Word Semantics for Information Retrieval: Moving One Step Closer to the Semantic Web

Rada F. Mihalcea
Southern Methodist University
Dallas, Texas, 75275-0122
rada@seas.smu.edu

Silvana I. Mihalcea
Faculty of Economic Sciences
Cluj-Napoca, Romania
dmihalcea@email.ro

Abstract

The goal of the Semantic Web is to create a new form of Web content meaningful to computers. The Semantic Web aims to provide greater functionality, via intelligent tools such as information extractors, brokers, reasoning services or question answering systems. Semantics can be addressed at several levels. In this paper, we focus on the lowest level - word semantics on which other higher levels such as concept, paragraph, or document levels can be based upon. This model, which we call Word Semantics (WS), does not include the rich set of tags proposed by the XML/RDF standards. Nevertheless, this simpler WS format comes with a big advantage: it is possible with existing technologies and resources. Practically, this new model relies on understanding word meanings, identifying important named entities such as person, organization and others, and linking all this information via an external general purpose ontology, namely WordNet. With these features, we regard the WS model as a short but strong step toward the long term goal of a Semantic Web.

1 Introduction

With the huge and continuously increasing amount of information available on the Web, languages for representing data and knowledge occupy a central place in managing this tremendous quantity of data.

The current Web technology hinders further possible applications such as information extraction, reasoning or question answering. The main drawback of the original and most commonly used Web language (HTML) stands in its inability to encode the semantic information essential to a large range of Web applications. This is the main reason why new languages have been developed, such as the well known XML [20] and RDF [11] and their corresponding schemas, as well as extensions or applications of these languages including DAML [9], OIL [5], SHOE [7] and others.

The Web was initially designed for direct human processing. With its current structure, machine-based approaches to Web applications are not possible unless its content is transformed into a machine-readable format encoding semantic information. This is the main purpose of the Semantic Web, as defined by Tim Berners-Lee [2]: to create a new form of Web content meaningful to computers.

There are several layers to be addressed in data representation, to make possible the final goal of understanding the Web content.

- *Syntax.* This layer defines the syntax of the data, i.e. the format employed and allowed in data representations. It does not make any assumptions regarding the meaning of the data, nor it defines the relations among various objects on the web. This level is implemented by the XML language. One of the most important requirements for this level is the *syntactic interoperability*, as defined in [4]: the way data is represented in this layer should be as reusable as possible.

- *Semantics.* To actually gather the meaning of Web content, data should be understandable. Therefore, semantics are required to provide the information needed to understand and analyze data. RDF gives a framework and schemas for the implementation of this layer.

  A thorough description of the data made at this level enables mappings among terms and objects, and complicated information extraction and reasoning via an ontology vocabulary.

- *Ontology.* Once the data on the Web is understandable and represented in a machine-processible format, further information, not necessarily explicit in text, can be gathered via ontologies. Information such as *feline* is an *animal*, or *zip-code* is part of an *address*, can be encoded in general-purpose or domain-specific ontologies that would provide the basis for extracting information, reasoning around the stored data, answering questions, and other applications.

