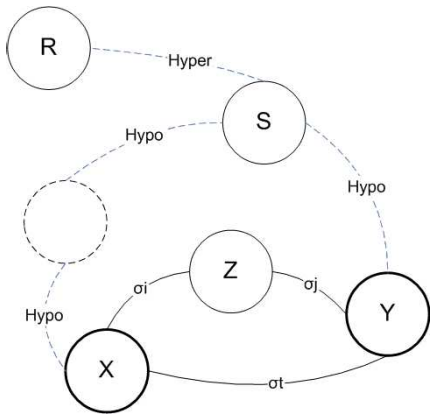


Fig. 5. Best path evaluation

Two paths exist between X and Y. They are labelled with their linguistic properties σ and have a common subsumer S having a distance of 8 levels from the WordNet root. We suppose that $\sigma_i = \sigma_j = 0.8$ and $\sigma_t = 0.3$. In this case the best path is the one traversing Z with a value of $l = 1.58$. We note that according to definition 1, the previous path is not the shortest one. Considering definition 2, the semantic relatedness between the considered terms is thus 0.364.

4.3 Topic Based Query Refinement

The documents returned by the semantic search engine may be too many to be manually inspected, even if semantically related to the query. A possible solution is to partition the answer set into smaller sets by means of some local clustering algorithms. What we propose in this paper is slightly different: given the answer set of the semantic search engine, the idea is to identify the main topics covered in the set of documents, then ask the user which topic she is interested in, and return the documents that cover the selected topic. The process can be repeated several times, until the answer set reaches an affordable size.

First of all, we select candidate topics. To this aim we assume that expressions in the documents such as noun phrases or named entities are appropriate to be considered as topics. We thus adopt a technique similar to that used for extracting keywords from queries.

As an example, topics within documents matching the query "museum" may be "modern art", "science museum" and so on, while topics within documents matching the query "car" may be "car rental", "car parts" or "car prices".

In order to identify candidate topics, retrieved documents undergo part of speech tagging and named entities recognition. Then stopwords, punctuation, verb phrases, adverbs are removed from the text and replaced with a separator. At this point we consider as candidate topics any sequence of words between two consecutive separators.

Example 4. Consider the piece of text *"Founded in 1929 as an educational institution, The Museum of Modern Art is dedicated to being the foremost museum of modern art in the world"*. The described processing produces *"- - - - - educational institution - The Museum of Modern Art - - - - - - museum - modern art - - world"*. "The Museum of Modern Art" is recognized as a named entity and the article "The" is not removed.

Once identified a set of topics, we would like to select and present to the user the ones that best allow to discriminate a small subset of actually relevant documents.

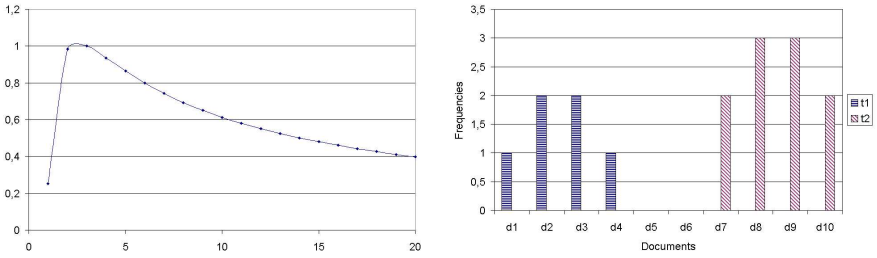
From a preliminary analysis we concluded that three main variables affect the ability of a topic t to effectively group documents: the number of documents containing t, the total number of occurrences of t and the length of t, expressed as the number of words. Let us now explain how to take these variables into account. The greater is the fraction p of documents containing a topic, the worst it can identify small document sets. As regard to the number of occurrences, we observed that the main subjects of a document occur just a few time - say $2-5$ - while higher frequencies denote very common terms. As regard to the length of topics, we observed that most meaningful ones contain 2 up to 4 words. We thus need a function as the one in figure 6.a, in order to take into account both frequency and topic length.

We can thus give the definition of *discriminating power* of a topic.

Definition 3 (Discriminating power). *Given a set $\mathcal{D}$ of documents and a topic t_i, we define the discriminating power Δ of t in $\mathcal{D}$ as*

$$\Delta_{\mathcal{D}}(t) = -\log(p) \cdot \frac{\log(f + \Delta f)}{f^{\alpha}} \cdot \frac{\log(w + \Delta w)}{w^{\beta}} \tag{3}$$

p being the fraction of documents containing t, f the average frequency of t over the documents containing t, w the number of words in t; Δf and Δw are used to shift the curve in figure 6.a in order to prevent Δ to be zero when $f = 1$ or $w = 1$; α and β are used to regulate the slop of the curve.



a b

Fig. 6. a) An example of function used to take into account the effect of frequency and length; b) an example of topic frequency distribution

Example 5. Given two topics t_1 and t_2, let us suppose that their frequency distributions among documents are as in figure 6.b.

