

Improving Web Search Results for Homonyms by Suggesting Completions from an Ontology

Tian Tian¹, James Geller¹, and Soon Ae Chun²

¹ New Jersey Institute of Technology,
Newark, NJ, USA
{tt25,geller}@njit.edu

² CSI, City University of New York
Staten Island, NY, USA
{soon.chun}@csi.cuny.edu

Abstract. The terms that users pass to search engines are often ambiguous, by referring to homonyms. The search results in these cases are a mixture of links to documents that refer to different meanings of the search terms. Current search engines provide suggested query completion terms as a dropdown list. However, such lists are not well organized, mixing completions for different meanings. In addition, the suggested search phrases are not discriminating enough. We propose an approach to suggesting well organized and visually separated search completions based on ontologies. In addition, our approach supports the use of negative terms to disambiguate the suggested completions in the list. We present an algorithm to generate the suggested search completion terms. We have developed musician and basketball player ontologies and an Ontology-Supported Web Search (OSWS) System for “famous people” that generates the suggested search term completions, based on these ontologies.

Keywords: Ontology-supported Web search, semantic search methods, homonyms, negative search terms, suggested completions.

1 Introduction

Users’ information needs in the digital era can be fulfilled by keyword-based search engines. Such search engines have become the universal catalogs for world-wide resources. Unlike the old library catalogs that are mostly searchable by fixed fields (e.g., by authors, titles, and keywords predefined by authors), modern Web search engines provide a flexible, easy way to express search terms. However, the search results are typically long lists of hits that contain many irrelevant links [1]. Past research has concentrated either on refining the search keywords or on sifting and filtering the search results, to improve the precision of the returned hit lists [1].

Search engines face an additional complication when a search term is a homonym (a keyword with multiple meanings or multiple references) and the user is not aware that there are several concepts for this term. She might not be aware of this homonymy at all, or it might escape her attention at the moment of performing the Web search. For example, when looking for information about former President George W.

Bush she might momentarily forget about President George H. W. Bush, the father of President George W. Bush. She would then get results about both of them, which is not what she desired.

When using a search engine to satisfy an information need about a homonymous concept, a user is faced with two kinds of problems. She might get an overwhelming number of responses about one homonym, especially if this meaning is more popular, while the second homonym with a less popular meaning that she might be really interested in is hidden in a snippet on a much later page of hits, returned by the search engine. This is the case with lopsided preferences in meanings. For instance, the “Michael Jackson” who is a singer is much more popular than the basketball player of the same name. Hence many more search results contain references to the singer. In this situation, the user is at least aware that the results she is getting are not about the basketball player that she has been looking for. When formulating the initial query, it escaped her attention that there are two concepts for her search term and that more information might be available on the Web about the homonym that she is not interested in. At this point, she needs to wade through pages of reported hits for the wrong Michael Jackson or append terms to her query that will exclude the unwanted homonym and re-execute the search. This constitutes a kind of feedback loop between the user and the search engine.

The situation is even worse if the user is completely unaware of the fact that the search term is a homonym with two (or more) references, and all results that appear on the first few pages of hits are to the “wrong” reference. For example, a user located in the New York area, who types “Penn Station” into Google will see many references to Penn Station in New York City (NYC) and some references to Penn Station in Newark. These two Penn Stations are separated by a 20 minute train ride. Unbeknownst to her, there is also a Penn Station in Philadelphia, Pennsylvania. However, a reference to the latter does not appear on the first page of search results.

Google has a popular feature of displaying up to ten suggested search term completions for a query while the user is typing. Typing “George Bush” leads to suggested completions such as games, wiki, jokes, billboard, etc. The only small hint that there are two Presidents George Bush is one suggested completion “sr” (meaning senior). Google’s suggestions are presumably based on the most common search terms entered by its millions of users.

