Querying the Web: A Multiontology Disambiguation Method *

Jorge Gracia, Raquel Trillo, Mauricio Espinoza†, Eduardo Mena
IIS Department
University of Zaragoza
María de Luna 1
50018 Zaragoza, Spain
{jogracia,raqueltl,mespinoz,emena}@unizar.es

ABSTRACT

The lack of explicit semantics in the current Web can lead to ambiguity problems: for example, current search engines return unwanted information since they do not take into account the exact meaning given by user to the keywords used. Though disambiguation is a very well-known problem in Natural Language Processing and other domains, traditional methods are not flexible enough to work in a Web-based context.

In this paper we have identified some desirable properties that a Web-oriented disambiguation method should fulfill, and make a proposal according to them. The proposed method processes a set of related keywords in order to discover and extract their *implicit* semantics, obtaining their most suitable senses according to their context. The possible senses are extracted from the knowledge represented by a pool of ontologies available in the Web. This method applies an iterative disambiguation algorithm that uses a *semantic relatedness* measure based on Google frequencies. Our proposal makes *explicit* the semantics of keywords by means of ontology terms; this information can be used for different purposes, such as improving the search and retrieval of underlying relevant information.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Query formulation, Search process*; I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*Semantic networks*

General Terms

Algorithms, Measurement

Keywords

Semantic Web, multiontology-based disambiguation.

*This work is supported by the CICYT project TIN2004-07999-C02-02.
†Work supported by a Santander Central Hispano - University of Zaragoza grant.

ICWE'06, July 11-14, 2006, Palo Alto, California, USA.
ACM 1-59593-352-2/06/0007.

1. INTRODUCTION

The Web has experienced an enormous growth during last decade, which has allowed economical transactions, communications and access to huge amounts of resources and data across the Internet. Nevertheless this fast growth has been accompanied with an overload of unstructured and heterogeneous information which hinders interoperability in the Web. To solve this problem there are many research efforts in order to construct an extension of the current Web, the so-called *Semantic Web*, "in which information is given well-defined meaning, better enabling computers and people to work in cooperation" [1]. The development of the Semantic Web involves different efforts: definitions of new markup languages to represent semantic content, use of ontologies to make explicit semantics of data repositories, software agent technologies to achieve a more flexible and automatic interaction among systems in the Web, etc [1].

Concerning some problems of the current Web, the lack of explicit semantics can lead to ambiguity problems. Let us suppose that we use the keyword "book" in some web application (for example as input to a traditional web search engine). We are probably interested in just one sense of "book" (as "a number of pieces of paper, usually with words printed on them together and fixed inside a cover" or as "to arrange to have something or use something in a particular time such as a hotel room or a ticket"[1]), however web search engines return links to web pages where the word "book" appears, independently of the sense in which it is used in those pages. When a word appears within a particular context it is not difficult for a human to discover its intended meaning. For example, if a human observer reads the following set of keywords "book, writer, publication" he could conclude that the sense given to the word "book" in this context is "a number of pieces of paper, usually with words printed on them together and fixed inside a cover". This is the kind of reasoning we want web search engines to do. However the selection of the most suitable sense according to the context could be a difficult task to be performed automatically, as the different senses of keywords are not made explicit.

Word Sense Disambiguation techniques can be used to solve ambiguity problems. This is a well-known problem in the domain of Computational Linguistics or Natural Language Processing [19]: Disambiguation is the process of pick-

[1]http://wordnet.princeton.edu.

ing up the most suitable sense of a polysemous word according to a given context. However, in the Semantic Web domain, Entity Identification/Disambiguation is a much harder yet fundamental problem [18]. Disambiguation techniques for the Semantic Web may vary widely depending on the nature of the processed data. In particular, *this paper focuses on processing unstructured data (a set of user keywords) in order to discover and extract their implicit semantics.* We propose a disambiguation method that adapts some traditional disambiguation methods in order to generalize them to a Web context. The most remarkable differences with traditional methods are: 1) the independence with respect to a single lexical database, corpus or dictionary to provide candidate keyword senses, and 2) the use of a *semantic relatedness measure* based on Google, to consider *all* possible interpretations for a word according to the content of the Web. These advantages are analyzed in Section 2.

In summary, our disambiguation method receives a set of user keywords as input. Then it discovers a set of candidate senses for each keyword by consulting a pool of ontologies available in the Web; according to [6], ontologies are specifications of conceptualizations that provide a non-ambiguous vision of terms within them. When a keyword matches an ontology term, such a term provides the keyword with a candidate sense by means of its ontological context (its set of hypernyms and hyponyms). The next step is to estimate the *semantic relatedness* between each candidate sense of a keyword and the candidate senses of its neighbor keywords; an algorithm that assigns a weight to its possible keyword senses is applied. This estimation uses a measure based on Google to compute the degree of *semantic relatedness* between two senses. Finally, the most suitable sense is selected for each keyword. Therefore the *implicit* semantics in the keyword set is extracted and becomes *explicit* (as ontology terms). This unambiguous output can be used for different purposes, such as the construction of semantic queries to retrieve relevant data [16].

