Using Semantic Techniques to Access Web Data

Raquel Trillo[a,*], Laura Po[b], Sergio Ilarri[a], Sonia Bergamaschi[b], Eduardo Mena[a]

[a] *IIS Department, University of Zaragoza María de Luna 1, 50018 Zaragoza, Spain*
{raqueltl, silarri, emena}@unizar.es
[b] *II Dept., Univ. of Modena and Reggio Emilia, Via Vignolese 905, Modena, 41125, Italy.*
{po.laura, bergamaschi.sonia}@unimore.it

Abstract

Nowadays, people frequently use different keyword-based web search engines to find the information they need on the Web. However, many words are polysemous and, when these words are used to query a search engine, its output usually includes links to web pages referring to their different meanings. Besides, results with different meanings are mixed up, which makes the task of finding the relevant information difficult for the users, especially if the user-intended meanings behind the input keywords are not among the most popular on the Web.

In this paper, we propose a set of semantics techniques to group the results provided by a traditional search engine into categories defined by the different meanings of the input keywords. Differently from other proposals, our method considers the knowledge provided by ontologies available on the Web in order to dynamically define the possible categories. Thus, it is independent of the sources providing the results that must be grouped. Our experimental results show the interest of the proposal.

Keywords: Semantic Search, ontologies, categorization, disambiguation, semantic annotation
PACS: 89.20.Hh, 89.20.Ff
2000 MSC: 68-02, 68P99

[*] Corresponding author. Phone Number:+ 34 976762472 Fax: + 34 976761914

1. Introduction

The big explosion of the World Wide Web in the last fifteen years has made a great and ever-growing amount of information available to users. In this context, search engines have become indispensable tools for users to find the information they need in such an enormous universe. However, traditional search engine techniques are not the best choice when the user is interested in finding information about topics that are not very popular on the Web.

Thus, for example, the use of polysemous words in traditional search engines leads to a decrease in the quality of their output [1, 2]. For example, if a fan of movies inputs "life of stars" as a query because he/she is interested in the style of life of the people who are "actors/actresses who play a principal role", a search engine can return a very high number of hits. Particularly, in this case, Yahoo returns about 624 014 000 hits[1]. However, the first hit that refers to the style of life of famous actors/actresses is on the sixth page (the 54th hit), while the previous hits refer to other meanings of the word "star", such as "a celestial body of hot gases that radiates energy", the title of a book, different companies, etc. Unfortunately, users usually check only one or two pages of the results returned by the search engine [3]. Therefore, an approach is needed to minimize the number of hits that a user needs to revise, in order to give users a faster access to the information that they need.

The problem is that hits referring to different meanings of user keywords are usually mixed up in the output obtained from a search engine. Moreover, the top positions in the ranking of hits are usually occupied by hits referring to the most popular meanings of the keywords on the Web (in the previous example, "star" as a "celestial body"), hiding the huge diversity and variety of information available to users on the Web. So, presenting the results to users classified into different categories, defined by the possible meanings of the user keywords, would be very useful. Indeed, as it is claimed in several previous works [4, 5, 6], it provides interesting advantages. Thus, it helps the user to get an overview of the results obtained, to access results that otherwise would be positioned far down in a traditional ranked list, to find similar documents, to discover hidden knowledge, to refine his/her query by selecting the appropriate groups of web pages, to remove groups of documents from consideration, and even to disambiguate queries such as acronyms. Summing

[1] Data obtained on 31st October 2009.

up, it improves the user's experience by facilitating browsing and finding the relevant information.

In this paper we propose a new approach, based on semantic technologies, that is able to classify the hits provided by a traditional search engine into categories according to the different meanings of the input keywords. Moreover, it discovers the possible meanings of the keywords to create the categories dynamically at run-time by considering heterogeneous sources available on the Web. As opposed to clustering and other categorization techniques proposed in the literature, our system considers knowledge provided by sources which are independent of the indexed data sources that must be classified. Thus, it exploits intensional knowledge provided by ontologies available on the Web and/or by lexical resources (such as dictionaries, thesauri, etc.), instead of extensional knowledge extracted from the sources to be grouped by means of statistical information retrieval techniques. The approach presented in this paper can be seen as a web-browser plugin or as a meta-search engine such as Wahoo or Gowgle [7]. Moreover, it could be integrated in the server-side of current web search engines (similarly to what is proposed in [8]), which would increase the performance of the approach. Thus, the web search engine could maintain inverted indexes based on the meanings of the words instead of considering their written representation.

The structure of the rest of the paper is as follows. Firstly, an overview of the main components of our system is presented in Section 2. Secondly, the two main steps performed by the system proposed are described in Section 3 and in Section 4. After that, an experimental evaluation for different configurations of the components of the system is provided in Section 5 and some related works are presented in Section 6. Finally, some conclusions and plans for future work appear in Section 7.

2. Overview of the System

Along the last decades, different techniques to group similar documents have been proposed. However, traditional clustering algorithms cannot be applied to search result clustering [4, 9, 10] because, for example, it is not feasible to download and parse all the documents due to the need to provide quick results to the user. So, in this section, we first present the features that a clustering/categorization approach for web search should present. Then, we overview the architecture of our proposal, that complies with these features, and we illustrate the workflow of the system with an example.

2.1. Requirements of a Web Search Clustering/Categorization Approach

Several works have identified the features that a suitable clustering/categorization approach in the context of web search should exhibit [4, 6, 9, 11, 12]. Thus, in [9] the following six features are indicated:

- *Relevance*: the approach should separate the pages relevant to the user query from the irrelevant ones.
- *Browsable summaries*: it should provide informative summaries of the groups of pages generated.
- *Overlap*: one web page could belong to more than one group, as web pages may have overlapping topics.
- *Snippet-tolerance*: methods that need the whole text of the document should be avoided, due to the excessive time that they would require to download and process the documents, in favor of approaches that only rely on the snippets[2]. Moreover, in [9] the authors indicate that "Surprisingly, we find that clusters based on snippets are almost as good as clusters created using the full text [...]".
- *Speed*: the process to create the different groups of Web pages should be fast.
- *Incremental*: the process should also start as soon as some information is available, instead of waiting until all the information has been received, and be performed incrementally.

From these features, in [11] the *relevance*, *overlap*, and *incremental* properties are emphasized. Similarly, some authors propose slightly different features. For example, [4] considers *coherent clusters* (which implies the need to generate overlapping clusters), *efficiently browsable*, and *speed* (*algorithmic speed* and *snippet-tolerance*), whereas [12] indicates that the clustering should be *semantic*, *hierarchical*, and *online*. A more recent work [6] cites *meaningful labels*, *computational efficiency*, *short input data description* (snippets vs. whole web pages), *unknown number of clusters/categories* (i.e., not predefined), *overlapping*, and a friendly *graphical user interface*. However, the previous six features mentioned above are clearly a good representative.

[2] A *snippet* is a segment that has been snipped off the document returned as a hit. Typically, it is a set of contiguous text around the user keywords in the selected document.

2.2. Architecture of the System

Considering the previous features, we designed a system whose goal is to provide users with data that satisfy their information needs with little effort, even when they are looking for information not popular on the Web. The proposed system performs two main steps (see Figure 1 for a general overview):

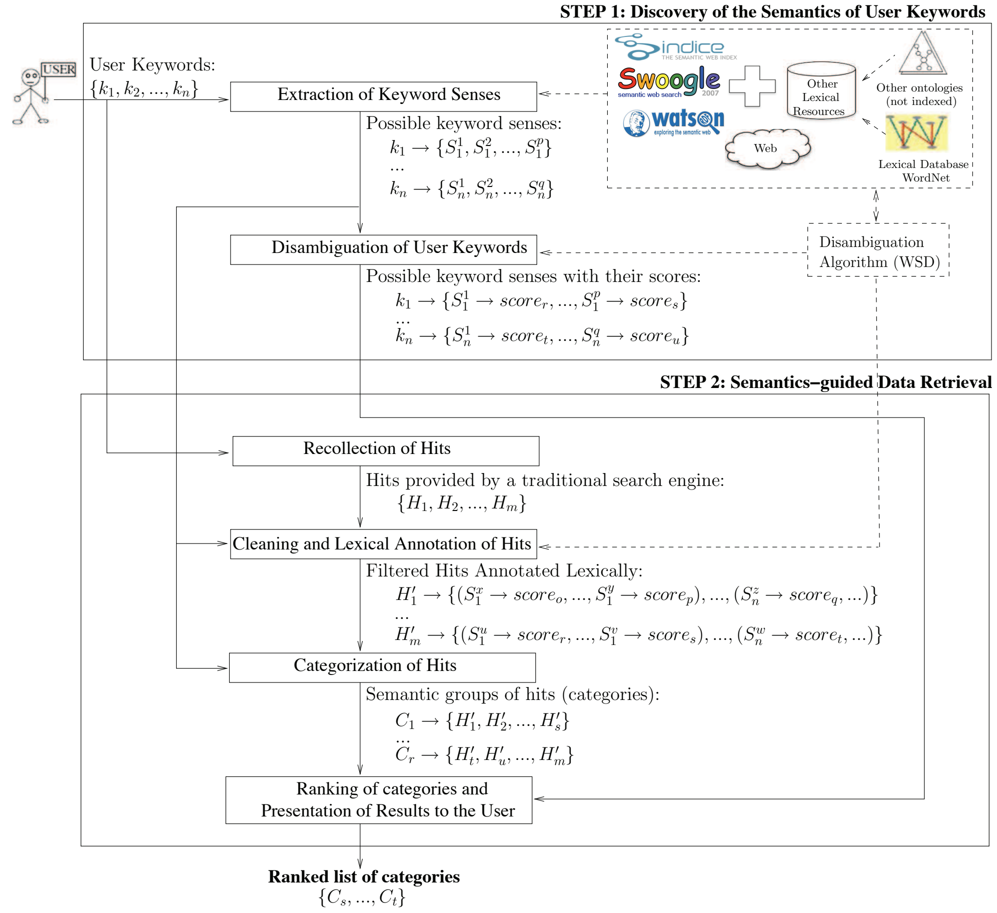

STEP 1: Discovery of the Semantics of User Keywords
USER
User Keywords:
$\{k_1, k_2, ..., k_n\}$
Extraction of Keyword Senses
indice THE SEMANTIC WEB INDEX
Swoogle semantic web search 2007
watson exploring the semantic web
Other Lexical Resources
Web
Other ontologies (not indexed)
Lexical Database WordNet
Possible keyword senses:
$k_1 \rightarrow \{S_1^1, S_1^2, ..., S_1^p\}$
...
$k_n \rightarrow \{S_n^1, S_n^2, ..., S_n^q\}$
Disambiguation of User Keywords
Disambiguation Algorithm (WSD)
Possible keyword senses with their scores:
$k_1 \rightarrow \{S_1^1 \rightarrow score_r, ..., S_1^p \rightarrow score_s\}$
...
$k_n \rightarrow \{S_n^1 \rightarrow score_t, ..., S_n^q \rightarrow score_u\}$
STEP 2: Semantics−guided Data Retrieval
Recollection of Hits
Hits provided by a traditional search engine:
$\{H_1, H_2, ..., H_m\}$
Cleaning and Lexical Annotation of Hits
Filtered Hits Annotated Lexically:
$H'_1 \rightarrow \{(S_1^x \rightarrow score_o, ..., S_1^y \rightarrow score_p), ..., (S_n^z \rightarrow score_q, ...)\}$
...
$H'_m \rightarrow \{(S_1^u \rightarrow score_r, ..., S_1^v \rightarrow score_s), ..., (S_n^w \rightarrow score_t, ...)\}$
Categorization of Hits
Semantic groups of hits (categories):
$C_1 \rightarrow \{H'_1, H'_2, ..., H'_s\}$
...
$C_r \rightarrow \{H'_t, H'_u, ..., H'_m\}$
Ranking of categories and Presentation of Results to the User
Ranked list of categories
$\{C_s, ..., C_t\}$