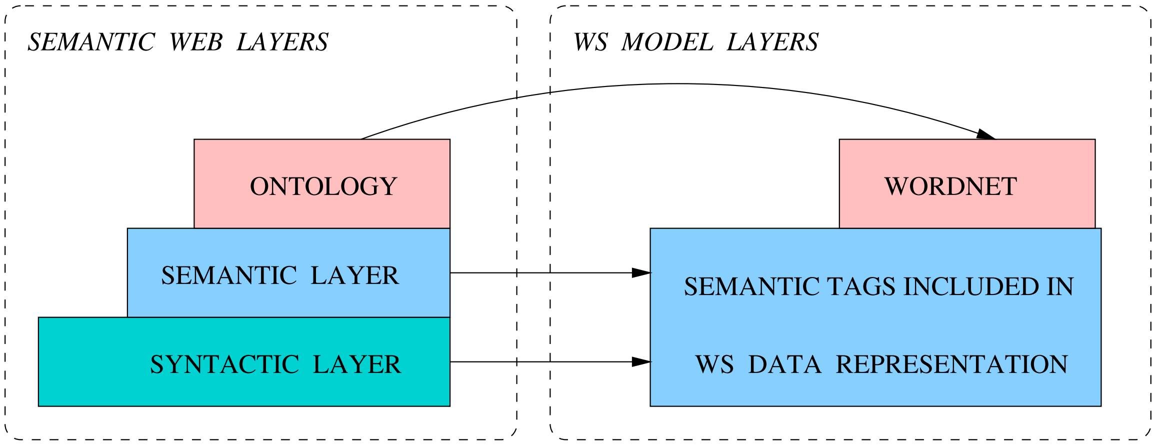

Figure 1. Layers in the Semantic Web and in the WS model

We propose in this paper a framework for transforming documents into machine processible forms, doable with existing technologies. We call this model Word Semantics (WS). As the name suggests, semantics are added at word level, which constitutes the finest level in text understanding, thereby enabling the use of semantic relations encoded in existing general purpose ontologies.

WS is simpler than RDF; it satisfies the requirement of *"markup for free"*, underlined by Hendler in [8]: users who are not logic experts (and the majority of users are not) should not even know that the web semantics exist. WS automatically labels free text, with no extra knowledge or markup required from the users. Practically, this new model relies on understanding word meanings, identifying important named entities such as person, organization and others, and linking all this information via an external general purpose ontology, namely WordNet. Figure 1 shows a layered representation of the Semantic Web, and the corresponding modules in our model.

The Semantic Web has to go beyond the simple bag of words approach currently used by search engines and get closer to the meaning of the texts. This need is exemplified by questions like *"What is the speed of ..."* with an answer in the form of *"... runs at a velocity of ..."*, or *"Why did X die?"* and the answer *"... committed suicide"*. There is no direct relation between these questions and their answers, but a hidden one, that stands in the synonymy of *speed* with *velocity*, or the more complex relation of *suicide* being a cause of *death* which is the event of *dying* [17]. This requires the understanding of both text and question, to be able to infer this type of semantic relations. Identifying information such as: *which is the closest hospital*, or *which are the available appointment times for a particular doctor*, as in the example given by Berners-Lee in [2], it is not possible unless the data posted on the Web is understood and manipulated meaningfully.

The use of semantics enables the extraction, from simple texts, of facts that are not stated but implied. For example, the sentence *"Extensive reconstruction of the M1 viaduct at Junction 34 is to be carried out for the Department of Transportation"* contains, even if not in an obvious representation, information such as *"M1 is a bridge"*, *"Junction 34 is a location"*, *"M1 is located closed to Junction 34"*, *"M1 needs repairs*, etc. This kind of implications can help a great deal towards the task of finding relevant information to given questions. Certainly, a question like *"What type of bridge is M1?"* cannot be answered without knowing that a *viaduct* is a *bridge*, and this information is readily encoded in already available semantic hierarchies. One of the most important things missing in current Web technologies is a link between the texts and external lexical resources encoding semantic information.

Several problems are arising here. First, it is not possible to even think of finding a relation of type *viaduct - bridge* without having the knowledge that *bridge* means here *construction* and not *cards game*. Hence, we have the problem of language ambiguity. Second, to answer a question like *"Where is M1 located?"* one needs to know that *Junction 34* is a location. Third, links between texts and an extensive knowledge base are needed for inferences such as *"reconstruction of M1"* $\rightarrow$ *"M1 needs repairs"*.

We propose a methodology of moving closer to the meaning of texts by enhancing the textual representations with semantic tags, and thus providing possible solutions to the problems described above.