Our goal in this research is to improve the mechanism of suggested search completions in two ways. First, the display of suggested search term completions should be categorized visually to make it clear that homonymous terms exist. For this, knowledge of the classes that terms belong to is necessary. This is the kind of knowledge normally contained in ontologies. Secondly, the knowledge in the ontologies should be used to increase the precision of results, by making the suggested completions as discriminating as possible. One tool for making Web searches more focused is to use negative search terms in addition to the normal “positive” search terms. Naturally, the suggested search completions should not be over-specified to the point that the search engine would not return any results. As we do not have access to the “most common search terms” collected by commercial search engines, we cannot use them to generate suggested completions. Instead we use ontologies both for creating the suggested completions and for providing the knowledge needed to visually categorize them.

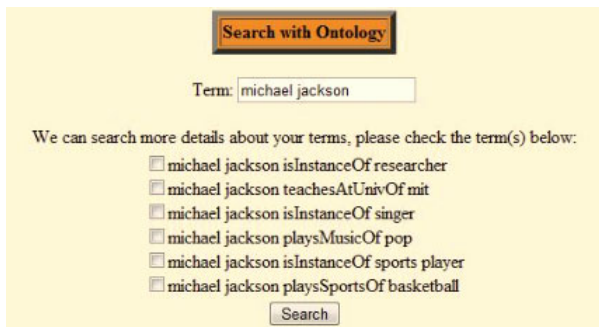
Including negative search terms in the search queries is a powerful tool for discriminating between wanted and unwanted results. In the past, negative search terms have not been used in suggested completions. We will discuss the generation of suggested completions with negative search terms and hint at the problems that arise out of this pursuit.

In Section 2 we present our previous work on ontology-supported Web search and explain some of its weaknesses. In Section 3.1 we describe an algorithm for generating suggested completions for homonymous search terms. Then, in Section 3.2, the Ontology-Supported Web Search (OSWS) System is introduced. Our approach to the use of negative search terms is discussed in Section 3.3. In Section 4, we briefly touch on the problem of ontology provisioning and present our musician and basketball player ontologies. Section 5 concludes the paper.

2 Background

In our previous research on an ontology-supported Web search system, the user was presented with a number of choices of additional search terms for her input. She could mark such terms as positive, i.e., they should be included in the Web search results, by clicking on associated check boxes (see Figure 1 and [2]). One problem with this approach was that users do not want to be bothered by (too many) questions. A more benign approach to eliciting additional information from a user can be seen in the use of suggested completions. While a user types in the first (few) word(s) of her search, the search engine displays up to ten suggested search completions, which will possibly describe the search that the user had in mind. These completions are presumably based on the observed frequencies of many searches of other search engine users [3]. While the user continues to type, the suggested completions change rapidly and are often limited to fewer than ten. Most major search engines have such a mechanism. Google calls them “query suggestions” appearing in the “search box” [3], Yahoo calls them “search assistant” [4], and Bing calls them “search suggestions” [5].

Another weakness of our approach in [2] was that it did not make use of the information that may be inferred by a form of closed-world assumption from the terms that



Search with Ontology

Term: michael jackson

We can search more details about your terms, please check the term(s) below:

- ☐ michael jackson isInstanceOf researcher
- ☐ michael jackson teachesAtUnivOf mit
- ☐ michael jackson isInstanceOf singer
- ☐ michael jackson playsMusicOf pop
- ☐ michael jackson isInstanceOf sports player
- ☐ michael jackson playsSportsOf basketball

Search

Fig. 1. Search screen example of previous ontology-supported Web search system

the user did *not* select with a check mark. According to the documentation of major search engines, the use of negative search words, marked with a minus sign before the word(s), constitutes a particularly powerful tool for discriminating between different results. Thus, we extend the approach in [2,6,7] in this paper by adding negative search terms.

Google's version of suggested completions has the problems described in Section 1. They do not reflect distinctions between different concepts that are expressed by the same word or the same multi-word term (homonyms). Suggested completions also do not appear to be optimized for discrimination between homonyms. Appending well chosen negative search words to a search term given by the user would result in improved discrimination between homonyms of that search term, if the appended words are characteristic for one of the homonymic senses. However, the use of too many negative search terms might exclude relevant results, i.e., the recall would suffer. It would be especially undesirable if the search is so over-specified that no results are reported at all.