Figure 1: Overview of the approach

- *Step 1: Discovery of the Semantics of User Keywords.* The goal of this process is to find out the possible meanings (or senses) of the keywords that the user has entered as a query. Besides, for each keyword in the query, its different possible meanings are ranked and annotated with a score by considering the *context of the keyword*, that is, the rest of keywords in the query and their possible meanings. The score of a sense is a quantitive measure that ranges between 0 and 1, such that the senses with the highest scores are likely the ones intended by the user. For example, for the query "mouse price" the system discovered as possible meanings of "mouse" (k_1) the following ones: "Numerous small rodents typically resembling diminutive rats, usually having hairless tails", "A hand-operated electronic device that controls the coordinates of a cursor on the computer screen as you move it around", and "The surname of the famous cartoon Mickey by Disney" (denoted by S_1^1, S_1^2 and S_1^3 respectively). Similarly, as possible meanings of "price" (k_2) it discovered: "The amount of money needed to purchase something", and "United States operatic soprano" (denoted by S_2^1 and S_2^2 respectively)[3]. Then, the system provided a score for each sense, such as: $[k_1 \rightarrow \{S_1^1 \rightarrow 0.17, S_1^2 \rightarrow 0.28, S_1^3 \rightarrow 0.16\}, k_2 \rightarrow \{S_2^1 \rightarrow 0.23, S_2^2 \rightarrow 0.09\}]$[4].

- *Step 2: Semantics-guided Data Retrieval.* The goal of this process is to create the categories to be considered and to classify the hits provided by a traditional search engine into the different categories. The categories are defined from the possible meanings of the user keywords discovered in Step 1 (categories are defined by the possible combinations of the *possible keyword senses*). Then, each hit is classified in specific categories by considering whether it represents the same intended meanings as the corresponding categories or not. Besides, the hits in each category are ranked by considering the likelihood that they belong to that category and the original position of the hit in the rank provided by the Web search engine. Therefore, popular hits within a category will appear first, as they reference well-known Web pages with topics likely belonging to that category. Thus, for the previous example the created categories are $\{<S_1^1,\ S_2^1>$, $<S_1^1,\ S_2^2>$, $<S_1^2,\ S_2^1>$, ...$\}$. The category $<S_1^1,\ S_2^1>$ should contain the hits referring to the amount of money needed to buy little ro-

[3] This example was performed by considering the ontologies provided by Watson [13] as input of this process and the configuration parameters that lead to the best results, on 31st October 2009.

[4] These scores were obtained by using the WSD method described in Appendix A.1.

dents (mice). For example, it contains a hit with title "Boa Constrictors – Frozen Mice Prices" (`http://www.repticzone.com/forums/BoaConstrictors/messages/37416.html`) in the top positions.

Our approach can save a lot of effort to the user in locating the relevant hits. Thus, in the previous example, among the first 100 hits only those that were originally in the positions 28th, 64th, 67th and 83rd are relevant for the category $<S_1^1, S_2^1>$. A more detailed explanation of the discovery of the semantics of user keywords (Step 1) and the retrieval of the data by using semantics techniques (Step 2) is provided in Sections 3 and 4 respectively.

3. Discovery of the Semantics of User Keywords

In this step, the system needs to discover the intended meaning of the user keywords to consider the hits (returned by a traditional search engine) that correspond with that semantics. This requires finding out the possible meanings (interpretations or senses) of each input keyword (k_i) of the user[5], and then ranking the possible interpretations for each keyword. In other words, a *Word Sense Disambiguation* (*WSD*) algorithm [2] is performed in two phases at run-time, corresponding to the extraction of the possible senses of the user keywords (see Section 3.1) and their disambiguation (see Section 3.2).

3.1. Extraction of Keyword Senses

In this first phase, the system has to obtain a list of possible meanings for each keyword k_i (called *possible keyword senses* and denoted as $\{S_i^1, S_i^2, ..., S_i^u\}$), so semantic descriptions are required. These descriptions are provided by different sources such as thesauri, dictionaries, ontologies, etc.

At this point, we consider two possibilities to tackle this task, depending on the knowledge sources used for the sense discovery: 1) consulting a shared thesaurus such as WordNet [14], and 2) considering the shared knowledge stored in different ontologies available on the Web. We discuss the two approaches in Sections 3.1.1 and 3.1.2, respectively. Besides, we describe the set of experiments performed to evaluate the trade-off between these techniques in Section 5.1.1.

[5] Our approach is not meant to work with concepts that span over multiple words. So, if the user wants to express such concepts, then the corresponding set of words should be in quotes, which will allow the system to consider that concept as a single keyword.

3.1.1. Consulting a Shared Thesaurus such as WordNet

The first choice is to rely on a well-known general-purpose shared thesaurus, such as WordNet [14]. This source could provide the possible senses for a large number of words and usually also other semantic information (e.g., synonyms, hypernyms, hyponyms, etc.) for each possible meaning of a word. Besides, WordNet (like other semantic resources) provides possible senses for some compound nouns and proper nouns such as “credit card” or “George Bush”.

The main advantage of this option is that it provides a reliable set of the possible meanings of a keyword, and allows to share with others the result of the process (e.g., the set of senses obtained). Moreover, the fundamental peculiarity of a thesaurus is the presence of a wide network of relationships between words and meanings.

The disadvantage is that it does not cover with the same detail different domains of knowledge. So, some terms or meanings may not be present; for example, the term “developer” with the meaning “somebody who designs and implements software” is not stored in WordNet. Moreover, new possible interpretations of a word appear along time; for example, life as “the name of a popular magazine” or “the name of a film”. These considerations lead to the need of expanding the thesaurus with more specific terms (this can be easily done using a tool called *WordNet Editor*, which allows adding new terms and linking them within WordNet [15]). In contrast, other terms in the thesaurus may have many associated and related meanings; for example, the word “star” in WordNet has three very similar meanings: “someone who is dazzlingly skilled in any field”, “an actor who plays a principal role”, and “a performer who receives prominent billing”. Some users could consider all these meanings as equivalent ones, whereas others could be interested in treating them as different ones.

3.1.2. Consulting Available Online Ontologies

Another alternative is to consult the shared-knowledge stored in different ontologies available on the Web or several specialized search engines (such as Watson [13], Swoogle [16], or Sindice [17]) that provide us with relevant ontologies ranked according to their quality. This retrieval could provide redundant interpretations from different sources of knowledge, so the techniques based on synonym probability measurements defined in [18, 19] are used to remove redundancy. In short, these works rely on the following idea: the more ontologies consulted (each one representing the point of view of its creators), the more chances to find the semantics that the user had in mind when he/she entered the keywords. Firstly, the *ontological terms* that match the keyword (and also those that match their synonyms) are extracted from the corresponding ontologies by means of several techniques described in [18]. Then, the system treats the possible overlapping (different ontological terms representing the same meaning) among the senses retrieved. For this task, the different terms extracted are considered as the input of the following iterative algorithm. Each ontological term is compared with the rest of ontological terms and its similarity degrees are computed by using synonym probability measures. If the similarity degree between two terms is lower than a threshold given as a parameter (*synonym threshold*), then the two terms are considered to represent different meanings of the keyword; otherwise, they are considered synonyms (i.e., they represent the same interpretation of the keyword) and they are merged (integrated) into a single sense following the techniques described in [18, 19]. So, the integrated senses are new multi-ontological terms that are also compared with the rest of terms. Finally, the output of the process is a set of (single or integrated) senses, where each element corresponds to a possible meaning of the keyword.

The main advantage of this approach is the use of ontologies available on the Web, since it maximizes the possible interpretations for the keywords. For example, the meaning of "developer" related to "computer science" (that does not appear in WordNet) appears in some ontologies[6]. Moreover, it allows new interpretations of the words to be considered without extra effort. The system can also be set up to work with different synonym thresholds and allow the output to be dynamically changed based on the threshold established. Notice that if the synonym threshold is risen, then the integration of terms decreases, so more fine-grained senses are provided as output, whereas if the synonym threshold decreases there is a higher degree of integration and less interpretations are provided for the same keyword.

The main disadvantage of this approach is that it could introduce noise and irrelevant information. Thus, unless the search is restricted to a limited set of trusted ontologies, unreliable ontologies may be accessed. Besides, if too many individual interpretations are considered, the complexity of the

[6] For example, "developer" as a "programmer" was found in the following ontologies: `http://keg.cs.tsinghua.edu.cn/ontology/software`, `http://www.mindswap.org/2004/SSSW04/aktive-portal-ontology-latest.owl`, and `http://tw.rpi.edu/wiki/Special:ExportRDF/Category:Software_Developer`.

process could have a high cost in time, decreasing the average speed [19].

3.2. Disambiguation of User Keywords

In this second phase, the system assigns scores to the senses obtained in the previous phase, by considering as context the user query. Thus, for each user keyword, this process obtains a ranked list of possible senses with their scores, denoted as: $[k_1 \rightarrow \{S_1^j \rightarrow score_k, ..., S_1^o \rightarrow score_p\}, ..., k_n \rightarrow \{S_n^q \rightarrow score_r, ..., S_n^s \rightarrow score_t\}]$. A score offers a quantitative measure for the likelihood that a keyword sense is the sense intended by the user for that keyword. Depending on the method used to compute the scores, they can be considered either a probability or simply a value from 0 to 1 such that the senses with the highest values are likely the ones intended by the user. Several features can be considered in WSD approaches to estimate the likelihood of a certain meaning for a word or a compound term in the context of a document written in natural language. However, the complexity of the disambiguation process increases when the number of words available as context is low, as in this case where the context is the list of plain user keywords and their possible meanings. Indeed, user queries in keyword-based search engines normally have a length lower than five words, so techniques that rely on the *syntax of whole sentences* [20, 21] or the *collocation of words* [22, 23] cannot be fully exploited.