As mentioned in [2], a large majority of the information we need is encoded along the specific-general lines. Among the multitude of lexical resources available today, WordNet [15] was designed to cover the *ISA* relation, and many other semantic relations such as *is-part*, *has-part* etc.

term	$\longrightarrow$	$word,\ stem,\ pos,\ semtag,$ $length,\ position$	
$word$	$\longrightarrow$	$STRING$	#word, as it appears in text
$stem$	$\longrightarrow$	$STRING$	#word root
pos	$\longrightarrow$	$NN\vert NNP\vert VB\vert JJ\vert...$ $PUNCT$	#part of speech tag
$semtag$	$\longrightarrow$	$WNoffset \mid NEtag$	
$WNoffset$	$\longrightarrow$	$INTEGER$	#synset id (offset) in WordNet
$NEtag$	$\longrightarrow$	$TPER \mid TLOC \mid TORG \mid$ $TDATE \mid TNUM \mid TMONEY \mid$ $TPCT \mid TSPEC$	#Named Entity classes
$position$	$\longrightarrow$	$INTEGER$	#position within the text
$length$	$\longrightarrow$	$INTEGER$	#one by default, larger than #one for compound words

Figure 2. Representation of a document term. Formal definition.

Additionally, WordNet is in its course of being transformed into a core knowledge base, by enhancing its word definitions with semantic tags and logic forms, enabling textual inferences [6]. Therefore, WordNet, with certain modifications, can play the role of the ontology needed for a form of Semantic Web.

In the following, we present in detail the WS model, which replaces the classic representation of texts with an enhanced semantic representation, closer to the meaning of texts. Next, we present the problems solved by the WS model and identify the technical challenges arising from the usage of this model; solutions for these difficulties are presented as well. We describe an implementation of the WS prototype and the results obtained. Finally, we present our conclusions and directions for further work.

2 Word Semantics model for enhanced text representations

As mentioned before, the WS model provides the means for enhancing simple texts with semantic information, which will enable the connection to an external general purpose ontology, as well as some forms of text understanding and improved information retrieval and extraction.

The two main components of a system aiming to *find information* are a **question**, or query, and a set of **documents** from which information is to be retrieved. The WS model proposes an enhancement of both these components, by replacing each keyword from the input question, and each term in the set of documents, with its semantic representation.

This section details on the formal definitions for a term, keyword and query, as defined in the WS model. The problems solved with the help of this model, as well as the new problems it raises are described in sections 3 and 4.

Figure 2 shows the formal definition of a document term. Keywords are represented using a similar format, but allowing for incomplete specifications. The keyword contains merely the same fields, namely

$$\textbf{keyword} \longrightarrow word,\ stem,\ pos,\ semtag$$

but any of these fields can be set to "$*$", meaning *"don't care"*. There are combinations which are not valid, for example it does not make sense to specify an *NEtag* and a *pos*. Also, the *position* and *length* fields of the term definition are not used within keywords. The valid combinations for a keyword are presented in Figure 3, together with various semantic operators that are now made possible.

One of the major improvements over classical retrieval systems, brought by the WS model, is the ability to use semantic operators, encoding relations such as *isa*, *subsumes*, *has-part* and others. This is possible due to the semantic tag added to each term, which practically solves the lexical and semantic ambiguity of the word and enables the connection to an external ontology. Some of these operators are shown in Figure 3 (most of them implemented in the WS prototype).

Any question asked against the index created with the documents will be transformed into a *query* that contains one or more keywords. Figure 4 gives the formal specification of a query. Notice that besides the classic operators, AND, OR and NEAR currently accepted by retrieval systems or search engines, two new locality operators are defined, PAR (paragraph) and SENT (sentence), encoding a *lexical* distance rather than *positional* distance.

Using these representations for document terms and question keywords, the matching at word level from the classic keyword-based retrieval technique is replaced with a matching between semantic forms.

$$
\begin{array}{rcll}
\textbf{keyword} & \longrightarrow & word & (1)\\
 & & |\, stem & (2)\\
 & & |\, word,\, pos & (3)\\
 & & |\, stem,\, pos & (4)\\
 & & |\, WNoffset & (5)\\
 & & |\, SemOp(WNoffset) & (6)\\
 & & |\, netag & (7)\\
SemOp & \longrightarrow & HYPERNYM \mid HYPONYM \mid & \\
 & & HYPE-HYPO \mid & \\
 & & SIBLING \mid RELATED & \\
HYPERNYM(offset) & \longrightarrow & \{offset_H \mid ISA(offset,\, offset_H)\} & \\
HYPONYM(offset) & \longrightarrow & \{offset_h \mid ISA(offset_h,\, offset)\} & \\
HYPE-HYPO(offset) & \longrightarrow & HYPERNYM(offset)\cup & \\
 & & HYPONYM(offset) & \\
SIBLING(offset) & \longrightarrow & \{offset_s \mid \exists\, offset_H \Rightarrow & \\
 & & offset \in HYPONYM(offset_H)\wedge & \\
 & & offset_1 \in HYPONYM(offset_H)\} & \\
RELATED(offset) & \longrightarrow & HYPE-HYPO(offset)\cup & \\
 & & SIBLING(offset) &
\end{array}
$$