The rest of the paper is as follows. In Section 2 we identify some desirable properties for a Web-oriented disambiguation method and include an overview of the proposed system. In Section 3 we explain how to obtain the candidate senses of each keyword and in Section 4 our disambiguation process is detailed. In Section 5 we briefly explain our prototype and some initial experimental results. Section 6 reviews traditional disambiguation approaches and presents some related work. Finally, conclusions and future work appear in Section 7.

2. PROPOSAL FOR A WEB-ORIENTED DISAMBIGUATION METHOD

Some characteristics are common in traditional *dictionary-based* disambiguation techniques[2]: They process a set of related words (which constitute a specific context), they usually exploit a particular lexical database as WordNet [12], and they use a disambiguation algorithm which relies on a particular measure of *semantic similarity* or *semantic relatedness*. We use these concepts in the sense explained in [2]: *similarity* between two entities is associated with their likeness degree (e.g. "car" and "truck" are *similar* concepts); *semantic relatedness* measures how related two concepts are according to different types of relationships: meronymy, antonymy, or any kind of functional relationship or frequent association (e.g. "car" and "wheel" are *dissimilar* but *related* concepts).

In our opinion any disambiguation method proposed for a Web-based context must have a broad scope and a high flexibility. In the following we enumerate some characteristics which, from our point of view, are desirable for a Web-oriented disambiguation method. Notice that second and third are rather innovative features compared to traditional methods:

1. *Unsupervised method.* Among the systems that treat disambiguation problems there are two different methodologies [20]: 1) *Supervised learning methods* (systems are trained with examples of manually disambiguated words) and 2) *Unsupervised* or *dictionary based* methods (systems are based on a specific algorithm and do not need a training set but electronic dictionaries or similar resources). Although *supervised* methods give good results, their main problem is the lack of the large amount of semantically tagged corpora which is required to use such methods at a wide scale. This is known as the "knowledge acquisition bottleneck" [4]. So we consider *unsupervised* methods more suitable in a wide application context like the Web (although some supervised techniques could be added later to enrich given results).

2. *Independence of a single lexical resource to provide candidate keyword senses.* Although using a huge lexical database, such as WordNet, provides many benefits, a dependence on a single lexical resource is not desirable due to different reasons: We depend on its availability and some possible senses could not appear on it. For example, the term "UML" (a well-known acronym of Unified Modeling Language) does not appear in WorNet 2.0, but it can be found in other ontologies in the Web, as in Book.daml[3]. So we propose the use of a pool of ontologies (including lexical resources like WordNet) to discover the candidate senses of a keyword.

3. *Independence of a single ontology to compute semantic relatedness.* Many traditional disambiguation methods use semantic measures restricted to terms in WordNet [14, 19, 8] or they rely on a corpus to compute information content [14], so if a keyword cannot be found in those resources, then its semantic relatedness with respect to other terms cannot be computed. Nevertheless, almost any term we can imagine can be found in the Web. Therefore, we suggest taking into advantage of querying Google in order to provide semantic measures, as they do in [7]. As disadvantage, the quality of the semantic information extracted from the Web is not as high as the extracted from a corpus or thesaurus. We assume some imprecision inherent to the Web in order to achieve a better behavior of the disambiguation method.

4. *Low computational cost.* Some of the reviewed algorithms compute similarity (or semantic relatedness) between the senses of each word and *all* the senses

[2]An interesting review of disambiguation techniques and semantic measures can be found in [19].

[3]http://islab.hanyang.ac.kr/damls/Book.daml.

of the words in the context. This approach leads to a high computational cost (increase with the number of senses of the words to be processed). Due to the highly dynamic nature of the Web and the need of a fast response to the end user, similarity computation should be performed in a parallel and incremental way to avoid high computational cost.

In the following we describe our approach to disambiguation a set of keywords that fulfills the above features.

Overview of our system

The system receives as input a set of keywords to disambiguate, and its output is the most suitable sense for each keyword according to the context (we call them *semantic keywords*). In the following, we summarize the main steps of our approach (see Figure 1):