Nevertheless, even under those circumstances, in many cases it is possible for a human to select/discard meanings of the polysemous words. Thus, for example, the word star is expected to be used with the meaning “celestial body” when it appears with the word “astronomy”, and with the meaning “famous actor” when it appears with the word “Hollywood”. Therefore, we try to emulate this behavior by taking into account the context in which user keywords appear. So, each possible sense is tagged with a score according to the probability that it represents the intended meaning of the user. For this purpose, the system uses a disambiguation algorithm that can obtain a list of potential senses along with their scores or probabilities (instead of just obtaining the most probable sense as many disambiguation techniques do [2]). We consider this approach more useful than just retrieving the most probable sense for each keyword, especially in the case of ambiguous keyword sets such as “life of stars” (where the user could possibly be interested in either the “life cycle of the celestial bodies” or in the “style of living of famous people”). The potential senses for each user keyword and the corresponding scores will be used later to rank the categories shown to the user (see Section 4.4).

It is important to emphasize that the proposed architecture does not depend on a specific WSD method. This is a very positive aspect because not all the WSD methods are valid or perform well in all possible contexts, as indicated in [24]. So far, we have tested our prototype with the *Web-based disambiguation* method presented in [1, 25] and with the *Probabilistic Word Sense Disambiguation* (*PWSD*) method described in [24, 26]. PWSD methods are based on a probabilistic combination of different WSD algorithms, so the process is not affected by the effectiveness of a single WSD algorithm in a particular context or application domain. However, other approaches, that also consider the profile of the user or other information available in the disambiguation process, could also be used.

4. Semantics-guided Data Retrieval

The goal of this step is to provide the user with the hits retrieved by a traditional search engine, such as Google or Yahoo!, classified in the categories defined by the meanings of the user keywords. Moreover, the categories are ranked according to the interests of the user. This process is performed in four phases at run-time. In the rest of this section, we explain these phases in detail.

4.1. Recollection of Hits

Firstly, this phase requires performing a traditional search of hits on the Web, by taking as input the user query and using a traditional search engine such as *Google* or *Yahoo!*. This search returns a set of relevant ranked hits, which represent the web pages where the keywords appear. The order of the hits (i.e., the ranking of the results provided by the search engine used) depends on the specific techniques used by the engine for that task and its numerous internal parameters. Then, the hits returned by the search engine are provided as input to the next phase (the *Cleaning and Lexical Annotation of Hits*) incrementally, in *blocks of hits* of a certain size. In this way, new hits can be retrieved while the first blocks are being processed. Our prototype uses blocks of 100 hits but the block size is a configuration parameter. Notice that this process can be performed in parallel with the *Discovery of the Semantics of User Keywords* step (see Section 3).

4.2. Cleaning and Lexical Annotation of Hits

In this second phase, each hit obtained in the previous phase (composed of a title, a URL and a snippet) is automatically annotated lexically. Thus, firstly, each hit H_j goes through a *cleansing* process where stopwords are filtered out (creating the filtered hit H'_j). After that, the relevant words of the title and the snippet of each filtered hit H'_j are considered to perform a *lexical annotation* of the hit. A lexical annotation is a piece of information added to a term that refers to a semantic knowledge resource such as a dictionary, a thesaurus, a semantic network, or any other resource which expresses, either implicitly or explicitly, a general ontology of the world or a specific domain. So, for each filtered hit, this process obtains a list of annotations denoted as $H'_j \rightarrow \{(S_1^x \rightarrow score_o, ..., S_1^y \rightarrow score_p), ..., (S_n^z \rightarrow score_q, ...)\}$. The list of annotations represents the senses with which the user keywords are likely used in that hit. Moreover, the score associated to each annotation indicates its reliability. It should be noted that, in some situations, selecting only one sense for a user keyword in a certain hit is a difficult task even for a human. For example, a keyword can appear in the same hit with different senses (e.g., in the case of a hit corresponding to a dictionary entry) or two different senses for a keyword may have some overlapping (e.g., for the keyword "star", the meanings "an actor who plays a principal role" and "a performer who receives prominent billing"). Therefore, a list of annotations must be considered.

In our case, the annotation process is performed by considering each appearance of the user keywords in the filtered hit and its context (i.e., its relevant neighboring words), and by using WSD [1, 25] and PWSD [24, 26] methods. For this, firstly, each appearance of a user keyword k_i in the title or the snippet of a filtered hit H'_j is marked with its probable senses, that is, $k_i^t \rightarrow \{S_i^x \rightarrow score_o, ..., S_i^y \rightarrow score_p\}$, where k_i^t denotes the t-th appearance of k_i in H'_j. These annotations are used to perform the global annotation of the hit. Thus, when a user keyword sense S_i^x appears only once in the annotations performed, the sense and its corresponding score ($S_i^x \rightarrow score_o$) are incorporated to the list of annotations of the hit. Nevertheless, as a user keyword k_i could appear several times in H'_j, the same user keyword sense S_i^x could appear in several annotations of k_i and have a different score in each of them; in this case, the maximum of these scores is considered for the global annotation of the hit.

Notice that not all the possible senses discovered for a certain user keyword k_i in the *Extraction of Keyword Senses* phase (see Section 3.1) participate in the lexical annotation of a hit, but only a subset of these. Moreover, an additional *unknown meaning* is also considered for each user keyword. This feature allows the system to take into account the evolution of a natural language: new senses for a keyword appear when these senses start being used, and only after these senses become widespread they are integrated in the resources where the semantic descriptions are provided. Indeed, a sense inventory that a priori lists all relevant senses will never be able to cope with new usages, new words, a usage in a specialized context that the sense inventory does not cover, or simply an omission. Thus, in Natural Language / Computational Linguistics there are several WSD approaches to deal with this problem [27]. The approaches focus on unknown words, i.e. terms that are not classified in the reference dictionary, and unknown senses, i.e. meanings for a term that are not classified. Regarding the unknown senses, some approaches assume an equal probability for every possible sense when the meaning for the word is unknown [28], and others make use of an "unassignable" tag for those words where none of the senses can be applied (like suggested in the *Senseval* evaluation initiative, see `http://www.senseval.org/`). We follow the latter approach. So, when a user keyword in the hit cannot be annotated, it is assumed that it corresponds with the unknown sense with a score equals to 1. Thus, this score acts as a neutral element in the formulas used to compute the ranking of the hits in a specific category (see Section 4.3).

It should be emphasized that only the information provided by the title and snippet of a hit is used for the annotation, without accessing the full document by means of the URL provided. This mimics the behavior of a user, who usually tries to determine the topic of a web page by just looking at the information provided on the page of search results, without accessing the web pages unless it is needed. Indeed, as explained in Section 2.1, an approach that needs to download and analyze the whole documents would be unsuitable in the context of a traditional interactive web search, due to the requirement to provide a quick answer to the user[7]. Moreover, if needed, several hits could be annotated in parallel by using multiple computers/processors. Besides, notice that this process can be performed in parallel with the *Disambiguation of User Keywords* phase (see Section 3.2), but the *Extraction of Keyword Senses* phase (see Section 3.1) must have finished previously.

[7] Nevertheless, the exploration of whole documents could be considered for search tasks where the response time is not an issue, or for the offline construction of inverted indexes based on the meanings of words (*semantic indexes*) in specialized semantic web search engines.

4.3. Categorization of Hits

In this third phase, the hits (already annotated as a result of the previous process) are grouped in categories by considering their lexical annotations. Firstly, the system defines the categories that are going to be considered. Then, blocks of hits are classified. The potential categories are defined by considering all the possible combinations of possible keyword senses of the input keywords (i.e., the cartesian product of the possible sense sets of the user keywords). For example, if the user introduces two keywords (k_1 and k_2) and, in the previous step, two senses are discovered for k_1 (S_1^1 and S_1^2) and one sense for k_2 (S_2^1), then the following potential categories are considered: $<S_1^1, S_2^1>$, $<S_1^2, S_2^1>$, $<U_1, S_2^1>$, $<S_1^1, U_2>$, $<S_1^2, U_2>$, and $<U_1, U_2>$, where U_1 and U_2 represent the unknown meanings considered for the keywords k_1 and k_2 respectively. Then, each hit is assigned to the categories defined by the meanings of the input keywords corresponding to the lexical annotation of that hit. So, depending on the scores of the meanings that are assigned to the user keywords in a hit, the hit could be classified in different categories at the same time (i.e., the categories may overlap).

Finally, the hits classified in a category are ranked according to their relevance for that category. That is, the system performs a *score-based ranking*. For this purpose, when a hit is assigned to a category a score is also computed for the hit. This score is calculated by multiplying the scores associated to that hit for the different senses defining the category. Then, the hits within the category are ranked according to their scores (and hits with the same score are ranked according to the order returned by the web search engine), as the hits in top positions are considered more relevant for that category.

This process (as the previous ones) is performed incrementally with blocks of hits, and can be executed in parallel with the *Disambiguation of User Keywords* phase (see Section 3.2), but once the *Extraction of Keyword Senses* phase (see Section 3.1) has finished. Moreover, the different hits can also be classified in parallel.

4.4. Presentation of Results to the User

Finally, the results of the *Categorization of Hits* phase are presented to the user. The system shows, in different tabs or category links, the categories considered that contain hits[8]. The order of the tabs or category links depends

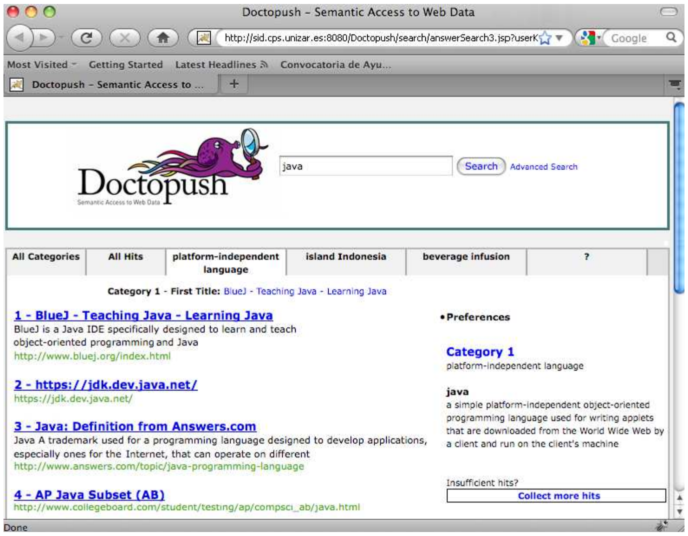
[8] Potential categories with no hits represent combinations of senses (of the input keywords) that are not detected in the hits collected.

on the probability that the corresponding category represents the semantics that the user had in mind when he/she wrote his/her query. So, to rank the categories, three elements are considered: 1) the scores obtained in the *Disambiguation of User Keywords* phase (see Section 3.2), 2) the percentage of hits classified in the category, and 3) the position of the first hit in that category in the ranking provided by the web search engine. The global score for a category C_x is defined as $score(C_x) = \alpha * score_{hitSenses} + \beta * score_{\%Hits} + \gamma * score_{pos1stHit}$, where α, β, and γ are coefficients to tune the formula[9], $score_{hitSenses}$ is obtained by multiplying the scores (computed in Step 1) for the senses defining that category, $score_{\%Hits}$ is equal to the number of hits assigned to the category C_x divided by the number of hits retrieved from the web search engine, and $score_{pos1stHit}$ is the inverse of the position of the first hit in C_x in the ranking provided by the web search engine. Besides, categories with unknown senses are considered less relevant, by assuming $score_{hitSenses} = 0$. Thus, for the scores previously obtained in the example of Section 2.2 ($S_1^1 \rightarrow 0.17$, $S_1^2 \rightarrow 0.28$, $S_1^3 \rightarrow 0.16$, $S_2^1 \rightarrow 0.23$, and $S_2^2 \rightarrow 0.09$), the category defined by $<S_1^1, S_2^1>$ is scored with 0.0488, that is $0.65 * (0.17 * 0.23) + 0.15 * 0.14 + 0.2 * (1/83)$, and it is ranked earlier than the category defined by $<S_1^1, S_2^2>$ (with global score 0.0115).