Figure 3. Representation of a keyword. Formal definition.

3 Problems solved by the WS model

The format presented above brings large flexibility to retrieval systems and search engines. The classic technique of keyword indexing is now a sub-set of this new methodology: keyword-based matching can be obtained by using the format (1) for **keyword**, as presented in Figure 3. With the representation we propose, several other formats can be used for a keyword, like word stem, word meaning, or simply an NE tag. Different keyword formats allow for different applications, proving the benefits of word semantics over the simple keyword-based retrieval approach[1].

3.1 Keyword matching

Problem Identify documents based on exact keywords.

Solution As stated before, the classic technique of keyword matching can be used with this new approach by simply setting the **keyword** to be equal with *word*.

$$\textbf{keyword} \;\longrightarrow\; word,\, *,\, *,\, * \qquad (1)$$

Example Consider the problem of finding documents related to the query *"children" AND "care"*. From a set of 500 documents, 13 documents are found containing both keywords, using the above keyword format.

[1] The examples reported in this section are extracted from an index of 500 files from the *L.A. Times* collection.

3.2 Word stems

Problem When using the keyword matching approach, it might happen that relevant information is missed for the simple reason that the term in the document uses a different morphological inflection. For example, searching for *"car"* will not return documents containing *"cars"* and vice-versa. A solution for this problem was brought through the wildcard operator [1], used to replace one or more letters. The wildcard solves cases like *"cars"* and *"car"* by using *"car*"*, but this format allows for other words such as *"carrot"* or *"caress"*, which are not at all related with *"car"*. Another solution, currently used by many retrieval systems is word stemming (e.g. [18]). The problem is that irregular cases of morphological inflections, such as *"bring"* and *"brought"* cannot be solved by the wildcard, and usually by no other stemmers, but can be solved by a morphological stemmer which knows that *"brought"* is the past form of *"bring"*.

Solution The model proposed here gives a simple solution to this problem by using word stems. The keywords in this case will use the format:

$$\textbf{keyword} \;\longrightarrow\; *,\, stem,\, *,\, * \qquad (2)$$

Example Consider again the problem of finding documents related to the query *"children" AND "care"*, but this time we set the stem field in the keywords to *"child"*, respectively *"care"*. We now have increased recall, namely 16 documents are found containing both keywords.

$$
\begin{array}{rcl}
\textbf{query} & \longrightarrow & keyword \\
 & & \mid LexicalOp(keyword,\ [keyword],\ [query]) \\
LexicalOp & \longrightarrow & AND \mid NEAR \mid OR \mid \\
 & & PAR \mid SENT \\
AND(x_1,\ x_2,\ ...,\ x_n) & \longrightarrow & x_1,\ x_2,\ ...,\ x_n \textit{ belong to the same document} \\
NEAR(x_1,\ x_2,\ ...,\ x_n) & \longrightarrow & x_1,\ x_2,\ ...,\ x_n \textit{ are at a distance of max. N} \\
OR(x_1,\ x_2,\ ...,\ x_n) & \longrightarrow & \textit{at least one of } x_1,\ x_2,\ ...,\ x_n \textit{ is in the document} \\
PAR(x_1,\ x_2,\ ...,\ x_n) & \longrightarrow & x_1,\ x_2,\ ...,\ x_n \textit{ belong to the same paragraph} \\
SENT(x_1,\ x_2,\ ...,\ x_n) & \longrightarrow & x_1,\ x_2,\ ...,\ x_n \textit{ belong to the same sentence}
\end{array}
$$

Figure 4. Representation of a query. Formal definition.