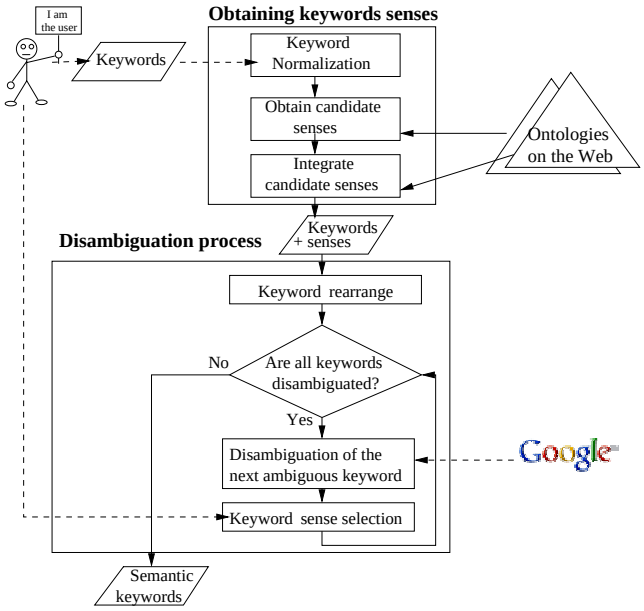

Figure 1: Discovering the sense of the keywords

1. *Obtaining candidate keyword senses*: The input is the list of keywords to disambiguate. First, they must be normalized in order to facilitate their comparison with ontology terms. Then, they are searched in a pool of ontologies available in the Web. The result is a set of possible senses (ontology terms and their hypernyms and hyponyms) for each keyword. Terms from different ontologies whose semantic similarity degree reaches a certain threshold are integrated as a single sense to avoid redundancy.

2. *Disambiguation process*: This process simulates the human behavior by extracting the semantics of a keyword from its context (the rest of user keywords). An iterative disambiguation algorithm ranks the possible senses of each keyword and the most suitable one is selected. This process finishes when all the keywords have been disambiguated. The output is the selected senses, expressed by means of ontology terms (semantic keywords). As we said before, they can be used for different purposes, such as the construction of formal queries.

These two steps will be detailed along in sections 3 and 4, respectively.

3. OBTAINING CANDIDATE SENSES

Before looking for the candidate senses of the keywords a *syntactic normalization* is performed. It is carried out by rewriting the keywords in lowercase, removing hyphens and other special characters, performing a morphological processing (plural names are transformed into single names, verbs into infinitives, etc.), and deleting stop terms. This process helps to compare keywords to ontology terms syntactically.

Then, it starts the process of *discovering candidate senses* for the set of normalized user keywords $K = \{k_1...k_n\}$. First, the normalized keywords are searched among terms in ontologies of pool $O = \{o_1...o_h\}$. We access to the ontologies by means of Swoogle [5], a system that indexes about 10,000 ontologies available on the Web. The result is a set of candidate senses for each keyword. A sense of a keyword k_i, denoted by s_{k_i}, is a tuple $s_{k_i} =< grph, descr, pop, syndgr >$ where *grph* is the ontological description of the sense by means of their hierarchical graph (of hyponyms, hypernyms and synonyms), *descr* is a description in natural language, if available, and *pop* (popularity) and *syndgr* (synonymy degree) are used when the sense is an integration of various ontology terms, as we will see later.

When matching terms are found in the ontologies, three types of terms could be returned: concepts, properties or values. They are treated separately due to their different semantic nature [11]. Therefore three list of possible senses are associated with each keyword k_i: $S^{concept}_{k_i}$, $S^{property}_{k_i}$ and $S^{value}_{k_i}$ to store the senses that k_i can play as concept, property, and value, respectively. We denote $S_{k_i} = S^{concept}_{k_i} \cup S^{property}_{k_i} \cup S^{value}_{k_i}$.

The main problem of a merely syntactic search among the ontology terms is the redundancy of results. For example, the term "star" has twelve senses as a concept in WordNet 2.0 ontology, but also appears as a concept in Mid-Level Ontology (MILO)[4] as a subclass of "Astronomical Body", with the comment "star is the class of hot gaseous astronomical bodies", which could be redundant with its first sense in WordNet: "(astronomy) a celestial body of hot gases that radiates energy derived from thermonuclear reactions in the interior". To solve this redundancy it is required an estimation of the *semantic similarity* degree between these terms from different ontologies, in order to conclude if they represent the same concept or not. A possible way is to estimate the *synonymy probability* using a statistical approach, as shown in [16]: the system discovers when two terms are equivalent by studying a sample of the hypernym/hyponym relationships among terms, incrementally. Other semantic similarity measures across different ontologies can be used, as the one presented in [15]. If *semantic similarity* degree among terms in different ontologies is high enough (above certain threshold), their senses are *integrated* and treated as a single candidate sense, with an associate value for its synonymy degree (*syndgr*) and popularity (*pop*, i.e., the num-

[4]http://reliant.teknowledge.com/DAML/Mid-level-ontology.owl.

ber of ontologies that participate in the sense integration). Otherwise they are considered as different senses.

If k_i is not found in O, then no sense can be obtained for that keyword using the available knowledge, but it can still be used to disambiguate the other keywords in K, as we will see in the next section.

4. DISAMBIGUATION PROCESS

Once the candidate keyword senses are obtained, a disambiguation process is initiated in order to discover the semantics implicit in the set K of user keywords. Every $s_{k_i} \in S_{k_i}$ is compared, using a *semantic relatedness* measure, with the senses of unambiguous keywords (monosemous or previously disambiguated ones) and with the senses of the other ambiguous keywords. The contribution of the other senses in the context is taken into account in order to assign a weight W_{k_i} to the evaluated sense.

Instead of using a *semantic similarity* measure (like used in Section 3) to weight a sense in its context, we consider a *semantic relatedness* measure more suitable than a *semantic similarity* one. Our *semantic relatedness* measure is based on the *Normalized Google Distance* [3] (see Subsection 4.2).