The suggested additional search terms in [2] (see Figure 1) were derived from an ontology. For a given user input, all homonymous concepts were located in the ontology. Then choices of additional terms were generated by looking at neighboring concepts in the ontology. Thus "Michael Jackson" is categorized as a singer, or in more technical terms, Michael Jackson is an instance of the class "singer." Thus, a statement of this nature with a check box was suggested to the user. If the user checks this box, then the word "singer" was automatically appended to the Web search query before executing it.

Finding information about Michael Jackson the singer on the Web is clearly not a problem. There are millions of hits for this search term. However, when somebody is interested in Michael Jackson the basketball player, or in any one of the other over 20 Michael Jacksons that have achieved some kind of fame over the years, then the task of finding relevant information becomes much more difficult. Appending negative search words such as "-songs" and "-lyrics" makes this task easier, by excluding the most widely used homonym of Michael Jackson the singer.

3 Generating Ontology-Based Search Term Completions

3.1 An Algorithm for Suggested Completions with Positive Terms Only

The basic idea of generating suggested completions with *positive* search terms was not changed from [2], however the interface model was changed considerably, improving both on our previous work and on common search engines. The following pseudocode demonstrates the processing steps.

ALGORITHM *DISPLAY_SEARCH_SUGGESTIONS*

INPUT: *SEARCH_TERM*, *KNOWLEDGE_BASE*

OUTPUT: *Display of SEARCH_SUGGESTIONS*

BEGIN

NODE_COLLECTION = {}

FOR EACH *NODE* **IN** *KNOWLEDGE_BASE*

```

IF NODE contains SEARCH_TERM
    NODE_COLLECTION = NODE_COLLECTION U {NODE}
/* NODE_COLLECTION now contains all homonyms */
ITH_SUGGESTION = 1
IF size_of(NODE_COLLECTION) > 4
    NODE_COLLECTION = MOST_COMMON(NODE_COLLECTION)
/* NODE_COLLECTION now contains at most 4 homonyms */
FOR EACH NODE IN NODE_COLLECTION
    NEIGHBOR_LIST = {}
    FOR N IN NEIGHBORS_PLUS_GRANDPAR(NODE)
        /* We add one additional level in the IS-A
           hierarchy to the immediate neighbors. */
        NEIGHBOR_LIST = NEIGHBOR_LIST U {<REL, N>}
    /* Pairs of all neighbors and their connecting
       relationships are collected in a list. */
    PRIOR_LIST = PRIORITIZE(NEIGHBOR_LIST)
    /* Pairs with important relationships, such as
       IS-A are placed first in the list. */
    SEARCH_SUGGESTIONS[ITH_SUGGESTION] = PRIOR_LIST
    ITH_SUGGESTION++
SEARCH_SUGGESTIONS = LIMIT_SIZE(SEARCH_SUGGESTIONS)
/* At most 12 lines are displayed over all
   homonyms. */
DISPLAY_WITH_SEPARATORS(SEARCH_SUGGESTIONS)
/* Suggestions for each homonym are displayed,
   visually separated from each other. */
END

```

The algorithm *DISPLAY_SEARCH_SUGGESTIONS* uses the following sub-algorithms: *MOST_COMMON* returns at most four homonyms. The selection is done based on the number of hit counts for each homonym. These hit counts are recorded in the ontology during creation time. More details can be found in Section 3.2.

NEIGHBORS_PLUS_GRANDPAR returns for every instance in the ontology all neighboring nodes that are one link away from it, plus the “grand parent,” i.e., the IS-A parent of the class it is an instance of.

PRIORITIZE sorts the list of neighbors by importance. The importance is determined by the types of connecting relationships. Thus IS-A relationships to parent classes are considered more important than lateral semantic relationships. See Section 3.2 for more details on the importance of different relationship types. If several neighbors are connected by the same relationship type, then the order of the connected concepts is chosen arbitrarily.

LIMIT_SIZE controls the total size of the output. In order to avoid overloading the user with information and in order to achieve a behavior similar to existing search engines, the total number of search suggestions displayed is limited to at most 12. The number 12 is divisible by 2, 3, and 4, which makes it a good choice for 2, 3, or 4 homonyms.

Finally, the sub-algorithm *DISPLAY_WITH_SEPARATORS* creates the actual dropdown box that is shown to the user. It contains the computed search suggestions with appropriate separators to express the semantic distances between them.