3.3 Word and part of speech

Problem There are cases when words are morphologically ambiguous, like *"match"* which could mean *"contest"* as a noun, or *"to be compatible"* as a verb. For the query *"team" AND "match"*, a search returns 17 documents. From these, 13 documents contain *"match"* as a verb, and only 4 documents contain the noun *"match"*.
Solution This simple example shows that it is sometime useful to be able to specify the part of speech. This is made possible by the WS representation, with the following two keyword formats:

$$\textbf{keyword} \longrightarrow word,\ *,\ pos,\ * \tag{3}$$
$$\textbf{keyword} \longrightarrow *,\ stem,\ pos,\ * \tag{4}$$

We are able to specify the keywords part of speech, and therefore select from the very beginning the information we are interested in: either documents containing *"match"* as a verb, or documents referring to the noun *"match"*.
Example Consider a user interested in finding information related to *"team" and "match"*, where *"match"* is intended as the noun *"contest"*, and not the verb *"to be compatible"*. The search returns 4 documents for the noun *"match"*, much less with respect to the number of documents returned when the keyword is set to *match* only. Fragments from documents returned by this search are presented below.
Fragment from relevant document: *Thirteen French junior golfers, 8 boys and 5 girls, will play a* **match** *against a Southern California team Jan. 29-30 at Industry Hills.*
Fragment from irrelevant document: *When you have two teams as evenly* **matched** *as we are, the game is going to be decided by turnovers and big plays.*

3.4 Semantic Indexing

Problem It is well known that words can exhibit different meanings, depending on the context where they are used. One classic example in this sense is a search for *"plant"*, which returns documents containing the word *"plant"* with its *"factory"* meaning, as well as documents referring to *"plant"* as a *"living organism"*. In these cases it is important to know the keywords meaning in the input query, as well as the meaning of the terms in the collection of documents. At this stage, we consider the morphological ambiguity eliminated, with the procedure described previously. Therefore verb forms of the word *"plant"* are not considered.
Solution The ambiguity problem is solved with the following formats for a keyword.

$$\textbf{keyword} \longrightarrow word,\ *,\ *,\ WNoffset \tag{5}$$
$$\textbf{keyword} \longrightarrow *,\ *,\ *,\ ,WNoffset$$

Either one of these formats can be used to specify the meaning of keywords. This way, a search for *"plant"* meaning *"living organism"* should not return documents which refer to *"plant"* as *"factory"*.
Example Suppose the meaning of *"plant"* in the input query is meant to be *"living organism"*. Still, a basic search returns both senses, due to the inability of a classic retrieval system to select based on word meanings. There are 37 documents returned containing the noun *"plant"*.
Fragment from irrelevant document: *A Nuclear Regulatory Commission panel has conditionally approved the New Hampshire part of the emergency plan for the Seabrook nuclear power* **plant**.
Fragment from relevant document: *Cockleburs, a common* **plant** *in fields and riverbeds, along roadsides and near reservoirs, is easily recognized by its prickly burs.*

With the representation shown above, the search changes to *"plant,*,*,8864"* or simply *"8864"* (8864 is the offset number of the WordNet synset including "plant#2"). The first format proposed here identifies only documents containing the word with a given sense. Thus the number of documents returned is dramatically decreased: there are only 7 documents found by this search. On the other hand, the second format allows for the retrieval of documents including *"plant"* with its meaning of *"living organism"*, as well as its synonyms, such as *"flora"* or *"plant life"*.

3.5 Semantic Indexing with semantic operators

Problem Consider another application very common in this field: a user who wants to identify information related to a given topic, as for example *"field games"*. With the existing retrieval techniques, a search for *"field game"* will return a certain number of documents, but a lot of useful information might be missing, just because other related keywords such as *"football"*, *"soccer"* etc. were not specified.
Solution There are keywords formats, as described in Figure 3, designed for this type of problems. The new semantic operators defined in our model allow the retrieval of documents containing semantically related terms.

$$\begin{array}{rcl} \mathbf{keyword} & \longrightarrow & *, *, SemOp(WNoffset), * \quad (6) \\ SemOp & \longrightarrow & HYPERNYM \mid HYPONYM \mid \\ & & HYPE-HYPO \mid SIBLING \mid \\ & & RELATED \end{array}$$

Example A search for *"field game"* is now reformulated as a search for HYPO(298803) (298803 is the offset of the *"field game"* synset). It retrieves documents containing also *field hockey, hockey, football, hurling, baseball, baseball game, ball game, cricket, lacrosse, polo, pushball*, thus more information related to the *"field game"* topic of interest.