The main steps of disambiguation process are:

1. *Keywords rearrange.* Keywords in K are sorted by the following heuristic rule: The words not disambiguated yet (monosemous or without available senses in the pool of ontologies), are placed at the beginning. Then, polysemous words are sorted by the number of meanings, decreasingly. As the processing is sequential, all the monosemous word have been considered when the first ambiguous word is processed. By disambiguating the highest polysemy first we improve the efficiency of the algorithm.

2. *Disambiguation of the next ambiguous keyword.* A weight $W_{s_{k_i}} \in [0-1]$ is calculated for each possible keyword sense s_{k_i} which is proportional to the probability of being the right sense according to the context (see subsections 4.1 and 4.2).

3. *Keyword sense selection.* Once every sense of a keyword is weighted, the system allows two operation modes:

   (a) *Semi-Automatic Mode*: A list shows the possible senses sorted by the weight decreasingly (senses with a very low weight are filtered out as it is explained in the Subsection 4.1). The system proposes the most relevant sense to be selected but the user can change this default selection. This possible user intervention is justified because the context could not be enough to disambiguate the keyword according to the user interpretation.

   (b) *Automatic Mode*: It automatically selects the sense with the highest weight as the right one.

   After selecting the right sense for a keyword, the system continues disambiguating the following keyword. Steps 2 and 3 are repeated until all the keywords are disambiguated.

The following two subsections, detail how the algorithm works and how the semantic relatedness is measured, respectively.

4.1 The Disambiguation Algorithm

The disambiguation algorithm is iterative. The m-th iteration computes the weights of the senses of $k_m \in K$, as it is shown in Figure 2 (it is supposed that the algorithm has already been run over the previous $m-1$ keywords).

```
for k ← 1 to <number of senses of keyword M-th> do
  currentSense_k ← sense to be weighted;
  /* To look upon previous keywords already disambiguated */
  for N ← 1 to M-1 do
      if keyword N-th is in pool of ontologies then
          previousSense ← disambiguated sense of keyword N-th;
          tempWeight_k ← tempWeight_k + relGoogle(currentSense_k, previousSense)
      else
          tempWeight_k ← tempWeight_k + relGoogle(currentSense_k, keyword N-th)
      end_if
  end_for;
  /* To look upon later keywords, not disambiguated yet */
  for N ← M + 1 to <total number of keywords> do
      for d ← 1 to <number of senses of keyword N> do
          laterSense_d ← sense d of keyword N;
          rel_k[d] ← relGoogle(currentSense_k, laterSense_d)
      end_for;
      tempWeight_k ← tempWeight_k + maximum (rel_k[d])
  end_for;
  /* Normalization of weight */
  normWeight_k ← tempWeight_k /(<total number of keywords> -1);
  /* Add other possible factors to sense weight */
  weight_k ← f_other otherWeight_k + (1 – f_other ) normWeight_k;
end_for
```

Figure 2: Disambiguation algorithm pseudocode

The expression $relGoogle(x, y)$ in the algorithm measures the *Google-based semantic relatedness* between the search terms x and y. To apply this function to two keyword senses, we must build a convenient string to perform the necessary queries on Google (using for that the available ontological information that characterizes each sense). This process is detailed in Section 4.2.

The disambiguation algorithm is iterative in order to avoid a high computational cost. It prevents to cross semantic measures among *all* the senses of different keywords in the disambiguation of a single keyword. The system considers *only* selected senses of the previously disambiguated keywords and all the senses of no-disambiguated ones. If there is a high number of keywords it is possible to limit the number of them that take part in the context establishing a *window*.

Besides *semantic relatedness*, some additional information can also be taken into account to compute the final weight (see Figure 2). In particular we consider, if available, the information about the frequency of use of the sense, according to the number of times that it is tagged in various semantic concordance texts[5]. This helps to solve conflicts in case of equally weighted senses, and it is useful when the context is small (very few keywords) or there is not context (only one keyword). To add this information to the final weight we use a light-weight correction factor f_{other}, which decreases with the number of disambiguated keywords and the number of total keywords.

When the disambiguation algorithm finishes, all the possible senses of a keyword have been ranked. Then, the system applies a *less relevant filter* in order to omit the senses with the lowest weights. This feature is specially interesting if many senses are discovered for a keyword, making easier the selection task if user intervention is required. This filter is performed in three steps:

[5]For example, WordNet 2.0 stores this kind of information.

1. The higher weight W_{max} is selected.

2. The linear function $f(w_i) = (w_i/W_{max})$ is computed for all the sense weights w_i.

3. A threshold u in the range [0,1] is applied, omitting those senses with $f(w_i) <= u$.

In our prototype we use a default filter with threshold $u = 0.4$ but it is configurable. If the system does not discover the intended sense, the threshold can be lowered or the filter could be disabled before trying again.