Showing all the categories instead of just the most probable one according to the scores computed, we acknowledge the possibility that the user might need to explore groups of web pages other than the most probable one. Nevertheless, the default behavior of the system is to hide the categories with a very low score, although the hidden categories will be shown if the user explicitly requests it. Our prototype owns an interface based on *AJAX* techniques [29] to allow users explore the first blocks of results while other blocks of hits are being processed in background. Currently, two graphical user interfaces are available (see Figures 2 and 3). One shows the information in tabs, which are labeled with relevant words related to the senses of the keywords defining the corresponding category ("?" for the unknown meanings). The other one allows navigating among the categories by using hyperlinks located on the left, and shows on the right the hits in the category selected.

The senses that define each category refer to knowledge sources (e.g., WordNet [14] or other online ontologies), where synonym words of such senses can be identified by the system. So, it is possible to retrieve new hits for

[9] The default values for our prototype are $\alpha = 0.65$, $\beta = 0.15$, and $\gamma = 0.2$.


Doctopush – Semantic Access to Web Data
http://sid.cps.unizar.es:8080/Doctopush/search/answerSearch3.jsp?userK
Google
Most Visited
Getting Started
Latest Headlines
Convocatoria de Ayu...
Doctopush – Semantic Access to ...
Doctopush
Semantic Access to Web Data
java
Search
Advanced Search
All Categories
All Hits
platform-independent language
island Indonesia
beverage infusion
?
Done

Category 1 - First Title: BlueJ - Teaching Java - Learning Java

1 - BlueJ - Teaching Java - Learning Java
BlueJ is a Java IDE specifically designed to learn and teach object-oriented programming and Java
http://www.bluej.org/index.html

2 - https://jdk.dev.java.net/
https://jdk.dev.java.net/

3 - Java: Definition from Answers.com
Java A trademark used for a programming language designed to develop applications, especially ones for the Internet, that can operate on different
http://www.answers.com/topic/java-programming-language

4 - AP Java Subset (AB)
http://www.collegeboard.com/student/testing/ap/compsci_ab/java.html

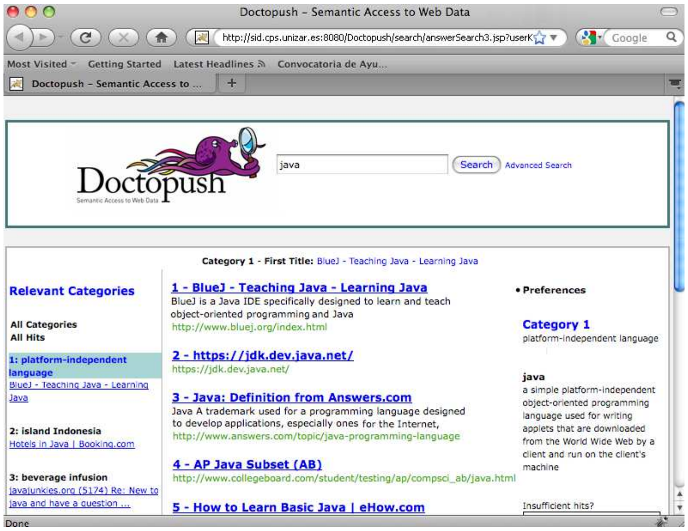
• Preferences

Category 1
platform-independent language

java
a simple platform-independent object-oriented programming language used for writing applets that are downloaded from the World Wide Web by a client and run on the client's machine

Insufficient hits?
Collect more hits

Figure 2: Snapshot of the current graphical user interface: tabs view


Doctopush – Semantic Access to Web Data
http://sid.cps.unizar.es:8080/Doctopush/search/answerSearch3.jsp?userK
Google
Most Visited
Getting Started
Latest Headlines
Convocatoria de Ayu...
Doctopush – Semantic Access to ...
Doctopush
Semantic Access to Web Data
java
Search
Advanced Search
Done

Category 1 - First Title: BlueJ - Teaching Java - Learning Java

Relevant Categories

All Categories
All Hits

1: platform-independent language
BlueJ - Teaching Java - Learning Java

2: island Indonesia
Hotels in Java | Booking.com

3: beverage infusion
javajunkies.org (5174) Re: New to java and have a question ...

1 - BlueJ - Teaching Java - Learning Java
BlueJ is a Java IDE specifically designed to learn and teach object-oriented programming and Java
http://www.bluej.org/index.html

2 - https://jdk.dev.java.net/
https://jdk.dev.java.net/

3 - Java: Definition from Answers.com
Java A trademark used for a programming language designed to develop applications, especially ones for the Internet,
http://www.answers.com/topic/java-programming-language

4 - AP Java Subset (AB)
http://www.collegeboard.com/student/testing/ap/compsci_ab/java.html

5 - How to Learn Basic Java | eHow.com

• Preferences

Category 1
platform-independent language

java
a simple platform-independent object-oriented programming language used for writing applets that are downloaded from the World Wide Web by a client and run on the client's machine

Insufficient hits?

Figure 3: Snapshot of the current graphical user interface: list view

a category by using these synonyms as input to the web search engine [30]. For example, the words “truck” and “lorry” can both be interpreted as “an automotive vehicle suitable for hauling”. However, if a user enters “lorry” in Google about 5 220 000 hits are obtained, but if the user enters "truck" the number of hits is approximately 166 000 000. Therefore, if the system considers both "truck" and "lorry" to search hits referring to that possible interpretation, a better coverage of the relevant results could be obtained. This process is particularly interesting when the user selects keywords that are not so common (instead of their most popular synonyms).

It should be noted that if a word used to extract new hits is polysemous, then the system needs to discard the hits that do not correspond with the meaning assigned to the category considered. For example, if a user inputs "actor" as a query and he/she is interested in the category with the meaning of "a theatrical performer", then he/she is also probably interested in "star" (as "film star"). However, there will be some hits returned by the search engine for the input "star" that are irrelevant for the category considered, such as those hits containing information about the "celestial body". So, the system performs a retrieval by using these synonyms and, after that, it must lexically annotate the new hits obtained with this enrichment (as explained in Section 4.2) and filter out those hits annotated with an irrelevant meaning. Nevertheless, this process does not increase the number of categories, as the new relevant hits retrieved will be added to the category under consideration.

5. Experimental Evaluation

After having developed a prototype of the system, we asked ourselves how to evaluate it. As indicated in [5], there is no general consensus about a good metric to evaluate the quality of the groups of web pages created; besides, comparing different searching approaches is not easy because most of them have different goals (oriented to a specific topic vs. of general scope, allowing queries whose length is one word vs. several words, etc.). Moreover, there is no well-known general benchmark for *Semantic Search*. Despite this fact, three different methodologies to evaluate this kind of systems are identified in [5]: 1) *anecdotal evidence* for the quality of the results, 2) *user surveys* with different input queries, and 3) *mathematical measures* (such as *purity*, *precision*, *recall*, etc.).

Based on these ideas and the experimental evaluation performed in similar works [30, 31], we evaluate our prototype by following a methodology based on *user surveys* and *mathematical measures* oriented to evaluate the system when the user queries are highly ambiguous even for a human. The first step was to select the queries to be considered. For this purpose, ten members of our research teams collected ambiguous queries from other works related to this topic [18, 30, 31]. Then, a set of twelve queries that show different behaviors of the system was selected for experimental evaluation: “java”, “jaguar velocity”, “desert storm”, “virus infection”, “life of stars”, “glasses catalogue”, “sun”, “mouse price”, “nurse job”, “house painting”, “bank account”, and “apple store”. The keywords in these sets have multiple possible meanings and the corresponding queries are highly ambiguous. The second step was to design the set of experiments to perform the evaluation of the different components of our approach: the *Extraction of Keyword Senses* (see Section 5.1.1), the *Disambiguation of User Keywords* (see Section 5.1.2), and the *Semantics-guided Data Retrieval* (see Section 5.2).

5.1. Evaluation of the Discovery of the Semantics of the User Keywords

In this section, the experiments to evaluate the first step of our approach, which implies the discovery of the intended meanings of the user keywords, is described. As explained in Section 3, this step consists of two phases: 1) the extraction of the possible senses for the user keywords, and 2) the disambiguation of the user keywords in order to obtain a ranked list of the potential senses for each keyword.

5.1.1. Extraction of Keyword Senses

The purpose of this experiment is to evaluate if the system can discover the potential meanings of the keywords entered by the users. As explained in Section 3.1, two alternatives can be considered as knowledge sources: a) a shared thesaurus such as WordNet and b) online ontologies available on the Web. Besides, when the second alternative is considered, the online ontologies can be accessed by means of specialized search engines such as Watson [13] and Sindice [17]. So, in this experiment, we compare the performance of these approaches.

To perform this experiment, 55 testers (students of Economics and Project Management at the University of Zaragoza) were recruited. First of all, the testers were asked to fill a form where they indicated all the possible senses (by writing a short definition) that came to their mind for the individual keywords in the queries considered (i.e., java, jaguar, velocity, desert, storm, virus, infection, life, stars, glasses, catalogue, sun, mouse, price, nurse, job, house, painting, bank, account, apple, and store). Afterwards, the testers were provided with the senses automatically discovered by the system, and they were asked to mark the senses that they considered redundant or noise. To obtain the senses, the system was run with three possible configurations: 1) using WordNet, 2) using online ontologies indexed by Watson, and 3) using online ontologies indexed by Sindice[10].

We analyzed the data provided by the testers in order to study the quality of the senses automatically discovered by using the three configurations considered. For this purpose, we adapted the classical concepts of *precision*, *recall*, and *F-measure* [5]. So, we considered the *coverage*, which indicates the percentage of a tester's senses discovered by the system (number of senses indicated by a tester which are discovered by the system / number of senses indicated by the tester), the *relevance*, which indicates the percentage of senses discovered by the system that are relevant for a tester (number of senses discovered by the system relevant for the tester / number of senses discovered by the system)[11], and the adapted *F-measure*, which considers the weighted harmonic mean of the coverage and the relevance ($2 * coverage * relevance/(coverage + relevance)$). Average results for all the testers are provided in the following.