3.6 Searching for answers

Problem In many cases, users are interested in finding answers to specific questions, such as *"Who was the first president of USA?"* or *"Which is the capital of Romania?"*. With current retrieval systems or search engines, these questions do not constitute a special case for the search: users have to create a query and then search for the information they need. Even if one knows that the expected answer is a person name, or a location, there is no way to specify this within a query. Being able to add a keyword such as *"Person name"*, meaning that the documents, or pieces of documents returned, should contain the name of a person, would drastically decrease the number of documents and it would return more meaningful information.
Solution It is possible to indicate a Named Entity (such as person name, location, etc) as one of the keywords, with the following keyword format:

$$\mathbf{keyword} \longrightarrow *, *, NEtag, * \quad (7)$$

where *NEtag* can be one of the following NE tags: TPER, TLOC, TORG, TDATE, TNUM, TMONEY, TPCT, TSPEC.
Example Suppose that a user wants to find an answer to the following question *"How much does it cost to go from New York to London?"*. A basic query would be *"New York" AND "London"* and it returns 9 documents. Within the WS model, it is possible to indicate $MONEY$ as a keyword, as we expect the answer to be expressed as a money value. In this way, the number of documents is reduced to 5, thus a significant improvement. More details and experiments are presented in section 4.

Two other relevant examples are constituted by the following questions posed during the TREC-9 Q&A competition, which had no answer found by the best performing system. One such question was Question 204 *"What type of bridge is the Golden Gate Bridge?"*. In the WS model, an answer is easily found in the paragraph *"The Golden Gate Bridge may soon undergo modern wind stress tests, [...] to study how much wind it would take to damage the* **suspension bridge***"*, as *suspension bridge* is found in WordNet to be a type of *bridge*. A possible representation in our model is *HYPO(02336596) AND Golden Gate Bridge*, where 02336596 is the synset identifier for *bridge* in WordNet 1.7.

Another unsolved question was Question 377 *"At what speed does the Earth revolve around the sun?"*, which now finds an answer due to the semantic relations determined between *revolve* and *motion* (in WordNet, *to revolve* ISA *to move*, and the nominalization of *move* is *motion*): *"... it has to cancel Earth's motion around the Sun of* **30 km/sec***"*.

3.7 Searching for specific information

Problem Often, users need to find specific information, rather than entire documents, such as an answer to a question or a definition.
Solution It was previously shown that retrieving pieces of texts rather than entire documents increases the precision of the search [16]. The model proposed here enables the retrieval of pieces of documents by using lexical operators, as presented in Figure 4. PAR and $SENT$ are operators designed to retrieve paragraphs or sentences containing the input keywords.

Adding semantic information to both query and documents improves the quality of the search, and offers a novel solution for some other problems, such as finding information related to a topic, or retrieving answer types in a text. We have shown the formats to be used for a *term*, *keyword* and *query* within the WS model and we showed how these formats can improve a classic search.

4 Technical challenges

Several problems are arising at this point. Looking at the representation of a term, as shown in Figure 2, one can see that a significant amount of semantic information is required to make this model possible:

1. *Part of speech tag*. In the NLP community, part of speech tagging is considered an *almost solved* problems, as several taggers are available with accuracies over 95%. In the current implementation, we use Brill's part of speech tagger [3]; the reasons for choosing this tagger are its state-of-the-art precision and the fact that is publicly available.

2. *Word stem*. Once we know the *word* and the *part of speech*, it is easy to determine the word stem with the help of WordNet stemming functions[2].