Since t_1 and t_2 occurs in the same number of documents, $p_1 = p_2$. However it's also clear that the topic t_2 is more appropriate to discriminate a subset of documents.

The topics are ranked by descending values of Δ and the system asks the user to choose one among the top-k. Documents containing the selected topic are presented to the user or they are further analyzed and the described process is repeated if the size of the answer set is still too large.

5 Experiments and Discussions

A prototype system has been implemented. In particular, the semantic search module extends the one we introduced in [15]. In this section we report and discuss the results of the very first experiments we carried out in order to evaluate the effectiveness of the approach.

Several very vague single-keyword queries have been submitted to the system. Table 1 lists a subset of those queries and reports the number of documents in the web matching them according to Google.

Query	# documents
car	154,000,000
museum	46,200,000
music	283,000,000
photography	45,500,000
soccer	24,900,000
train	39,600,000

Table 1. Examples of very vague queries

The one presented is clearly the worst case: at the beginning, the user does not clearly specify what she's actually looking for and just types in some general terms. The main goal of our first experiments was to verify that the proposed approach allows to reduce the size of the answer set in a very few iterations, keeping the relevant information.

For each query in the test set, users were asked to specify a meaning for the single term in the query. The semantic search engine then considered the first 100 documents returned by Google, selecting among them the ones whose semantic relatedness w.r.t. the query was higher than a given threshold. An average number of 52 pages were considered relevant. Those pages were then analyzed in order to identify the most discriminating topics.

Table 2 shows the top 8 topics identified in the first step of the refinement of the queries $q_1 =$ "*car*" and $q_1 =$ "*museum*" respectively. The topics are ranked for descending values of Δ. The table also reports the number P of documents matching a topic, the total number f of occurrences and the size w of each topic, expressed as the number of words.

Topic	Δ	P	f	w
used car values	0.623	2	4	3
car reviews	0.612	2	4	2
find new cars	0.599	1	2	3
car loan calculator	0.599	1	2	3
premium cars	0.589	1	2	2
midsize cars	0.599	1	2	3
msn autos	0.573	1	3	2
dollar rent a car	0.560	1	2	4

$q_1 =$ "*car*"

Topic	Δ	P	f	w
national museum	0.736	2	7	2
bishop museum	0.695	1	4	2
nobel prize	0.625	1	5	2
asian art museum	0.617	1	3	3
design museum	0.607	1	3	2
american museum	0.571	2	4	2
san francisco museum	0.509	1	2	3
science museum	0.500	1	2	2

$q_2 =$ "*museum*"

Table 2. Identified topics

The results clearly show that the topic based query refinement is able to identify some meaningful and very discriminating topic, since the first iteration: in fact the maximum number of documents related to each of the top 8 topics in the reported example is 2. In order to evaluate if the relevant information is actually returned in the answer set, we asked a group of students to judge

the relevance of each document in the result set w.r.t. the original query plus the selected topic. Around 92% of the documents were considered really relevant while 84% of the answers were considered satisfying.

6 Conclusions

In this paper, we have presented a novel information retrieval system from the web based on an interactive paradigm. In particular, we have extended a classic search engine with some semantic capabilities and query refinements techniques, trying to identify the main topics the users are interested in. We have also described some preliminary experiments using a prototypal system. Further investigation should be devoted first to conduct a more extensive experimentation and then to integrate management of multimedia data into the system.