Altogether the algorithm specifies the following behavior. A user types words of a search term into the search box. The algorithm locates nodes (classes or instances) in the stored ontologies that correspond to the input words. If only one node is located, then there is no problem with homonymy, at least according to the knowledge incorporated in the set of all loaded ontologies. On the other hand, if two (or more) nodes are located in the ontologies that match the user input, then additional processing is necessary.

For each located node, its neighbors¹ in the ontology network are retrieved, starting with the parent(s), if it is a class, or if it is an instance, the class that it is an instance of. Neighbors that are common to more than one sense (meaning) of the search term are eliminated, as they have no discriminatory power. The algorithm now appends subsets of these retrieved terms to the user terms to generate several suggested completions.

Knowledge from different domains is assumed to be stored in separate ontologies. However, when using this implemented knowledge, all ontologies are considered connected and combined into a single knowledge base.

Consider the following abstract example.

- The user types in two words *A B*, for example *A*=Michael and *B*=Jackson.
- The system identifies two concepts referred to as *A B*, let us call them *AB1* and *AB2*.
- *AB1* is an instance of *K*. *AB1* has a neighbor *L*.
- *AB2* is an instance of *M*. *AB2* has a neighbor *N*. The concepts *K*, *L*, *M* and *N* are distinct.
- The search engine generates the following suggested completions, three for *AB1* and three for *AB2*: *A B K*; *A B L*; *A B K L*; *A B M*; *A B N*; and *A B M N*.
- The total number of suggested completions is limited by a threshold and controlled by strict priorities in which order to select neighbors (Section 3.2).
- The suggested completions are presented to the user in a way that visually separates the *AB1* meaning from the *AB2* meaning, for example by using a bold line to separate them or by different background colors (see Section 3.2).

3.2 The Ontology-Supported Web Search (OSWS) System

The Ontology-Supported Web Search (OSWS) System for “famous people” provides search suggestions based on the user input, every time she types a new character. As seen in Figure 2, after the user completes the search term “Martina,” the system finds all the famous people in the knowledge base with “Martina” in their names. Additional background information about these famous people is extracted from the knowledge base for generating suggested completions. In this example, the tennis

¹ The immediate neighbors of a class are the following: parent classes (more general), child classes (more specific) and classes that are reachable from it by traversing a “semantic relationship.” The immediate neighbors of an instance are the class which the instance belongs to, and the object properties and the data type properties of the instance.

players Martina Hingis and Martina Navratilova and the singer Martina McBride are found. From the information related to these three famous people the suggested completions in the dropdown box are generated and displayed to the user.



Fig. 2. Interface of the OSWS System for search term "Martina"

In more detail, for each concept of a famous person of the same name, all immediate neighbors along with the connecting relationships are retrieved from the ontologies. In the OSWS System, the first proposed suggestion about a famous person is always based on the class (modeling the occupation) of the person, which defines the name of the domain that the person belongs to. For instance, Martina Hingis has the first suggested completion "Martina Hingis tennis player" and Martina McBride has the first suggested completion "Martina McBride singer."

Then the remaining suggestions about each famous person are constructed based on the knowledge retrieved from the ontologies. They may include the background information of a person like the date of birth and the place of birth, and sometimes the birth name. Besides, for musicians, the ontology stores the genres of music the artist performs. For sportsmen, the league and the team he or she belongs to are represented in the ontology. For instance, in Figure 2, from the suggested completions the user could learn that Martina McBride plays country music, adult contemporary music, and country pop music, which she may not have been aware of.

Different famous Martinas are separated by horizontal lines and background colors. This separation clearly expresses the fact that there are conceptual distances among the homonyms expressed by different sets of suggested completions. This makes it easier for the user to learn or remember that she is dealing with a homonym. Current search engines do not support such a separation. In fact, the visual display in the example expresses the fact that Martina Hingis is conceptually closer to Martina Navratilova (both tennis players) than to Martina McBride (the singer) by applying separating lines of different thickness.

Besides the separating lines, the background color design in the dropdown box also distinguishes famous people from different domains. In Figure 2, the suggestions for the two tennis players are generated by the system with a blue background, in contrast to the suggested completions of the singer that are displayed with a pink background.