3. *WordNet offset*. Finding the corresponding WordNet offset for a word means practically to identify its semantic sense. The disambiguation of word senses is one of the hardest problem in NLP. Previous results have shown that it is possible to disambiguate with 92% precision about 60% of the words in a text [13]. We have used this algorithm, briefly described below, in building the WS prototype.

4. *Named Entity tags*. There are several taggers designed for the recognition of named entities; we are using the tagger implemented by Dekang Lin, at University of Manitoba, with a precision closed to 90% [12].

We will detail below on the last two problems, namely word sense disambiguation and named entity recognition, as they pose complex issues.

4.1 A Word Sense Disambiguation algorithm

Word Sense Disambiguation (WSD) is a hard problem by itself and constitutes one of the central tasks in NLP. As mentioned before, language ambiguity represents a barrier in the way of building semantic representations of texts. Several approaches have been considered so far, including the Senseval effort [10], that aims at evaluating word sense disambiguation systems. The performance of the systems participating in this competition was in the range 70%-80%.

Out of the large range of existing WSD algorithms, we have decided to use a partial and highly accurate system which disambiguates about 60% of the nouns and verbs with over 90% precision [13]. The algorithm starts by identifying unambiguous words in text (such as proper nouns, words with only one sense) and based on these it incrementally builds a set of disambiguated words, using the semantic relations derived from WordNet, as well as information extracted from existing sense tagged corpora. This system was used and found useful in an information retrieval experiment, reported in [14].

[2]Specifically, the *morphstr* function from the WordNet 1.7 software package

4.2 Named Entity recognition

There are several approaches used for Named Entity (NE) recognition. The performance for this problem is significantly higher than for WSD, and usually NE systems have precisions of over 90%. We are using the tagger implemented by Dekang Lin, at University of Manitoba [12].

A series of experiments were conducted to verify the impact of using NE tags in a retrieval system. We have identified the named entities in 125,000 documents (again from the *L.A.Times collection*), and then index these documents including the NE tags. Next, a test benchmark was created with 75 questions known to have an answer in the given set of documents (this information was only required for the purpose of measuring the system recall). Comparative experiments have shown that indexing named entities almost doubles the precision of the system, while the recall remains the same (basically, a large reduction in the number of returned documents is obtained).

5 Implementation of a WS prototype

We have implemented a system which encodes part of the WS representation format. It employs (a) part of speech tagger, (b) stemmer and (c) a word sense disambiguation algorithm to create a rich semantic representation for both document terms and question keywords.

The prototype was tested using the Cranfield collection, containing 1400 documents, and a set of 50 questions associated with this collection.

To measure the impact of the WS model over the classic retrieval model, three measures are used to evaluate the system performance: (1) *precision*, defined as the number of relevant documents retrieved over the total number of documents retrieved; (2) *recall*, defined as the number of relevant documents retrieved over the total number of relevant documents found in the collection and (3) *F-measure*, which combines both precision and recall into a single formula [19]:

$$F_{measure} = \frac{(\beta^2 + 1.0) * P * R}{(\beta^2 * P) + R}$$

where P is the precision, R is the recall and β is the relative importance given to recall over precision. We consider both precision and recall of equal importance, and therefore this factor β is set to 1.

Every document term is replaced with the format shown in Figure 2, and every question keyword is replaced with the format (5) from Figure 3. Documents are indexed, and queries are formed and run against this index.

Tests over the entire set of 50 questions led to 0.20 precision and 0.22 recall using the classic retrieval methodology, respectively 0.23 precision and 0.29 recall when a meaning-based (*WNoffset* based) indexing is used. The relative gain of the meaning-based indexing respect to the word-based indexing is 31% increase in recall and 15% increase in precision.

The conclusion of these experiments is that indexing by meanings actually improves effectiveness. Moreover, this is the first time to our knowledge when a WSD algorithm for open text is actually used to automatically disambiguate a collection of texts prior to indexing, with a disambiguation accuracy high enough to actually increase the performance of the IR system.

An issue to be addressed here is the efficiency of such a system: we have introduced a WSD stage into the classic retrieval process and it is well known that WSD algorithms are usually computationally intensive. On the other side, the disambiguation of a text collection is a process which can be highly parallelized, therefore this does not constitute a real problem.