The *coverage* measures comparing the three configurations are provided in Figure 4. The figure shows that a high percentage of senses can be already found in WordNet (on average 89%), although the percentage using the online ontologies is slightly higher (on average, 94% by using Watson and 96% by using Sindice). Besides, we would like to emphasize that in most cases the senses not found by the system with any of the three configurations refer to proper names (e.g., *House* as the name of a popular TV series, *Storm* as the name of a character of the superheroes *X-Men*, *Life* as a popular magazine and the name of an album by Ricky Martin, etc.). However, for some specific keywords (*jaguar*, *sun*, and *apple*), using the online ontologies the system allows to discover more possible senses referring to proper names (the brand of a car, *Sun Microsystems*, and *Apple Inc.*). Moreover, the highest coverage is obtained by using Sindice because most retrieved senses are from high-quality semantic resources such as *DBpedia* [32], *OpenCyc* [33] and *YAGO* [34], which provide us with a wide range of senses for each user keyword.

The *relevance* measures comparing the three configurations are provided

[10] In our prototype, by default, a maximum of 30 semantic resources are analyzed per keyword.

[11] Notice that both numerators and denominators of the formulas defining the metrics are different.

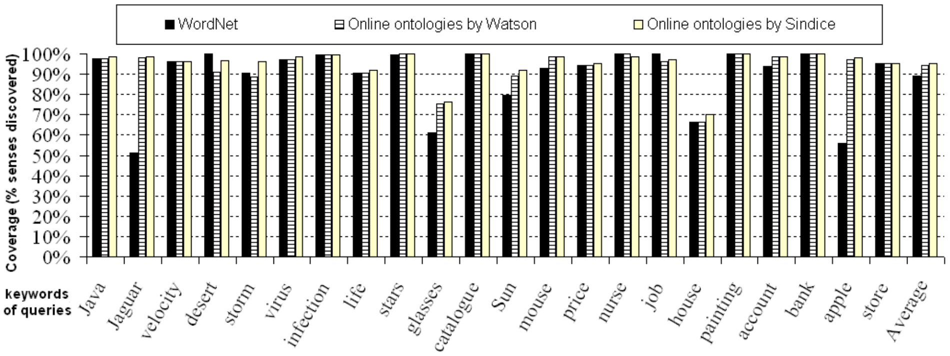


Figure 4: Extraction of Keyword Senses: coverage

in Figure 5. The figure shows that many senses discovered with the online ontologies are just *noise* (on average, the relevance is 41% with Watson and 49% with Sindice, and it increases up to 86% with WordNet). Moreover, by using Sindice, we observed that the relevance decreased when the system accessed ontologies different from DBpedia, OpenCyc and YAGO. Thus, for the set of keywords where other ontologies were also retrieved (java, velocity, storm, infection, life, sun, mouse, job, house, painting, bank, account, and store) the relevance (on average 38%) was lower than if only the three ontologies mentioned had been accessed (on average 64%). In some cases, such as for example with *infection*, *bank*, and *life*, many senses indicated by WordNet are not considered relevant because the testers interpreted that several senses actually referred to the same meaning. So, using online ontologies contributes to increase the coverage of the discovery of senses, but it also introduces some *noise*, since some ontologies available online are unreliable or poorly defined. Moreover, the sense integration approach mentioned in Section 3.1.2 may be unable to integrate some senses that actually refer to the same meaning (especially in cases where the senses are poorly defined), leading to redundancy in the senses provided.

The values of the adapted *F-measure* for the three configurations are provided in Figure 6.

Summing up, the system is able to discover the possible senses of the keywords entered by the users with a high accuracy. Using only WordNet as a knowledge source increases the relevance of the senses discovered, but the use of online ontologies increases the coverage. We have considered this trade-off, based on the F-measure obtained, and decided to use only WordNet

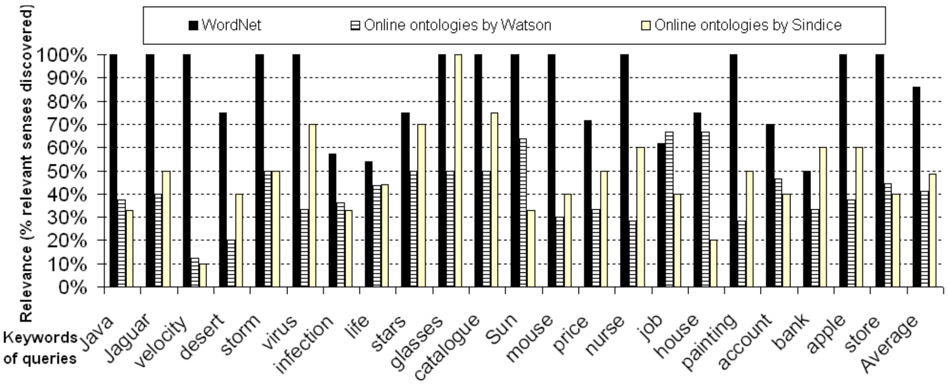


Figure 5: Extraction of Keyword Senses: relevance

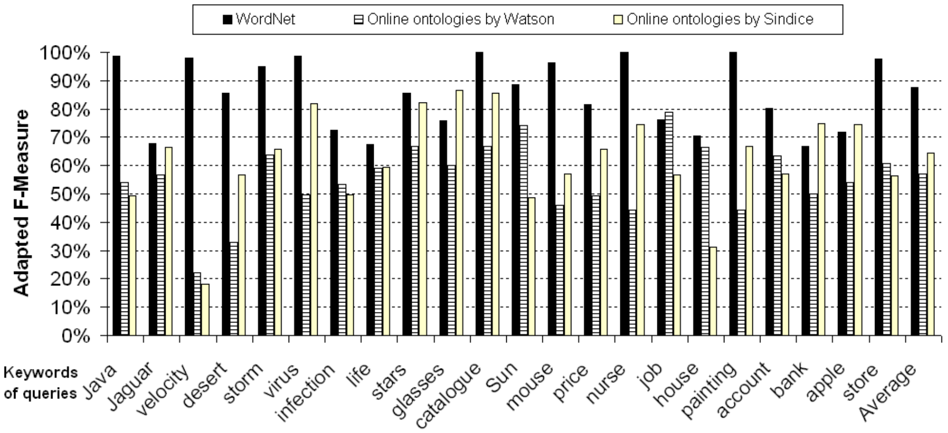


Figure 6: Extraction of Keyword Senses: adapted F-measure

in the rest of our experiments. However, it is important to emphasize that our approach can also work with online ontologies, either a predefined set of trusted ontologies or ontologies retrieved by a specialized search engine such as Watson [13], Swoogle [16], or Sindice [17]. The interest of this alternative will probably grow as the quality of online ontologies and the mechanisms to rank them improve.

5.1.2. Disambiguation of User Keywords

The purpose of this experiment is to evaluate the ability of the system to obtain a ranked list of possible senses for the user keywords. In this experiment, we consider the disambiguation method described in [1, 25], although the other disambiguation methods tested in our system [24, 26] could be used instead (see Appendix A.2).

In this case, the testers are presented with the different queries mentioned at the beginning of Section 5, along with the senses discovered by the system for each keyword in the query considered. Then, the testers were asked to rank the senses of each individual keyword from most probable to least probable according to the context (i.e., the query considered). After that, the testers' rankings were compared with the ranking obtained by our system.

We analyzed the data provided by the testers in order to assess if our system is actually able to estimate the user's intention. For this purpose we define the *System Disagreement for a tester k* (SD_k), for each query, in the following way:

$$SD_k = \frac{\sum_{j=1}^{N} \frac{\sum_{i=1}^{n_j} \left| Pos_{s_j^i}^{tester_k} - Pos_{s_j^i}^{system} \right|}{\sum_{i=1}^{n_j} i}}{N}$$

where n_j is the number of senses of the keyword k_j, $Pos_{s_j^i}^{tester_k}$ is the position of the i-th sense of the keyword k_j (s_j^i) in the ranking provided by the tester k for the keyword k_j, $Pos_{s_j^i}^{system}$ is the position of the sense s_j^i in the ranking provided by the system, N is the number of keywords in the query, and $\sum_{i=1}^{n_j} i$ is an upper bound of the maximum value that $\sum_{i=1}^{n_j} |Pos_{s_j^i}^{tester_k} - Pos_{s_j^i}^{system}|$ can reach. So, the value of SD_k ranges between zero and one: zero indicates that the two rankings are exactly the same and values close to one indicate that one ranking is the inverse of the other. A *global level of disagreement* SD between all the testers and the system is defined as $SD = \frac{\sum_{k=1}^{M} SD_k}{M}$, where M is the number of testers (55 in our case).

Considering the SD value in isolation would be unfair, as different users usually propose different rankings for the same set of keywords. So, we also compute a measure of the differences between the rankings provided by the users for each query. So, we define the *Tester's Disagreement* for a tester l regarding the rest of the testers (TD_l) in the following way:

$$TD_l = \frac{\sum_{k=1,k\neq l}^{M} \frac{\sum_{j=1}^{N} \frac{\sum_{i=1}^{nj} \left| Pos_{s_j^i}^{tester_k} - Pos_{s_j^i}^{tester_l} \right|}{\sum_{i=1}^{n_j} i}}{N}}{(M-1)}$$

where $Pos_{s_j^i}^{tester_l}$ indicates the position of the sense s_j^i for the keyword k_j in the ranking provided by the tester l, and the rest of the elements have the same meanings as in the formulas for SD_k and SD. A *global level of disagreement* TD among all the testers for a specific query is defined as $TD = \frac{\sum_{i=1}^{M} TD_i}{M}$.

We consider that the ranking provided by the system for the senses of the keywords for a specified query is correct as long as $SD \leq TD$. According to this criterion, the system performs well except for "bank account" (see Figure 7). So, we analyzed this query in detail. When the terms of the query are considered independently, the result for "bank" is SD=0.0978 and TD=0.1638, and the result for "account" is SD=0.5533 and TD=0.4222. Therefore, we focused on the ranked lists provided for the term "account". We noticed that, even though for the senses ranked in the top positions by the system there was agreement with the users, the rankings provided by the testers for the least relevant senses of "account" and the one provided by the system were slightly different. The reason is that the testers kept the least relevant senses in the same order as the possible senses were presented to them, while the system exchanged the order of two of these senses that had a very similar score. Overall, the good results obtained show the interest of disambiguating the user keywords in order to rank the categories that are shown to the user.

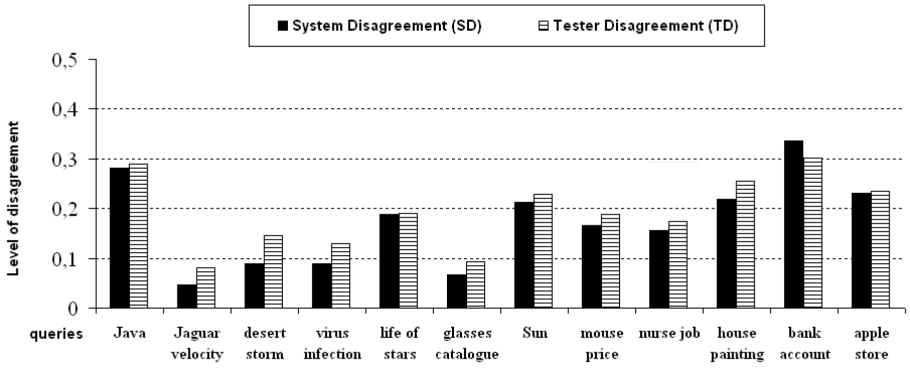


Figure 7: Disambiguation of User Keywords: Tester Disagreement and System Disagreement

5.2. Evaluation of the Semantics-guided Data Retrieval

In this experiment, we evaluate the semantics-guided data retrieval step described in Section 4. For this purpose, we considered the first 100 hits returned by Yahoo! for each query presented at the beginning of Section 5. After that, two experts manually classified these hits in the potential categories defined by the possible senses of the user keywords (as described in Section 4.3). The experts classified the hits by looking at the whole web pages, when needed, as in many cases looking at the URL, title and snippet was not enough to decide the topic of a hit. Expert disagreements were carefully analyzed and sometimes lead to classifying one hit within more than one category (i.e., assigning more than one sense to a user keyword in that hit).