Four preselected background colors are used for the at most four homonyms for which suggested completions may be displayed.

After the user chooses one suggestion that fits her search needs and clicks the “Google Search” button, she will be led to the result page of a “normal” Google search. Here we mimic the Google functionalities by having the “I’m Feeling Lucky” button, which will lead directly to the Web page with the highest Google ranking.

To avoid overwhelming the user with too many suggestions, and to stay close to the Google look and feel of the interface, we designed the system to show up to a maximum of 12 suggestions for a maximum of four famous people.

As noted in Section 3.1, there may be too many potential suggested continuations for one concept, and a selection process is required. The selection of lines for one homonym is achieved by assigning different priorities to different relationship types. For example, the IS-A link to the domain name (occupation) is considered to have the highest priority. For musicians, the genres of music they play have higher priorities than their dates of birth and places of birth. For basketball players, the team and league they play in are treated as more important than their birth information. Thus, we only show the high priority suggestions if there is more knowledge in an ontology than available space in the dropdown box. For example, in Figure 3, we show 12 suggestions in the search box by eliminating the date of birth and place of birth information of the singer Michael Jackson, since these have the lowest priorities.



Fig. 3. Interface of the OSWS System for search term "Michael Jackson"

If there are more than four homonyms (such as the over 20 Michael Jacksons) then the OSWS System selects up to four senses based on some criteria. There are two candidate approaches for this selection process. One possible selection criterion is the amount of information available in the ontologies about each sense. Thus, senses with a large amount of attached knowledge should be preferred over other senses. This is based on the pragmatic assumption that system implementers would not make the effort of including a large amount of information about a concept in an ontology if that concept is considered unimportant. However, this selection approach requires

mature ontologies covering many domains with rich knowledge. Unfortunately, such ontologies do not always exist, and it is still a big challenge to build them (Section 4).

In the absence of sufficiently complete ontologies a second approach is needed, which is the one used in the OSWS System. A possible criterion to select the most popular homonyms is by using the Google hit count estimates. We assume that people with higher hit count estimates are more popular and famous. For instance, the query “Michael Jackson singer” returns almost twice the number of Web pages than the query “Michael Jackson basketball.” Thus, Michael Jackson the singer should be preferred over the others. The representation of three homonymous “Michael Jacksons” in the knowledge base can be seen in Figure 4.

Classes in the figure are represented as boxes. Instances are shown as ellipses. IS-A links are drawn as arrows from the child class to the parent class. Dashed arrows connect instances to the classes that they are instances of. Finally, lines terminated by little black squares indicate semantic relationships other than IS-A and instance of relationships.

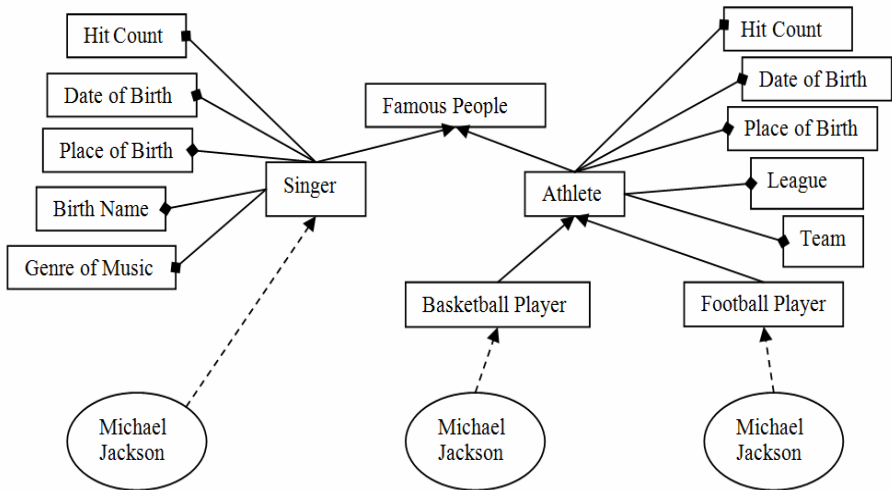


Fig. 4. Excerpt from the “famous people” knowledge base with homonym example “Michael Jackson”

We have collected the Google hit count estimates and assigned them to the appropriate instances of famous people while building the musician and basketball player ontologies. Thus, this information is available before the user starts with her search. However, this solution has several disadvantages. Hit counts are not stable. For example, after the singer Michael Jackson’s untimely death, the number of hits greatly increased. Thus the previously mentioned ontology-size-based criterion would be preferable.