4.2 Semantic Relatedness Between Senses

In order to measure the semantic relatedness between any two terms, we consider the hypothesis done in [3]: the relative frequency whereupon two terms appear in the Web within the same documents gives an idea of their semantic distance. The authors define a semantic measure based on this intuitive idea called *Normalized Google Distance* [3]:

$$NGD(x,y) = \frac{max\{\log f(x), \log f(y)\} - \log f(x,y)}{\log M - max\{\log f(x), \log f(y)\}} \quad (1)$$

where x and y are search terms, M is the total number of pages indexed by Google and $f(x)$ is the number of pages where x appears in Google. The smaller the value of NGD is, the more related the terms are semantically. For example, the NGD("red", "blue") = 0.25 and NGD("blue", "October") = 0.48 (further semantically). This definition has not some of the mathematical properties of distance, for example, it fails the triangular inequality property. However, it provides a relative measure of how far two terms are semantically, which is very suitable for our purposes.

Although most of NGD values come between 0 and 1, it ranges from 0 to ∞. Nevertheless, we need a *bounded* measure that increases according to the degree of semantic relatedness (instead of the distance). So we look for a function $fngd(x)$ which transforms the *Normalized Google Distance* and fulfills the following properties:

1. $fngd(x) : [0, \infty) \rightarrow (0, 1]$
2. $\lim_{x \rightarrow \infty} fngd(x) = 0$
3. $fngd(0) = 1$
4. If $x1 > x2$ then $fngd(x1) < fngd(x2)$

The function considered is $fngd(x) = e^{-2x}$, so we define the *Google-based semantic relatedness* between two search terms x and y as:

$$relGoogle(x,y) = e^{-2NGD(x,y)} \quad (2)$$

Initially the function (2) can be applied to *any* pair of search terms indexed by Google[6]. Nevertheless in our disambiguation algorithm we use it to discover how much semantically related are a pair of senses (or a sense and a keyword) as we see in Figure 2, so we must convert keyword senses into convenient search terms (strings) which permits to perform queries on Google. These strings are built using part of the available semantics that characterize the considered keyword senses s_{k_i} and s_{k_j}. It is extracted from the ontology or ontologies where the keyword were found, and lets us restrict the *semantic field* where the senses can be located on the Web.

For example, the keyword "book" in its sense "to reserve something" can be characterized by its synonyms ("reserve" OR "hold" OR "book"), or by the synonyms of its direct hypernyms ("request" OR "bespeak" OR "call for" OR "quest"). We consider these two levels of characterization: *Level 0* the term itself and its synonyms, and *Level 1* its *father* terms and their synonyms. Consequently two different search terms can be built to characterize a single sense s_{k_i}: $s_{k_i}^{lev0}$ and $s_{k_i}^{lev1}$. We calculate the semantic relatedness between senses computing each level separately and then combining them with certain weights.

$$relGoogle(s_{k_i}, s_{k_j}) =$$
$$w_0 \cdot relGoogle(s_{k_i}^{lev0}, s_{k_j}^{lev0}) + w_1 \cdot relGoogle(s_{k_i}^{lev1}, s_{k_j}^{lev1})$$

Our prototype uses a heuristic weight of $w_0 = w_1 = 0.5$. The construction of $s_{k_i}^{lev0}$ is simple for any kind of term. It is a conjunction of synonyms, if they are available:

$$synonym_1 + \ OR \ + synonym_2 + \ OR \ + ... + synonym_n$$

Nevertheless, the construction of $s_{k_i}^{lev1}$ depends on the type of term considered:

1. $s_{k_i} \in S_{k_i}^{concept}$: We characterize $s_{k_i}^{lev1}$ by their direct hypernyms, as the conjunction of the disjunction of synonyms of each direct hypernym in the correspondent ontology:

   $$(synonym_{11} + \ OR \ + synonym_{12} + \ OR \ + ...) + \ AND$$
   $$+ (synonym_{21} + \ OR \ + synonym_{22} + \ OR \ + ...) + \ AND$$
   $$+ ... + (synonym_{N1} + \ OR \ + synonym_{N2} + \ OR \ + ...)$$

   Where $synonym_{ij}$ is the $j-th$ synonym of the hypernym $i-th$.

2. $s_{k_i} \in S_{k_i}^{property}$: Although properties has their own $is-a$ hierarchy in ontologies, it is not usual in most of them. Therefore we adopt the *domain* of the property (that is a concept) to characterize the *level 1* instead of its fathers in the hierarchy of properties. We built their search term as a conjunction of the synonyms of the domain.

3. $s_{k_i} \in S_{k_i}^{value}$: If the value is an instance of a concept, we built the search term for $s_{k_i}^{lev1}$ as a conjunction of the synonyms of the associated concept. If it is an instance of a property, we use the conjunction of the synonyms of its domain.

For example, let us suppose that the keywords "book" and "film" are in the same disambiguation context. One of the calculations performed by the disambiguator could be to compare semantically a sense of "book", obtained as a property from Bibliographic-ont ontology[7], with the 4th sense of "film" as a WordNet concept. We should calculate (we denote these senses as $book_{bib}$ and $film_{WN}$):

[6]Note that this method does not depend on Google intrinsically, and could be adapted to any other search engine.