Once the hits were manually classified, the first 100 hits retrieved for each user query by Yahoo! were also automatically classified by using the general approach described in Section 4. Notice that, as mentioned previously, this approach is orthogonal to the specific disambiguation algorithm used in the lexical annotation of the hits. Therefore, we decided to analyze the performance of four algorithms, described in Appendix A: the *Web Relatedness* method, and three *PWSD methods* denoted *DMP-A*, *DMP-B*, and *DMP-C*. That is, four different configurations of our system are evaluated.

For each configuration and also for the direct search performed by using Yahoo!, we evaluate the effort required by a user to locate a relevant result. For this purpose, we count the number of hits that he/she should revise to reach the first hit with the intended meaning given by the user to the input query. Of course, each query can have multiple interpretations. For example, for “mouse price” the user can ask about “the cost of the animal”, “the cost of the computer pointer device”, etc. So, we consider all the possible interpretations that appear in the first 100 hits and report average results. We would like to clarify that the expert classifications were used to determine automatically where the first hit relevant for a category is.

The experimental results are shown in Figure 8. Notice that, when using Yahoo!, a user may need to revise many hits to reach a particular meaning which is not popular on the Web. For example, for “apple store” the first hit returned by Yahoo! where “apple” has the meaning “fruit with red or yellow or green skin and sweet to tart crisp whitish flesh” appears in position 22nd. However, with the four configurations considered for our system it can be found in the top position of the corresponding category. As shown in the figure, all the disambiguation algorithms evaluated provide good results, although their relative accuracy depends on the keyword set considered. Obviously, a little overhead is introduced in order to select the intended category. Nevertheless, it is worth it when the intended meanings are not the most popular on the Web. Thus, for example, when the DMP-C method is used, the number of categories with hits is only 7 on average. Besides, the categories are ranked. Moreover, the user can also see the original result returned by the traditional search engine used, which avoids the aforementioned overhead when the most popular interpretation on the Web is required.

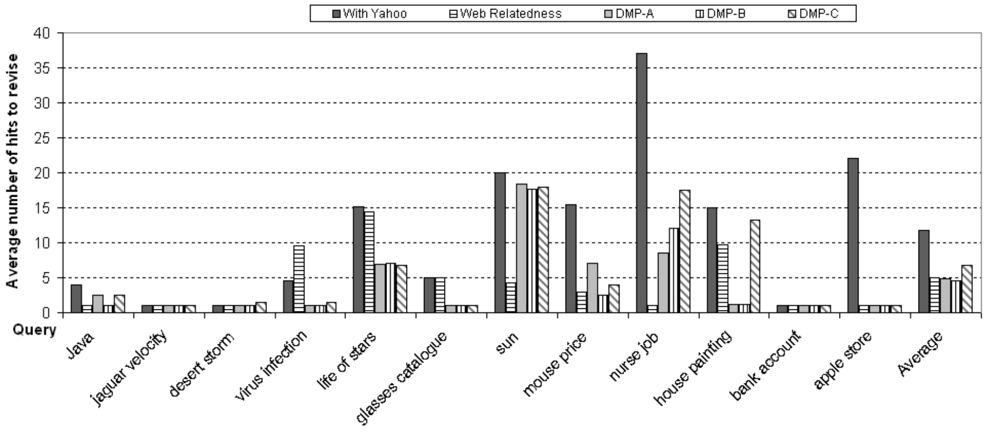


Figure 8: Evaluation of the Semantics-guided Data Retrieval

With the automatic classification, it is unavoidable that some hits appear in the wrong categories, particularly when the context used to automatically annotate the hits is poor (i.e., when the number of words in the title and snippet is low and/or when the words are not very relevant). Indeed, as mentioned before, in many cases not even a human can tell the topic of a web page just by looking at the information provided as part of the hit (URL, title, and snippet). However, even with these difficulties, as highlighted by the *Average* column in Figure 8, with all the WSD techniques tested the effort required by users is reduced. Regarding the time required by our approach, when the system is configured to use the *PWSD methods* considered, on average 20 seconds are spent to process a block of hits (100 hits) sequentially. Thus, even though our prototype has not been optimized yet for performance, the time spent proves that the proposal can be used in an interactive online scenario. So, the results obtained are promising and show the interest of the approach described in this paper.

6. Related Work

The clustering of data is a problem that has been studied for a long time and applied in many different scenarios, based on the so-called *Cluster Hypothesis* that states that relevant documents tend to be more similar to each other than to irrelevant documents [5, 35] (i.e., there is a high intra-cluster similarity and a low inter-cluster similarity [5]). In particular, several methods have been proposed to cluster collections of documents (hierarchical methods [36], K-means-based approaches [9], etc.). As opposed to *classification methods* [37, 38, 39], *clustering methods* do not require predefined categories (the categories are discovered during the clustering process) and they are therefore more adaptive to different types of queries [5, 10]. It may seem that our approach considers a set of predefined categories (the meanings of the user keywords); however, the potential categories used to allocate the hits retrieved are dynamically obtained depending on the user keywords and the knowledge bases queried. Besides, different senses retrieved from the knowledge sources can be merged or refined by considering *synonym probability measures* and a *synonym threshold* given as a parameter.

In the context of the web, document clustering can be either pre-computed over a complete collection of documents (e.g., [40]) or computed on-the-fly considering only the documents returned as a result of a search. The latter case (called *ephemeral clustering* in some works, such as [41]),which our proposal is based on, leads to better results because it focuses only on the documents (hits) that are considered relevant for the user query [4, 35, 42]. Besides, it adapts more naturally to the fact that the Web is constantly evolving.

Several search (or meta-search) engines that perform clustering of results are available on the Web, such as *Clusty* (`http://clusty.com`) by *Vivisimo* (a Carnegie Mellon University's software spin off), *Grokker* (`http://www.grokker.com`), *Armil* [43] (`http://armil.iit.cnr.it`), *iBoogie* (`http://iboogie.com/`), *Cuil* (`http://www.cuil.com`), *JBraindead* (`http://bender.lsi.uned.es:8080/ModuloWeb/jbraindead.html`), *WebCat* [44] (`http://ercolino.isti.cnr.it/webcat`), and *SnakeT* [5, 45] (*SNippet Aggregation by Knowledge ExtracTion*, `http://snaket.di.unipi.it`). Unfortunately, most of the previous search engines are commercial products and little technical details are available about their implementation. Anyway, from the previous works, *Clusty* is considered the state-of-the-art in many researches in this area [5, 46, 47, 48].

It is not our goal to provide an in-depth or exhaustive study of existing clustering approaches oriented to web snippets; for that, we refer the interested readers to [5, 6]. Nevertheless, it should be emphasized that our approach distinguishes itself from other proposals because of its ability to use, not the extensional knowledge provided by the data that have to be grouped by means of statistical techniques from Information Retrieval, but the intensional knowledge provided by sources which are independent of the indexed data sources. Thus, we precisely consider the intended semantics of the keywords introduced by the user. Some other proposals that also use semantic techniques are [30, 31]. However, in [30] only WordNet can be used and the existence of a predefined set of categories is assumed. In [31], the use of WordNet is also proposed but they group different senses of a word. Nevertheless, this system is limited to queries with a single keyword and does not allow overlapping categories (i.e., hits being classified in more than one category). These limitations are avoided with our proposal.

To conclude this section, it is also interesting to reference works that focus on query reformulation/refinement, such as the *Wahoo* and *Gowgle* demonstrators (available at `http://watson.kmi.open.ac.uk/wahoo` and `http://watson.kmi.open.ac.uk/gowgle`) described in [7]. These systems are also considered meta-search engines (based on Yahoo! and Google, respectively), and they also use semantic information obtained from Watson [13]. However, Wahoo and Gowgle use the information provided by online ontologies to refine (generalize or specialize) the user query, whereas the approach presented in our paper focuses on the categorization of the hits obtained by the web search engine for the user query. Thus, along with the information retrieved from the web search engine, and for each user keyword, Wahoo and Gowgle provide terms (obtained from online ontologies) that the user can select to either replace the original user keyword or to add it to the user query. After the reformulation of the query, the new user query is sent to the web search engine again. This process is iterative, allowing the user to refine his/her query as many times as he/she wants. On the other hand, the approach presented in our paper classifies the hits obtained by the web search engine in different categories created by using semantic information and Word Sense Disambiguation techniques. Therefore, we consider Wahoo and Gowgle complementary works to ours. Indeed, our approach could be integrated in those systems to show the hits retrieved in several categories instead of in a single list. Moreover, the demonstrators could be integrated in our system to support the iterative refinement of the user keywords.

7. Conclusions and Future Work

We have presented a semantics-based approach to group the results returned by a standard search engine in different categories defined by the possible senses of the keywords in the user query. Our proposal has been designed considering desirable features identified in the literature [9] for this kind of systems: 1) *relevance*: the hits that are probably more relevant for the user query are presented in a single category at the top of the ranked list of the groups identified; 2) *browsable summaries*: each category is labeled with the selected meanings of the user keywords and with the title of the most representative hit; 3) *overlap*: a single hit could be classified in different categories; 4) *snippet tolerance*: only the snippets of the hits are used to form the groups, without accessing their whole corresponding document; 5) *speed and incremental*: hits are processed in blocks of a configurable size incrementally, and the prototype uses AJAX techniques [29] to continue processing in background while the user is interacting with the system. Moreover, the proposed framework can be assembled using different components for some of the proposed tasks, such as different disambiguation algorithms or sense discovery approaches that use different types of knowledge sources. Thus, we have adapted our previous work in other areas such as ontology matching [26] and probabilistic word sense disambiguation for Data Integration Systems [49] to the context of web information retrieval.