The suggested completions in the search box of the OSWS System change dynamically after every single input character, just as in Google. The response time of

the OSWS System in the current implementation is near instantaneous, limited more by the typing speed of the user than by the response time of the system. The current ontologies in the system contain semantic information about more than 5000 musicians, more than 3000 basketball players and a sampling of sportsmen in other domains. Altogether, only three of the over 20 Michael Jacksons known to us are included in the ontologies. Clearly, with more and much larger ontologies in the OSWS knowledge base, the response time will degrade. More ontology building and experimentation is needed to investigate the effect of ontology size on response time.

3.3 Suggested Completions with Positive and Negative Search Words

In our previous work [2,6,7] we did not use negative search terms. The idea of using negative search terms is akin to mutual inhibition as it occurs in neural networks. If different neurons compete for achieving maximum activation, they inhibit neighboring neurons. This should be seen only as a metaphor, not as a technical model, as there are vast differences between the numeric approach of a neural network and the symbolic approach of an ontology. Based on this metaphor, if the user types in Michael Jackson and the ontology knows about Michael Jackson the singer and Michael Jackson the basketball player, then two useful suggested completions would be:

Michael Jackson Singer –Basketball
Michael Jackson Basketball –Singer

In both those suggested completions, we are using a bold font to indicate the words that have been entered by the user. Thus, neighbors of a node that are used as positive search terms for one homonym should be introduced as negative search terms for the other homonym. To our knowledge, none of the major existing search engines suggests completions with negative search terms to the users.

Many search engine users appear to be unfamiliar with the meaning of a minus sign (–) in front of a search word. Thus, suggesting a completion with a minus sign is syntactically unsatisfactory. Rather, the above completions need to appear as:

Michael Jackson Basketball [*but not*] Singer
Michael Jackson Singer [*but not*] Basketball

Using negative search words in suggested completions raises both conceptual and practical problems. One big practical problem that we have discovered in this research was that three major search engines do not process negative search terms as would be expected from their documentations or from a logical understanding of the meaning of “negative” words. This issue is, however, not central to the current paper. Results of this research are presented in [8].

4 Ontology Unde Venis?

(Ontology, from where do you come?) Probably the biggest problem with all ontology-based approaches is from where to take the necessary ontologies. Developing them in-house is time consuming and person-hour and/or budget intensive. Wide-scale ontology reuse has still not materialized, even though the Semantic Web [9], ontology search engines such as Swoogle [10] and ontology repositories [11,12] have

attempted to solve this problem. Many approaches to automatically generating or extending ontologies [2] have met with partial success but have also not reached the state of “shrink wrapped solutions.”

Outside of Medical Informatics, where huge terminological repositories are the norm, many ontologies are small. There appears to be little interest in building large, fact-oriented, regularly structured ontologies. Researchers prefer to focus on intricately structured, abstract “upper level ontologies” [13,14] or on rules and axioms. Building even one ontology with sufficient depth, breadth and domain coverage is a major challenge. Building ontologies just for the many existing categories of famous people in science, religion, art, history, politics, etc., and their many subcategories such as biology, chemistry, physics, computer science etc. in science, is a daunting task, even if we limit those ontologies mostly to simply-structured facts and instance categorizations.

Nevertheless, human intelligence relies on both rules and large numbers of simply-structured facts, including basic categorizations. For example, humans know about a large number of people how they are categorized, such that each one is known as a family member, friend, workmate, politician, sportsman or woman, somebody from “the history book,” a service provider (electrician, plumber, etc.), a teacher, student or classmate, etc., etc., or as completely unknown. We have, therefore, developed ontologies of musicians and basket ball players and are working on an ontology of sports stars to demonstrate the effectiveness of ontology-supported Web search.