[7]http://orlando.drc.com/daml/ontology/Bibliographic/3.1/-Bibliographic-ont.

$$relGoogle(book_{bib},\ film_{WN}) =$$
$$w_0 \cdot relGoogle(\text{“}book\text{”},\ \text{“}film\text{”}\ OR\ \text{“}plastic\ film\text{”}\ ) +$$
$$w_1 \cdot relGoogle(\text{“}bibliographic\text{”},(\ \text{“}sheet\text{”}\ OR\ \text{“}flat\ solid\text{”})$$
$$AND\ (\text{“}wrapping\text{”}\ OR\ \text{“}wrap\text{”}\ OR\ \text{“}wrapper\text{”}\ )) = 0.3$$

If a keyword introduced by the user does not exist in the pool of ontologies, we do not know its senses so it cannot be characterized. In this case we use directly the written expression of the keyword as characterization to be compared to $s_{k_i}^{lev0}$ and $s_{k_i}^{lev1}$ of other keyword senses. This keyword cannot be disambiguated (because its senses are not known) but, if it appears in Google, it will influence the disambiguation of the other keywords, as it is deducted from Figure 2.

5. EXPERIMENTAL EVALUATION

We have developed a prototype in Java to test our disambiguation process. Our system uses a lot of different ontologies accessed by means of Swoogle, including WordNet.

Subsection 5.1 shows some initial results about testing the Google-based semantic relatedness measure. Subsection 5.2 illustrates, with examples, the scope and capabilities of our disambiguation method.

5.1 Testing the Google-based Semantic Relatedness

We have done an assembly of tests in order to evaluate if the use of a *semantic relatedness* measure based on NGD is acceptable, comparing the Google based measure to the judgment of human observers.

This experiment was inspired in the Miller and Charles's one [13]. A group of 20 human observers gave us their judgment about the *semantic relatedness* degree of a group of 30 word pairs. They were not asked about *similarity* degree, as in Miller and Charles's experiment, but about the degree of *semantic relatedness*. The words in the experiment were nouns extracted from WordNet, in order to allow comparisons with WordNet based semantic measures (we do not include the considered word pairs here due to space limitations). Then, the Google based semantic relatedness and other traditional measures[8] were calculated between the same word pairs. Spearman correlation coefficients were computed between computer measures and human ranking (see Table 1). They showed a positive correlation between the Google based measure and human judgment, higher than the results obtained for path length and context vector measures, and similar to other well-established measures as Adapted Lesk's [19] or Resnik's [14].

Table 1: Correlation with human judgment

Measure	*Value*
Resnik	0,58
Google Based	0,56
Adapted Lesk	0,51
Path Length	0,36
Context vector	0,28

Although we do not consider this initial experiment conclusive, it showed us a good behavior of Google based semantic relatedness which (in addition to other benefits, as WordNet independence) justifies to continue using it.

5.2 Illustrating the Disambiguation Algorithm

In this section we apply our disambiguation algorithm to particular examples in order to show the scope of our method and some of their advantages.

Let us suppose that a user wants to find some online tutorials about different programming languages. The initial keyword set is: {*tutorials about computer languages online*}. A Google[9] search returns 13,900,000 hits and among the first results some pages about courses to learn languages (French lessons, English grammar, etc.) can be found. These unwanted results are caused by the imprecision inherent to that query. Disambiguation of polysemous keywords could help to improve the final retrieved information. After the normalization process the keywords become: {*tutorial computer language online*}.

WordNet-based methods. If disambiguation is done by means of any existing WordNet-based method, some immediate problems arise: 1) "online" does not appear in WordNet[10], 2) "tutorial" appears as noun in WordNet: "session of intensive tuition given by a tutor to an individual or to a small number of students" but it could not be the intended sense. Therefore "online" can not be considered as context in the disambiguation process, and "tutorial" does not have a suitable candidate sense, so any WordNet disambiguation process would be unsuccessful or incomplete.

Proposed method. The previous problems can be solved by our disambiguation method: even if "online" does not appear in the ontologies it can still contribute in the disambiguation process, because it is indexed by Google and the function (2) can be computed and used to disambiguate another keywords (Figure 2). Nevertheless "online" is retrieved from other ontologies in our system. For example it appears in iso-metadata[11] as a property, or in course ontology[12]. The keyword "tutorial" is also obtained from other ontologies. In swpg.rdf[13] it appears with a sense closer to the user intended meaning (subclass of "instructional material"). The Table 2 is a summary of the senses returned in our ontology pool.

Table 2: Number and type of candidate senses

	Keyword	*Type*	*#Senses*
1	language	Concept	6
		Property	1
2	computer	Concept	4
3	tutorial	Concept	3
4	online	Concept	2
		Property	1

After the process of obtaining candidate senses, the keywords are rearranged (as they are shown in Table 2). Then, the disambiguation algorithm is run over the first ambiguous keyword: "language" to weight their seven possible senses. The highest score is for its third WordNet sense. The right

[8]They were computed by using WordNet::Similarity software. See http://marimba.d.umn.edu/cgi-bin/similarity.cgi where some additional information about the used measures is also available.