References

1. S. Abney, M. Collins, and A. Singhal. Answer extraction. In *Proceedings of the 6th Applied Natural Language Processing Conference (ANLP 2000)*, pages 296–301, 2000.
2. R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieaval*. ACM Press, 1999.
3. J. Callan and T. Mitamura. Knowledge-based extraction of named entities. In *Proceedings of the 4th International Conference on Information and Knowledege Management (CIKM'02)*, pages 532–537, November 1998.
4. M. Carey, F. Kriwaczek, and S. Ruger. A visualization interface for document searching and browsing. In *Proceedings of CIKM 2000 Workshop on New Paradigms in Information Visualization and Manipulation*, 2000.
5. J. Chen, A. R. Diekema, M. D. Taffet, N. McCracken, N. E. Ozgencil, O. Yilmazel, and E. D. Liddy. Question answering: CLNP at the TREC-10 question answering track. In *Proceedings of the 10th Text REtrieval Conference (TREC 2001)*, pages 296–301, 2002.
6. H. Chu. *Information Representation and Retrieval in the Digital Age*. Information Today Inc., 2003.
7. S. Dumais, M. Banko, E. Brill, J. Lin, and A. Ng. Web question answering: Is more always better? In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, August 2002.
8. V. Estivill-Castro. Why so many clustering algorithms - a position paper. *SIGKDD Explorations*, 4(1):65–75, 2002.
9. L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, and E. Ruppin. Placing search in context: the concept revisited. In *Proceeding of the Tenth International World Wide Web Conference*, pages 406–414, 2001.
10. M. Fuller, M. Kaszkiel, C. Ng, M. Wu, J. Zobel, D. Kim, J. Robertson, and R. Wilkinson. Ad hoc, speech and interactive tracks at mds/csiro. In *Proceedings of the 7th Text REtrieval Conference (TREC-7)*, pages 465–474, 1998.
11. T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, June 1993.
12. M. Halliday and R. Hasan. *Cohesion In English*. Longman, 1976.

13. K. Kummamuru, R. Lotlikar, S. Roy, K. Singal, and R. Krishnapuram. A hierarchical monothetic document clustering algorithm for summarization and browsing search results. In *Proceedings of the 13th international conference on World Wide Web (WWW 2004)*, pages 658–665, 2004.
14. Y. Li, Z. A. Bandar, and D. McLean. An approach for measuring semantic similarity between words using multiple information sources. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):871–882, July/August 2003.
15. A. P. M. Albanese and A. M. Rinaldi. A semantic search engine for web information retrieval: an approach based on dynamic semantic networks. In *Proceedings of SIGIR Semantic Web and Information Retrieval Workshop (SWIR 2004)*, July 2004.
16. M. J. Mana-Lopez, M. De Buenaga, and J. M. Gomez-Hidalgo. Multidocument summarization: An added value to clustering in interactive retrieval. *ACM Transactions on Information Systems*, 22(2):215–241, April 2004.
17. G. A. Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, November 1995.
18. D. I. Moldovan and R. Mihalcea. Using WordNet and lexical operators to improve internet searches. *IEEE Internet Computing*, 4(1):34–43, 2000.
19. R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W. R. Swartout. Enabling technology for knowledge sharing. *AI Magazine*, 12(3):36–56, September 1991.
20. P. Pantel and D. Lin. Clustering: Document clustering with committees. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval (Tampere)*, pages 199–206, August 2002.
21. D. Paranjpe, G. Ramakrishnan, and S. Srinivasan. Passage scoring for question answering via bayesian inference on lexical relations. In *Proceedings of the 12th Text REtrieval Conference (TREC 2003)*, pages 305–310, 2004.
22. D. Roussinov and J. Robles. Web question answering through automatically learned patterns. In *Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries*, pages 347–348, June 2004.
23. G. Salton and C. Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 41(4):288297, 1990.
24. R. N. Shepard. Towards a universal law of generalisation for psychological science. *Science*, 237:1317–1323, 1987.
25. A. Sheth, C. Bertram, D. Avant, B. Hammond, K. Kochut, and Y. Warke. Managing semantic content for the web. *IEEE Internet Computing*, 6(4):80–87, 2002.
26. M. Stairmand. *A Computational Analysis of Lexical Cohesion with applications in Information Retrieval*. PhD thesis, Centre for Computational Linguistics, UMIST, Manchester, 1996.
27. K. Sumi, Y. Sumi, K. Mase, S. ichi Nakasuka, and K. Hori. Takealook: Personalizing information presentation according to user's interest space. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SCM'99)*, volume 2, pages 354–359, October 1999.
28. M. Sussna. Word sense disambiguation for free-text indexing using a massive semantic network. In *CIKM '93: Proceedings of the second international conference on Information and knowledge management*, pages 67–74, New York, NY, USA, 1993. ACM Press.
29. P.-H. Wang, J.-Y. Wang, and H.-M. Lee. QueryFind: Search ranking based on users'feedback and expert's agreement. In *Proceedings of the 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service (EEE'04)*, 2004.

30. M. Wu, M. Fuller, and R. Wilkinson. Using clustering and classification approaches in interactive retrieval. In *Inf. Proc. Manage*, volume 3, pages 459–484, 2001.
31. O. Zamir and O. Etzioni. Web document clustering: a feasibility demonstration. In *Proceedings of the 21th Annual International ACM/SIGIR Conference on Research and Developement in Information Retrieval*, pages 46–54, 1998.