The knowledge represented in these ontologies was mined from Wikipedia using a Web parsing program and stored in a temporary relational database. Not all information was available about every one of the over 5000 musicians.

A Protégé ontology was built, using the Protégé API, by extracting the mined data from the temporary database. One of the problems encountered in this process was the uniqueness requirement of Protégé. Thus, the city “Washington” and the state “Washington” could not both be represented by the same atom, and we had to append qualifiers to distinguish between them, in this case by appending the letters “DC” to the city.

To make our work available to the ontology community, we have submitted the musician ontology to Ontology Design Patterns (ODP) as an exemplary ontology: http://ontologydesignpatterns.org/wiki/Ontology:Musician_Ontology. In the future, we intend to publicize the sports star ontology in the same manner.

5 Conclusions

We have described an algorithm and its implementation in the OSWS System that improves both our own previous work on ontology-supported Web search and on the method that common search engines use for suggesting completions of user search terms for cases of homonyms. Our interface clearly separates between suggested completions for up to four homonymous concepts that fit the search terms that a user has already typed in. Furthermore, suggested completions in our interface may contain positive and negative search words. We have developed a knowledge base consisting of ontologies of musicians and basketball players used in the OSWS System and are currently working on a sports star ontology that covers additional popular disciplines.

Acknowledgments. We thank our students Christopher Ochs, Shrutee Singh, Mansi Pedgaonkar, Jalaj Asher and Anushri Mahajan for their work on the implementation of the OSWS System.

References

1. Radev, D.R., Fan, W., Zhang, Z.: WebInEssence: A Personalized Web-Based Multi-Document Summarization and Recommendation System. In: NAACL Workshop on Automatic Summarization, Pittsburgh, PA (2001)
2. An, Y., Chun, S., Huang, K., Geller, J.: Enriching Ontology for Deep Web Search. In: Bhowmick, S.S., Küng, J., Wagner, R. (eds.) DEXA 2008. LNCS, vol. 5181, pp. 73–80. Springer, Heidelberg (2008)
3. Google Query Suggestion,
<http://www.google.com/support/websearch/bin/answer.py?hl=en&answer=106230>
4. Yahoo Search Assistant,
<http://tools.search.yahoo.com/newsearch/searchassist.html>
5. Bing Search Suggestions,
http://help.live.com/help.aspx?project=wl_searchv1&querytype=keyword&query=tseggusotua&mkt=en-US
6. An, Y., Geller, J., Wu, Y., Chun, S.: Semantic Deep Web: Automatic Attribute Extraction from the Deep Web Data Sources. In: Proceedings of the 2007 ACM Symposium on Applied computing, pp. 1667–1672. ACM-SAC, Seoul (2007)
7. An, Y., Chun, S., Huang, K., Geller, J.: Assessment for Ontology-Supported Deep Web Search. In: 2008 10th IEEE Conference on E-Commerce Technology and the Fifth IEEE Conference on Enterprise Computing, E-Commerce and E-Services, pp. 382–388. IEEE Computer Society, Los Alamitos (2008)
8. Tian, T., Geller, J., Chun, S.A.: Predicting Web Search Hit Counts. In: WIC, Toronto, Canada, (2010) (accepted for Publication)
9. Lee, T.B., Hendler, J., Lassila, O.: The Semantic Web. Scientific American Magazine (2001)
10. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R.S.: Swoogle: A Search and Metadata Engine for the Semantic Web. In: Proceedings of the thirteenth ACM international conference on Information and knowledge management, pp. 652–659. ACM Press, New York (2004)
11. Ontology Design Patterns (ODP),
http://ontologydesignpatterns.org/wiki/Main_Page
12. Open Biological and Biomedical Ontologies (OBO),
<http://www.obofoundry.org/>
13. Niles, L., Pease, A.: Towards a standard upper ontology. In: Proceedings of the international conference on Formal Ontology in Information System, pp. 2–9. ACM, New York (2001)
14. Sowa, J.F.: Knowledge Representation: Logical, Philosophical, and Computational Foundations. Brooks Cole Publishing Co., Pacific Grove (2000)