The experimental results obtained prove the effectiveness of our approach, especially when users are searching for information which is not the most popular on the Web. However, the user may still need to invest some effort to locate the hits relevant to his/her query, even when browsing the hits within the correct category (i.e., when the semantics of the keywords in both the hits and the user keywords have been identified). Thus, the user could be looking for hits corresponding to several different queries within a category. For example, if a user inputs “fish” and “person” and indicates his/her interest in the category whose meanings are “a creature that lives and can breathe in water” and “a human being”, he/she could be trying to find information for either mermaids (“a fabled marine creature, typically represented as having the upper part like that of a woman, and the lower like a fish”) or fishermen/fisherwomen (“people that earn their living fishing”) and these options are mixed yet. So, one more phase might be included to deal with these situations. Moreover, currently our approach does not deal with compound nouns[12] (such as "credit card") and proper nouns (such as "Tower Bridge"), that could appear in the title and/or the snippet of the retrieved hits. So, techniques that support the disambiguation of compound nouns (e.g., [50, 51]) and Named Entity Recognition (NER) techniques [52] will be adopted. In particular, we will study the possibility to apply or adapt the approach for compound nouns interpretation that we proposed in [53] in the context of schema matching.

We will also invest efforts in improving the current prototype and we plan to make it available on the Web, to allow more users to evaluate it. Particularly, an important goal is to study usability issues in order to enhance the current graphical user interface and apply parallelization and cache techniques to improve the performance. Finally, we plan to adapt the approach presented in this paper to be used against heterogeneous structured or semi-structured data repositories such as databases [54] or UDDI repositories following the ideas of Bernstein et al. [55], who consider that "Keyword search may even be used against structured data to get a quick feel for what is available and set the stage for more precise integration."

Acknowledgements

This paper is the result of a joint work started during Raquel Trillo's visiting period at the IIS Department of the University of Modena and Reggio Emilia from April to September 2009. This work was partially supported by the MUR FIRB Network Peer for Business project (`http://www.dbgroup.unimo.it/nep4b`), the "Searching for a needle in mountains of data!" project funded by the Fondazione Cassa di Risparmio di Modena within the Bando di Ricerca Internazionale 2008 (`http://www.dbgroup.unimo.it/keymantic/`), and the CICYT project TIN2007-68091-C02-02. We would like to thank Arantza Illarramendi and Serena Sorrentino for their valuable comments and support. We also thank the anonymous reviewers, who have provided us very interesting and useful comments that helped us to improve our work. The current graphical user interface of the prototype has been developed by Víctor Martínez-Burgos and its logo has been designed by José Ilarri.

[12] Compounds nouns are two or more nouns that function as a single unit.

Appendix A. A Brief Overview of the Disambiguation Algorithms Evaluated

In this appendix, we present the basics of the disambiguation algorithms considered in this paper for experimental evaluation. All of them proved useful to validate the interest of our proposal (see Section 5).

Appendix A.1. Web-based Disambiguation

The *Web-based disambiguation* method processes each user keyword iteratively, taking into account the possible senses of the other keywords to disambiguate it. In this process, each keyword sense is compared with the rest of senses in the context. Besides, a score is assigned to each possible sense. The comparisons are performed by computing a relatedness measure based on the frequencies of hits returned by a search engine[13]. Specifically, we define the *Search-engine-based semantic relatedness* between two search terms x and y as:

$$relSearchEngine(x, y) = e^{-2NSED(x,y)}$$

where $NSED(x, y)$ is the *Normalized Search Engine Distance* [56] between x and y. For more details, see [1, 25].

Appendix A.2. Probabilistic Word Sense Disambiguation Algorithms

The combination paradigm known as *ensembles of classifiers* is a very well-know approach for WSD in the *NLP* (*Natural Language Processing*) community. It helps to reduce variance and to provide more robust predictions. Ensemble methods are becoming more and more popular, as they allow to overcome the weaknesses of single approaches [2]. The key for improving the classification results is that different WSD algorithms combined commit to non-correlated errors. Different combination strategies can be applied, such as *majority voting*, *probability mixture*, *rank-based combination*, *maximum entropy combination*, and *probabilistic combination*.

Specifically, PWSD follows a *probabilistic combination* strategy. PWSD satisfies three important constraints: (1) it is an automatic technique (only a few configuration settings are required), (2) it is flexible (i.e., it can combine any set of WSD algorithms), and (3) the output of the method does not

[13] Any web search engine can be used (e.g., Google or Yahoo!).

commit to an exact sense for a term under consideration, but to a set of possible senses that represent the term. It relies on the *Dempster-Shafer theory of evidence* [57, 58], as it combines the output of a set of WSD algorithms by using the Dempster-Shafer's rule of combination. In this way, PWSD associates a probability value to each sense selected to disambiguate a term; this value shows the uncertainty of the disambiguation process. It should be noted that, in general, the sum of the probabilities assigned to the senses of a keyword is not 1 when the Dempster-Shafer's combination is used. The reason is that there is a certain probability assigned to the *ignorance* of the WSD algorithms combined.

In the experiment presented in Section 5.2, we considered three different PWSD algorithms (denoted *DMP-A*, *DMP-B*, and *DMP-C*), which we summarize in the following. In all the evaluations, the disambiguation has been performed over all the terms contained in the snippets. This choice was due to the fact that the disambiguation of one word can affect others in its context [59]. Disambiguating the keywords only is unfeasible, as the keywords need the context where they are located in order to be correctly disambiguated.

Appendix A.2.1. DMP-A

The first evaluation (called *DMP-A* in Section 5.2) involves three WSD algorithms: the WordNet Domains Disambiguation algorithm (WND) [49], the Gloss Similarity Disambiguation algorithm (GS) [60], and the WordNet First Sense heuristic (WN1S).

The *WordNet Domains Disambiguation* (*WND*) algorithm tries to disambiguate terms exploiting information supplied by *WordNet Domains* (`http://wndomains.itc.it/`). WordNet Domains has been proven a useful resource for WSD [61], as it allows to overcome one of the main issues of WordNet: its excessive granularity in distinguishing the different senses. WND computes the prevalent domains over all the terms belonging to a snippet and produces an ordered list of the more frequent domains. Choosing the maximum number of domains is the only configuration setting required (by default the number of domains is 3).

The *Gloss Similarity Disambiguation* (*GS*) algorithm is based on mining the glosses (namely, textual definitions) belonging to terms in a thesaurus (WordNet in our case). GS is inspired by Lesk's method [62] and is based on maximizing the similarity of the meanings assigned to the schema elements. The rationale of GS is that the words contained in the glosses of the possible senses for the terms in the vicinity of a given term t should contain more words related to a particular gloss of a sense of t. For example, when disambiguating *bank* in a snippet that contains the terms *account*, *bank*, *branch*, *number*, and *IBANcode*, we can expect the glosses of these terms to collectively contain more words related to the sense of *bank* as a financial institution than to its sense as a hydraulic engineering artifact.

The *WordNet First Sense* (*WN1S*) heuristic returns the first WordNet meaning for a given term. It has been observed that it is quite difficult for a WSD algorithm to beat the WN1S. We consider that the WN1S provides a good default value for words which cannot be disambiguated by using other WSD algorithms. Therefore, WN1S forms a part of our first evaluation case *DMP-A*.

Appendix A.2.2. DMP-B

In the second evaluation (called *DMP-B* in Section 5.2) we remove the algorithm that votes for the most frequent sense (WN1S), so the evaluation is performed only on the outputs of WND and GS. In this case, for the keywords where no algorithm provides a disambiguation, each of the possible senses of the keyword is assigned the same probability value.

Appendix A.2.3. DMP-C

The third evaluation (called *DMP-C* in Section 5.2) combines WND and GS as in *DMP-B*, but it changes the configuration of WND by selecting only the most prevalent domain instead of the three more prevalent domains.

References

[1] J. Gracia, R. Trillo, M. Espinoza, E. Mena, Querying the web: A multi-ontology disambiguation method, in: Sixth Int. Conf. on Web Engineering (ICWE'06), Palo Alto, California, USA, ACM, 2006, pp. 241–248.

[2] R. Navigli, Word sense disambiguation: A survey, ACM Computing Surveys 41 (2) (2009) 10:1–10:69.

[3] B. J. Jansen, U. Pooch, A review of web searching studies and a framework for future research, Journal of the American Society of Information Science and Technology 52 (3) (2000) 235–246.

[4] O. Zamir, O. Etzioni, Grouper: a dynamic clustering interface to web search results, in: Eighth Int. World Wide Web Conference (WWW'99), Toronto, Canada, Elsevier, 1999, pp. 1361–1374.

[5] A. Gullí, On two web IR boosting tools: Clustering and ranking, Ph.D. thesis, Dipartimento di Informatica, Universitá degli Studi di Pisa, Italy (May 2006).

[6] C. Carpineto, S. Osiński, G. Romano, D. Weiss, A survey of web clustering engines, ACM Computing Surveys 41 (3) (2009) 17:1–17:38.

[7] M. d'Aquin, M. Sabou, E. Motta, S. Angeletou, L. Gridinoc, V. Lopez, F. Zablith, What can be done with the semantic web? An overview Watson-based applications, in: Fifth Int. Workshop on Semantic Web Applications and Perspectives (SWAP'08), Rome, Italy, Vol. 426 of CEUR Workshop Proceedings, ISSN 1613-0073, online `http://ceur-ws.org/Vol-426/swap2008_submission_50.pdf`, CEUR-WS.org, 2008.

[8] S. Agrawal, K. Chakrabarti, S. Chaudhuri, V. Ganti, A. C. Konig, D. Xin, Exploiting web search engines to search structured databases, in: 18th Int. World Wide Web Conference (WWW'09), Madrid, Spain, ACM, 2009, pp. 501–510.

[9] O. Zamir, O. Etzioni, Web document clustering: a feasibility demonstration, in: 21st Int. Conf. on Research and Development in Information Retrieval (SIGIR'98), Melbourne, Australia, ACM, 1998, pp. 46–54.

[10] H.-J. Zeng, Q.-C. He, Z. Chen, W.-Y. Ma, J. Ma, Learning to cluster web search results, in: 27th Int. Conf. on Research and Development in Information Retrieval (SIGIR'04), Sheffield, United Kingdom, ACM, 2004, pp. 210–217.

[11] Y. Wang, M. Kitsuregawa, Link based clustering of web search results, in: Second Int. Conf. on Advances in Web-Age Information Management (WAIM'01), Xi'an, China, Vol. 2118 of Lecture Notes in Computer Science (LNCS), Springer, 2001, pp. 225–236.

[12] D. Zhang, Y. Dong, Semantic, hierarchical, online clustering of web search results, in: Sixth Asia Pacific Web Conference (APWeb'04),

Hangzhou, China, Vol. 3007 of Lecture Notes in Computer Science (LNCS), Springer, 2004, pp. 69–78.

[13] M. d'Aquin, C. Baldassarre, L. Gridinoc, S. Angeletou, M. Sabou, E. Motta, Characterizing knowledge on the semantic web with Watson, in: Fifth Int. Workshop on Evaluation of Ontologies and Ontology-based Tools (EON'07), Busan, Korea, 2007, pp. 1–10.

[14] G. Miller, WordNet: A lexical database for English, Communications of the ACM 38 (11) (1995) 39–41.

[15] R. Benassi, S. Bergamaschi, A. Fergnani, D. Miselli, Extending a lexicon ontology for intelligent information integration, in: 16th European Conf. on Artificial Intelligence (ECAI'04), Valencia, Spain, IOS Press, 2004, pp. 278–282.