[9]http://www.google.com.

[10]However it appears as 'on-line' in WordNet2.0, but only as an adjective without belonging any hierarchy.

[11]http://loki.cae.drexel.edu/∼wbs/ontology/2004/02/iso-metadata.

[12]http://harth.org/2002/course.

[13]http://139.91.183.30:9090/RDF/VRP/Examples/SWPG.rdf.

Table 3: Disambiguated senses for {language computer tutorial online}

Keyword	*Ontology*	*Type*	*Level* 1	*Level* 0	*Description*	*Score*
language	WordNet	Concept	{Word, language unit, linguistic unit}	{language, terminology, nomenclature}	"a system of words used to name things in a particular discipline"	0.36
computer	WordNet	Concept	{Machine}	{computer, data processor, computing device, ...}	"a machine for performing calculations automatically"	0.38
tutorial	swpg.rdf	Concept	{Instructional material}	{tutorial}	"...connected to the contents of a Course and provide helpful tips in order to achieve certain goals"	0.13
online	course	Concept	{location}	{online}	-	0.49

Table 4: Disambiguated senses for {astronomy star planet}

Keyword	*Ontology*	*Type*	*Level* 1	*Level* 0	*Description*	*Score*
astronomy	WordNet	Concept	{physics}	{astronomy, uranology}	"the branch of physics that studies celestial bodies and the universe as a whole"	0.36
star	WordNet	Concept	{celestial body}	{star}	"(astronomy) a celestial body of hot gases..."	0.32
planet	WornNet	Concept	{celestial body}	{planet}	"any of the celestial bodies that revolve..."	0.35

sense is selected and the algorithm is repeated to disambiguate "computer", and the rest of the keywords. Table 3 shows final disambiguation results, the *semantic keywords*, which reflect the intended user meaning for all the keywords. This example was run in *semi-automatic* mode.

To conclude this section, we show in Table 4 the disambiguation result of a second example, with the keyword set: {*Astronomy star planet*}. Our prototype discovers 1, 10 and 4 senses respectively for them, and selects convenient senses for the keywords in their context. This example was run in *automatic* mode.

6. RELATED WORK

In this section we describe other approaches to disambiguate words and we compare them to our approach. Depending on how the *semantic similarity* or *semantic relatedness* is computed, three main techniques can be distinguished, within the *unsupervised* disambiguation methods:

1. *Measures based on glosses:* A gloss is a definition or explanation of a word in a dictionary. These methods are based on the glosses provided by an electronic dictionary. For example, the Lesk algorithm [9] disambiguate a word by comparing its possible definitions to the definitions of the words in its context. The common words between the different glosses of the word and the glosses in the context are counted and these numbers provide a weight to the different senses. This idea is the seed of a lot of current methods, one of the most remarkable is described in [19].

2. *Measures based on concept trees:* They are based on a tree or is-a hierarchy, like WordNet. A simple measure of this type is the length of the shorter path between two terms (number of intermediate edges) in the hierarchy: The less distance, the more similarity. Different measures based on path lengths has been developed, as in [8], with the inclusion of some correcting factors to improve this basic idea.

3. *Measures based on information content:* These algorithms are based on the idea of assigning a measure of specificity to each concept in a hierarchy: The more information content the less specificity. For example, "thing" is more general than "monkey wrench" and therefore it contains less information. We mention as example the Resnik algorithm [14] to disambiguate nouns. Resnik defines a semantic similarity measure between two nouns in WordNet based on the information content which has their closer common hypernym. To calculate information content, Resnik uses the frequency of appearance of the words in a given corpus.

These approaches depend directly on a single hierarchy (usually WordNet) so they do not consider other possible interpretations for a keyword. Some of these methods also depend on a proper corpus (as they do in [14] to measure *information content*). However, our disambiguation process combines the information provided by a pool of ontologies available on the Web, as well as WordNet, to provide the senses of a keyword; so we have more possible interpretations for the same keyword. Other difference with traditional methods is the use of a *semantic relatedness* measure based on Google, which avoids intrinsic dependence on a particular dictionary or corpus, and it also allows every keyword to take part in the disambiguation, process although they do not appear in the ontology pool.

There also have been efforts specifically oriented to disambiguation in a Web search. In [10] they classify search results into semantic classes defined by the different senses of a query term. In this system the disambiguation is made after retrieving the information, classifying it, so it is not an independent process. Senses are extracted from MultiWordNet ontology[14] (and uses hypernyms, hyponyms and glosses to characterize them). So they do not benefit from consider the different semantics that could be found in an ontology pool. The same observation can be applied to the system described in [17]. They uses WordNet to extend user queries with the synonyms and hyponyms of the selected senses.

Finally, in [7] is presented a method for automatic disambiguation of nouns in a domain specific corpora. It is also limited to WordNet in senses selection, but it uses Google searches to build a semantic measure between a sense of a term and a document collection.