6 Conclusions

The full understanding of text is still an elusive goal. The scope of the Semantic Web is to translate current Web content into a format meaningful to computers. Several languages have been developed for this purpose, including XML, RDF, DAML, OIL, SHOE and others.

We proposed in this paper the *Word Semantics* model, which is simpler than XML or RDF, but it comes with the big advantage of being feasible with existing technologies and resources. Practically, this new model relies on understanding word meanings, identifying important named entities such as person, organization and others, and linking all this information via an external general purpose ontology, namely WordNet. With these features, we regard the WS model as a short but strong step toward the long term goal of creating a Semantic Web.

References

[1] Altavista, 2000. Digital Equipment Corporation, "http://www.altavista.com".

[2] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 1(501), May 2001.

[3] E. Brill. A simple rule-based part of speech tagger. In *Proceedings of the 3rd Conference on Applied Natural Language Processing*, Trento, Italy, 1992.

[4] S. Decker, S. Melnik, F. van Harmelen, D. Fensel, M. Klein, J. Broekstra, M. Erdmann, and I. Horrocks. The Semantic Web: the roles of XML and RDF. *IEEE Intelligent Systems*, 15(5):2–13, September/October 2000.

[5] D. Fensel, F. van Harmelen, I. Horrocks, D. McGuinness, and P. Patel-Schneider. OIL: and ontology infrastructure for the Semantic Web. *IEEE Intelligent Systems*, 16(2):38–45, March/April 2001.

[6] S. Harabagiu, G. Miller, and D. Moldovan. WordNet 2 - a morphologically and semantically enhanced resource. In *Proceedings of SIGLEX-99*, pages 1–8, Univ. of Maryland, June 1999.

[7] J. Heflin and J. Hendler. A portrait of the Semantic Web in action. *IEEE Intelligent Systems*, 16(2):54–59, March/April 2001.

[8] J. Hendler. Agents and the Semantic Web. *IEEE Intelligent Systems*, 16(2):30–37, March/April 2001.

[9] J. Hendler and D. McGuinness. The DARPA agent markup language. *IEEE Intelligent Systems*, 16(6):67–73, January/February 2000.

[10] A. Kilgarriff and M. Palmer, editors. *Computer and the Humanities. Special issue: SENSEVAL. Evaluating Word Sense Disambiguation programs*, volume 34, April 2000.

[11] O. Lassila and R. Swick. Resource description framework (RDF) model and syntax specification. In *W3C Recommandation*, February 1999. available online at http://www.w3.org/TR/REC-rdf-syntax/.

[12] D. Lin. Principar - an efficient, broad-coverage, principle-based parser. In *In Proceedings of the Fifteenth International Conference on Computational Linguistics COLING-ACL '94*, pages 42–48, Kyoto, Japan, 1994.

[13] R. Mihalcea and D. Moldovan. An iterative approach to word sense disambiguation. In *Proceedings of FLAIRS-2000*, pages 219–223, Orlando, FL, May 2000.

[14] R. Mihalcea and D. Moldovan. Semantic indexing using WordNet senses. In *Proceedings of ACL Workshop on IR & NLP*, Hong Kong, October 2000.

[15] G. Miller. Wordnet: A lexical database. *Communication of the ACM*, 38(11):39–41, 1995.

[16] D. Moldovan and R. Mihalcea. Using WordNet and lexical operators to improve Internet searches. *IEEE Internet Computing*, 4(1):34–43, 2000.

[17] D. Moldovan and V. Rus. Logic form transformations of WordNet and its applicability to Question Answering. In *Proceedings of the 39th Annual Meeting of the Association for Computational Lingusitics (ACL-2001)*, pages 394–401, Toulouse, France, July 2001.

[18] M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[19] C. Van Rijsbergen. *Information Retrieval*. London: Butterworths, 1979. avialbale on-line at http://www.dcs.gla.ac.uk/Keith/Preface.html.

[20] XML language, 2000. Introduction to XML and XML schemas available at www.xml.com.