[16] L. Ding, T. Finin, A. Joshi, R. Pan, R. S. Cost, Y. Peng, P. Reddivari, V. Doshi, J. Sachs, Swoogle: a search and metadata engine for the semantic web, in: 13th ACM Int. Conf. on Information and Knowledge Management (CIKM'04), Washington D.C., USA, ACM, 2004, pp. 652–659.

[17] E. Oren, R. Delbru, M. Catasta, R. Cyganiak, H. Stenzhorn, G. Tummarello, Sindice.com: a document-oriented lookup index for open linked data, International Journal of Metadata, Semantics and Ontologies 3 (1) (2008) 37–52.

[18] R. Trillo, J. Gracia, M. Espinoza, E. Mena, Discovering the semantics of user keywords, Journal on Universal Computer Science (JUCS). Special Issue: Ontologies and their Applications 13 (12) (2007) 1908–1935.

[19] J. Gracia, M. d'Aquin, E. Mena, Large scale integration of senses for the semantic web, in: 18th Int. World Wide Web Conference (WWW'09), Madrid, Spain, ACM, 2009, pp. 611–620.

[20] S. K. Ala, N. M. Kavi, Significance of syntactic features for word sense disambiguation, in: Fourth Int. Conf. on Advances in Natural Language Processing (EsTAL'04), Alicante, Spain, Vol. 3230 of Lecture Notes in Computer Science (LNCS), Springer, 2004, pp. 340–348.

[21] S. de Lin, K. Verspoor, A semantics-enhanced language model for unsupervised word sense disambiguation, in: Ninth Int. Conf. on Computational Linguistics and Intelligent Text Processing (CICLing'08), Haifa, Israel, Vol. 4919 of Lecture Notes in Computer Science (LNCS), Springer, 2008, pp. 287–298.

[22] D. Yarowsky, One sense per collocation, in: Second ARPA Workshop on Human Language Technology (HLT'93), Plainsboro, New Jersey, USA, Association for Computational Linguistics, 1993, pp. 266–271.

[23] I. P. Klapaftis, S. Manandhar, Word sense induction using graphs of collocations, in: 18th European Conf. on Artificial Intelligence (ECAI'08), Patras, Greece, IOS Press, 2008, pp. 298–302.

[24] L. Po, Automatic lexical annotation: an effective technique for dynamic data integration, PhD in Computer Science, Doctorate School of Information and Communication Technologies, University of Modena e Reggio Emilia, Italy (2009).

[25] J. Gracia, E. Mena, Web-based measure of semantic relatedness, in: Ninth Int. Conf. on Web Information Systems Engineering (WISE'08), Auckland, New Zealand, Vol. 5175 of Lecture Notes in Computer Science (LNCS), Springer, 2008, pp. 136–150.

[26] L. Po, S. Sorrentino, S. Bergamaschi, D. Beneventano, Lexical knowledge extraction: an effective approach to schema and ontology matching, in: 10th European Conf. on Knowledge Management (ECKM'09), Vicenza, Italy, 2009, pp. 617–626.

[27] E. Agirre, P. Edmonds (Eds.), Word Sense Disambiguation: Algorithms and Applications, Springer, 2007.

[28] P. Resnik, Selection and information: A class-based approach to lexical relationships, Ph.D. thesis, Department of Computer and Information Science, University of Pennsylvania, Pennsylvania, USA (December 1993).

[29] R. Asleson, N. T. Schutta, Foundations of Ajax, Apress, 2005.

[30] T. Hao, Z. Lu, S. Wang, T. Zou, S. GU, L. Wenyin, Categorizing and ranking search engine's results by semantic similarity, in: Second

Int. Conf. on Ubiquitous Information Management and Communication (ICUIMC'08), Suwon, Korea, ACM, 2008, pp. 284–288.

[31] R. Hemayati, W. Meng, C. T. Yu, Semantic-based grouping of search engine results using WordNet, in: Eight Int. Conf. on Web-Age Information Management (WAIM'07), Huang Shan, China, Vol. 4505 of Lecture Notes in Computer Science (LNCS), Springer, 2007, pp. 678–686.

[32] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, S. Hellmann, DBpedia – a crystallization point for the web of data, Journal of Web Semantics: Science, Services and Agents on the World Wide Web 7 (3) (2009) 154–165.

[33] C. Matuszek, J. Cabral, M. Witbrock, J. DeOliveira, An introduction to the syntax and content of Cyc, in: AAAI Spring Symposium on Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering, Stanford, California, USA, 2006.

[34] F. M. Suchanek, G. Kasneci, G. Weikum, YAGO: A large ontology from Wikipedia and WordNet, Journal of Web Semantics: Science, Services and Agents on the World Wide Web 6 (3) (2008) 203–217.

[35] M. A. Hearst, J. O. Pedersen, Reexamining the cluster hypothesis: Scatter/Gather on retrieval results, in: 19th Int. Conf. on Research and Development in Information Retrieval (SIGIR'96), Zurich, Switzerland, ACM, 1996, pp. 76–84.

[36] Y. Zhao, G. Karypis, Hierarchical clustering algorithms for document datasets, Data Mining and Knowledge Discovery 10 (2) (2005) 141–168.

[37] F. Sebastiani, Machine learning in automated text categorization, ACM Computing Surveys 34 (1) (2002) 1–47.

[38] W. M. Kules, Supporting exploratory web search with meaningful and stable categorized overviews, Ph.D. thesis, University of Maryland at College Park, Maryland, USA (April 2006).

[39] X. Qi, B. D. Davison, Web page classification: Features and algorithms, ACM Computing Surveys 41 (2) (2009) 12:1–12:31.

[40] T. H. Haveliwala, A. Gionis, P. Indyk, Scalable techniques for clustering the web, in: Third Int. Workshop on the Web and Databases (WebDB'00), Dallas, Texas, USA, 2000, pp. 129–134.

[41] Y. S. Maarek, R. Fagin, I. Z. Ben-Shaul, D. Pelleg, Ephemeral document clustering for web applications, Tech. Rep. RJ 10186, IBM Research (2000).

[42] A. Tombros, R. Villa, C. J. V. Rijsbergen, The effectiveness of query-specific hierarchic clustering in information retrieval, Information Processing and Management: an International Journal 38 (4) (2002) 559–582.

[43] F. Geraci, M. Pellegrini, P. Pisati, F. Sebastiani, A scalable algorithm for high-quality clustering of web snippets, in: 21st Annual ACM Symposium on Applied Computing (SAC'06), Dijon, France, ACM, 2006, pp. 1058–1062.

[44] F. Giannotti, M. Nanni, D. Pedreschi, F. Samaritani, WebCat: Automatic categorization of web search results, in: 11th Italian Symposium on Advanced Database Systems (SEBD'03), Cetraro, Italy, Rubettino Editore, 2003, pp. 507–518.

[45] P. Ferragina, A. Gulli, A personalized search engine based on web-snippet hierarchical clustering, Software Practice & Experience 38 (2) (2008) 189–225.

[46] S. Osiński, An algorithm for clustering of web search results, Master's thesis, Poznań University of Technology, Poland (June 2003).

[47] X. Wang, M. Bramer, Exploring web search results clustering, in: 26th SGAI Int. Conf. on Innovative Techniques and Applications of Artificial Intelligence (AI'06), Cambridge, United Kingdom, Springer, 2006, pp. 393–397.

[48] F. Geraci, M. Pellegrini, M. Maggini, F. Sebastiani, Cluster generation and cluster labelling for web snippets: A fast and accurate hierarchical solution, in: 13th Symposium on String Processing and Information Retrieval (SPIRE'06), Glasgow, United Kingdom, Vol. 4209 of Lecture Notes in Computer Science (LNCS), Springer, 2006, pp. 25–36.

[49] S. Bergamaschi, L. Po, S. Sorrentino, Automatic annotation for mapping discovery in data integration systems, in: 16th Italian Symposium on Advanced Database Systems (SEBD'08), Mondello, Italy, 2008, pp. 334–341.

[50] A. M. Buckeridge, R. F. E. Sutcliffe, Disambiguating noun compounds with latent semantic indexing, in: Second International Workshop on Computational Terminology (COMPUTERM'02), within the 19th International Conference On Computational Linguistics (COLING'02), Taipei, Taiwan, Vol. 14, Association for Computational Linguistics, 2002, pp. 1–7.

[51] S. N. Kim, T. Baldwin, Disambiguating noun compounds, in: 22nd National Conf. on Artificial Intelligence (AAAI'07), Vancouver, British Columbia, Canada, AAAI Press, 2007, pp. 901–906.

[52] D. Nadeau, S. Sekine, A survey of named entity recognition and classification, Lingvisticae Investigationes 30 (1) (2007) 3–26.

[53] S. Sorrentino, S. Bergamaschi, M. Gawinecki, L. Po, Schema normalization for improving schema matching, in: 28th Int. Conf. on Conceptual Modeling (ER'09), Gramado, Brazil, Vol. 5829 of Lecture Notes in Computer Science (LNCS), Springer, 2009, pp. 280–293.

[54] F. Guerra, S. Bergamaschi, M. Orsini, A. S. 0002, C. Sartori, Keymantic: A keyword-based search engine using structural knowledge, in: 11th Int. Conf. on Enterprise Information Systems (ICEIS'09), Milan, Italy, 2009, pp. 241–246.

[55] P. A. Bernstein, L. M. Haas, Information integration in the enterprise, Commununications of the ACM 51 (9) (2008) 72–79.

[56] P. M. V. Rudi L. Cilibrasi, The Google similarity distance, IEEE Transactions on Knowledge and Data Engineering (2007) 370–383.

[57] G. Shafer, A Mathematical Theory of Evidence, Princeton University Press, 1976.

[58] S. Parsons, A. Hunter, A review of uncertainty handling formalisms, in: Applications of Uncertainty Formalisms, Vol. 1455 of Lecture Notes in Computer Science (LNCS), Springer, 1998, pp. 8–37.

[59] J. Cowie, J. Guthrie, L. Guthrie, Lexical disambiguation using simulated annealing, in: Workshop on Speech and Natural Language (HLT'92), Harriman, New York, USA, Association for Computational Linguistics, 1992, pp. 238–242.

[60] D. Beneventano, F. Guerra, M. Orsini, L. Po, A. Sala, M. D. Gioia, M. Comerio, F. de Paoli, A. Maurino, M. Palmonari, C. Gennaro, F. Sebastiani, A. Turati, D. Cerizza, I. Celino, F. Corcoglioniti, Detailed design for building semantic peer, Networked Peers for Business, Deliverable D.2.1, Final Version, available at `http://www.dbgroup.unimo.it/publication/d2_1.pdf` (2008) 52–57.

[61] A. M. Gliozzo, C. Giuliano, C. Strapparava, Domain kernels for word sense disambiguation, in: 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05), Ann Arbor, Michigan, USA, Association for Computational Linguistics, 2005, pp. 403–410.

[62] M. E. Lesk, Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone, in: Fifth Int. Conf. on Systems Documentation (SIGDOC'86), Toronto, Canada, ACM, 1986, pp. 24–26.