7. CONCLUSIONS AND FUTURE WORK

In this paper we have identified some properties that are desirable for a disambiguation method in the Web. Then, we

[14]http://multiwordnet.itc.it/english/home.php.

have proposed a method that makes explicit the semantics of keywords by means of ontology terms, and fulfills those properties:

1. *It is an unsupervised method.* Our method does not use the large semantically tagged corpora needed for supervised learning approaches.

2. *It uses an ontology pool to provide candidate keyword senses*, instead of consulting just one lexical resource. To improve flexibility and avoid local maintenance, the ontologies of the pool are not downloaded but consulted from the Web directly.

3. *It uses a semantic relatedness measure based on Google frequencies.* Therefore, if a keyword is not found in any available ontology of the pool but is indexed by Google, it is not ignored either causes an error: It will be useful to disambiguate the other keywords.

4. *It follows an iterative algorithm to reduce the high computational cost.* Other techniques are also used, such as parallelism in performing searches among the Web (to retrieve ontology terms or Google frequencies).

Some initial tests show us that the chosen semantic relatedness measure behaves well. Also we have explored some particular disambiguation examples to illustrate the benefits of our proposal.

As future work, this method will be tested in a more systematic and exhaustive way. Ontology metrics will be studied in order to consider quality of the information sources in our system. We will also study the enrichment of our disambiguation algorithm by adding some techniques based on glosses. The application of our method in the construction of semantic queries will guide our next steps.

8. REFERENCES

[1] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientic American*, May 2001.

[2] A. Budanitsky and G. Hirst. Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures. In *Workshop on WordNet and Other Lexical Resources, in the North American Chapter of the Association for Computational Linguistics (NAACL-2001), Pittsburgh, PA*, 2001.

[3] R. Cilibrasi and P. M. B. Vitanyi. Automatic meaning discovery using google, December 2004. http://xxx.lanl.gov/abs/cs/0412098v1.

[4] G. Escudero, L. Màrquez, and G. Rigau. Boosting applied to word sense disambiguation. In R. L. de Mántaras and E. Plaza, editors, *Proceedings of ECML-00, 11th European Conference on Machine Learning*, pages 129–141. Springer Verlag, Heidelberg, DE, 2000.

[5] T. Finin, L. Ding, R. Pan, A. Joshi, P. Kolari, A. Java, and Y. Peng. Swoogle: Searching for knowledge on the Semantic Web. In *AAAI 05 (intelligent systems demo)*, July 2005.

[6] T. R. Gruber. Towards principles for the design of ontologies used for knowledge sharing. In N. Guarino and R. Poli, editors, *Formal Ontology in Conceptual Analysis and Knowledge Representation*. Kluwer Academic Publishers, 1993.

[7] I. Klapaftis and S. Manandhar. Google and wordnet based word sense disambiguation. In *Proceedings of the 22nd International Conference on Machine Learning Workshop on Learning and Extending Ontologies by using Machine Learning Methods*, 2005.

[8] C. Leacock and M. Chodorow. Combining local context and wordnet similarity for word sense identification. pages 265–283, 1998.

[9] M. Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *SIGDOC '86: Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26. ACM Press, 1986.

[10] E. W. D. Luca and A. Nrnberger. Ontology-based semantic online classification of documents: Supporting users in searching the web. In *Proc. of the European Symposium on Intelligent Technologies EUNITE*, 2004.

[11] E. Mena and A. Illarramendi. *Ontology-Based Query Processing for Global Information Systems*. Kluwer Academic Publishers, ISBN 0-7923-7375-8, 2001. June 2001.

[12] G. Miller. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11), nov 1995.

[13] G. A. Miller and W. G. Charles. Contextual Correlates of Semantic Similarity. *Language and Cognitive processes*, 1991.

[14] P. Resnik. Disambiguating noun groupings with respect to Wordnet senses. In D. Yarovsky and K. Church, editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 54–68. Association for Computational Linguistics, 1995.

[15] M. Rodriguez and M. Egenhofer. Determining semantic similarity among entity classes from different ontologies, 2003.

[16] J. Royo, E. Mena, J. Bernad, and A.Illarramendi. Searching the web: From keywords to semantic queries. In *Third International Conference on Information Technology and Applications (ICITA'05), July 4-7, 2005, Sydney (Australia)*, pages 244–249. IEEE Computer Society, ISBN 0-7695-2316-1, July 2005.

[17] C. Y. Shaung Liu and W. Meng. Word sense disambiguation in queries. In *14th ACM International Conference on Information and Knowledge Management (CIKM2005)*, pages 525–532. ACM Press, November 2005.

[18] A. P. Sheth, C. Ramakrishnan, and C. Thomas. Semantics for the semantic web: The implicit, the formal and the powerful. *Int. J. Semantic Web Inf. Syst.*, 1(1):1–18, 2005.

[19] S. P. Ted Pedersen, Satanjeev Banerjee. Maximizing semantic relatedness to perform word sense disambiguation, 2005. University of Minnesota Supercomputing Institute Research Report UMSI 2005/25.

[20] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics, 1995.